XIV Conferencia Latinoamericana de Informática
17avas. Jornadas Argentinas de Informática
e Investigación Operativa
Buenos Aires, Setiembre 1988.

"Un Sistema de Compilación, Ejecución y Análisis de Redes de Petri Extendidas para Computadoras IBM-PC Compatibles"

Autores: Ludueña Verónica
Oyarzabal Juan Carlos

Director: Gallard Raúl Héctor

UNIVERSIDAD NACIONAL DE SAN LUIS
Ejército de los Andes 950, Local 106 - 5700 San Luis ARGENTINA
Teléfonos (0652) 25882/25108/26744/26747/26888
Telex 58125 UNSL AR

RESUMEN

En esta presentación se describe la implementación de un Sistema para la compilación, ejecución y análisis de Redes de Petri Extendidas sobre IBM-PC compatibles.

El compilador admite como entrada la definición de una Red de Petri, en forma gráfica o explicitandolo como ente matemático.
El proceso de compilación produce una representación interna ejecutable de la red.

El ejecutor procede a ejecutar la representación interna de la Red de Petri en forma visible (gráfica) o invisible (sin salida gráfica) al usuario, según su elección.

Bajo ambas opciones el analizador responde a las preguntas del análisis, recurriendo a las conocidas técnicas del árbol de alcanzabilidad o de las ecuaciones matriciales.
En el caso de utilizar la definición extendida de ejecución de una red podrán obtenerse respuestas adicionales a las preguntas del análisis.

1. INTRODUCCION

Este trabajo nace como consecuencia de la tarea de los autores y del grupo de Sistemas Operativos de la U.N.S.L. en el estudio e implementación de Software para el control de sintaxis, ejecución y análisis de Redes de Petri.

El Sistema desarrollado sobre una IBM-PC Compatible, en Quick Basic, constituye una facilidad automática para la modelización de sistemas que exhiban actividades concurrentes y asincrónicas.

Además, la implementación de la definición extendida de la ejecución de una Red de Petri, en base a un trabajo preliminar publicado en 1987 [8] amplía su capacidad de modelización permitiendo una más fiel representación del comportamiento de sistemas reales y una mejor comprensión de

los mismos, al responder a un conjunto importante de preguntas para las cuales el enfoque tradicional no tiene respuesta.

2. SOPORTES DE LA IMPLEMENTACION

Se decidió implementar el sistema sobre una computadora IBM PC AT compatible a efectos de hacer altamente accesible, este software, a una variedad de usuarios.
El equipo cuenta con 1 Mb de memoria principal, dos discos winchester de 20 Mb cada uno, impresora y tarjeta EGA. No obstante menores requerimientos son necesarios para correrlo, existiendo versiones que pueden ejecutarse en configuraciones mínimas de XT compatibles.

Se decidió utilizar un Compilador Quick Basic por la bondad de este lenguaje en su interface gráfica, por su satisfactoria estructuración, por sus facilidades para el desarrollo de Software y por su velocidad de ejecución.

3. CONCEPTOS PREVIOS

Una Red de Petri es un multigrafo, bipartito, dirigido, que tiene dos tipos de nodos: "sitios" y "transiciones". Los primeros modelizan las condiciones (estado) del sistema y las últimas modelizan los eventos que pueden darse en el mismo.
Sitios y transiciones se vinculan a través de arcos, conformando asi una estructura.

En los sitios pueden residir "tokens", en este caso se dice que la red esta marcada y se llama "marca" a una particular distribución de tokens en la red.
La distribución de tokens gobierna la ejecución de una red, habilitando o deshabilitando transiciones y permitiendo o no, consecuentemente, su disparo.
La ejecución de la red consiste en una secuencia de transiciones de estado producida por el disparo de transiciones.
Más precisamente:

Dada una Red de Petri marcada $M = \langle S,u \rangle$

donde

u es una marca inicial y S la estructura definida por;

$S = \langle P,T,I,O \rangle$

con
- $P = \{p1,p2,\ldots,pn\}$ Conjunto de n sitios, $n \geq 0$
- $T = \{t1,t1,\ldots,tm\}$ Conjunto de m transiciones, $m \geq 0$
- $I : T \dashrightarrow P'$ Función de entrada
- $O : T \dashrightarrow P'$ Función de salida
- P' Conjunto de todas las bolsas posibles formadas con elementos de P.

el espacio de estados R(S,u) de la red, tiene por elementos a todas la marcas alcanzables desde dicha marca inicial u.

Este conjunto de alcanzabilidad, R(s,u), tiene una representación finita por medio del árbol de alcanzabilidad. Cada nodo de dicho árbol representa una marca alcanzable, o un conjunto de marcas alcanzables si la red es no acotada (presencia del símbolo w en el árbol).

Además para cada uno de estos nodos se determina unívocamente el conjunto Tau contenido en T, de transiciones habilitadas. Esto es cierto aun en los casos para los cuales en el nodo en cuestión se halle presente el símbolo w, pues la presencia de "tokens" extra en un sitio no afecta la habilitación ni el disparo de las transiciones.

En consecuencia para una marca particular (estado de la red) existe un único conjunto Tau de transiciones que pueden disparar.

Como ejemplo, la siguiente red modeliza la sincronización entre un proceso productor y otro consumidor a través de un buffer no acotado:

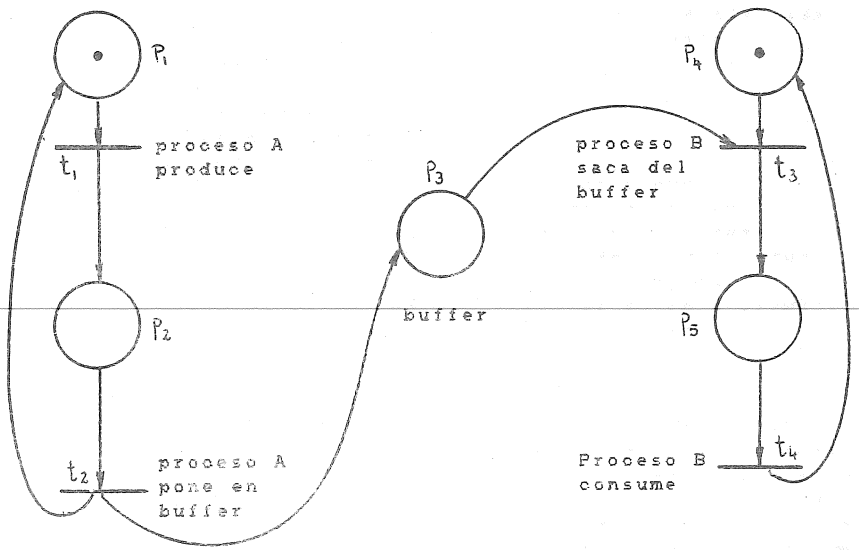

Figura 1

Habiendo remarcado estos conceptos se describen a continuación los componentes del software.

4. COMPONENTES DEL SOFTWARE

4.1. EL COMPILADOR

Este componente produce una representación interna ejecutable de la Red de Petri admitiendo la definición de la red en forma gráfica o como ente matemático.

4.1.2. ENTRADA DE LA DEFINICION DE LA RED EN FORMA GRAFICA

La entrada en forma gráfica se realiza por medio de comandos implementados a través de las teclas de función del teclado IBM compatible.

A continuación se describe la gramática (1) del lenguaje de definición de la red en forma gráfica:

```
                    <grafo> ::= <dibujo de nodos>
          <dibujo de nodos> ::= <trans. horiz.> | <sitio> |
                                <trans. vert.> |
                                <trans. sudoeste> |
                                <trans. sudeste> |
                                <cambio a dibujo de arcos>
              <trans. hor.> ::= F1
                    <sitio> ::= F2
             <trans. vert.> ::= F3
          <trans. sudoeste> ::= F4
           <trans. sudeste> ::= F5
<cambio a dibujo de arcos> ::= F6 <dibujo de arcos>
          <dibujo de arcos> ::= <punto inicial> |
                                <punto intermedio> |
                                <punto final común> |
                                <punto final inhibidor> |
                                <redibujar nodos> |
                                <cambio a marca inicial>
            <punto inicial> ::= F1
        <punto intermedio> ::= F2
      <punto final común> ::= F3
  <punto final inhibidor> ::= F4
        <redibujar nodos> ::= F5 <dibujo de nodos>
<cambio a marca inicial> ::= F6 <marca inicial>
           <marca inicial> ::= <entero sin signo>
                               <marca inicial> |
                               <entero sin signo>
```

El término "trans. sudoeste" (o sudeste) describe una transición con inclinación de 45 grados orientada al sud oeste (sud este).
El proceso de entrada, que incluye el análisis sintáctico, se realiza en forma interactiva y amigable al usuario, implementado por medio de ventanas que lo guian acorde con sus necesidades y siguiendo pasos lógicamente estructurados (dibujo de nodos, dibujo de arcos, redibujo de nodos y consecuentemente de arcos).
La salida del compilador en este caso es una representación gráfica de la red en la pantalla y una traducción, transparente al usuario, de las estructuras de información para el soporte de la red (multilistas encadenadas).

(1) Tal como ocurre con los lenguajes de programación, donde las gramáticas tipo 2 no permiten controlar todas las características del mismo, se hace necesaria aqui la existencia de tablas de símbolos para efectuar los controles no provistos por la gramática. De esta forma se controla la asociación de marcas a sitios definidos en la red y sólo a ellos.

4.1.3. ENTRADA DE LA DEFINICION DE LA RED COMO ENTE MATEMATICO

La entrada como ente matemático se hace simplemente definiendo una estructura de Red de Petri y una marca inicial ⟨S,u⟩.
A continuación se describe la gramática (2) del lenguaje de definición de la red como ente matemático:

```
        ⟨Red de Petri⟩ ::= ⟨estructura⟩ (⟨marca inicial⟩)
          ⟨estructura⟩ ::= ⟨sitios⟩ ⟨transiciones⟩
                           ⟨fción de entrada⟩
                           ⟨fción de salida⟩
                ⟨sitios⟩ ::= ⟨cantidad de sitios⟩ | ⟨lambda⟩/
          ⟨transiciones⟩ ::= ⟨cantidad de trans.⟩ | ⟨lambda⟩/
     ⟨fción de entrada⟩ ::= I⟨id. trans.⟩=⟨bolsa de sitios⟩ |
                           ⟨lambda⟩/
      ⟨fción de salida⟩ ::= O⟨id. trans.⟩=⟨bolsa de sitios⟩ |
                           ⟨lambda⟩/
   ⟨cantidad de sitios⟩ ::= ⟨entero sin signo⟩/
   ⟨cantidad de trans.⟩ ::= ⟨entero sin signo⟩/
                ⟨lambda⟩ ::=
             ⟨id. trans.⟩ ::= ⟨entero sin signo⟩
      ⟨bolsa de sitios⟩ ::= ⟨entero sin signo⟩,
                           ⟨bolsa de sitios⟩ |
                           ⟨entero sin signo⟩/
         ⟨marca inicial⟩ ::= ⟨entero sin signo⟩,
                           ⟨marca inicial⟩ |
                           ⟨entero sin signo⟩ | ⟨lambda⟩
```

Asi por ejemplo dada la Red de Petri M = ⟨S,u⟩ con

S = ⟨P,T,I,O⟩ donde

P = {p1,p2,p3,p4}
T = {t1,t2}

I : T ---⟩ P' tal que

I(t1) = {p1}
I(t2) = {p3,p3}

O : T ---⟩ P' tal que

O(t1) = {p1,p3}
O(t2) = {p2,p4,p4}

y u =(1,0,0,0)

sería especificada de la siguiente forma:

4/2/I1=1/I2=3,3/O1=1,3/O2=2,4,4/(1,0,0,0)

La salida del compilador en este caso consiste simplemente en la traducción, de la entrada a las estructuras de información para soportar la red.

(2) Al igual que en (1) las tablas de símbolos permitirán controlar la asociación de marcas a sitios definidos en la red y sólo a ellos, asi como también la correspondencia establecida por las funciones de entrada y salida entre transiciones y sitios definidos por la estructura y sólo entre ellos.

4.2. EL EJECUTOR

Este módulo es el encargado de implementar la ejecución de la red previamente compilada.

En ambos casos, entrada gráfica o como ente matemático, el ejecutor permite al usuario optar por la acotación de sitios a su elección y por efectuar o no tests de conservatividad y vitalidad para un número especificado de disparos de transiciones.
También se ha previsto registrar opcionalmente la secuencia de disparos y/o la de estados por los que transita la red, que pueden ser listadas luego de la ejecución.
Una vez finalizada la ejecución (por haber ejecutado el número de disparos especificados o haber llegado a un estado de muerte) se puede optar por una nueva ejecución para una nueva marca inicial y un distinto número de disparos.
También, opcionalmente, podrá ejecutarse la red acorde con la extensión en la definición de la ejecución (Ver punto 4.3.1.1).

En el caso de entrada gráfica, la red muestra en pantalla el cambio de las marcas, la habilitación, deshabilitación y el disparo de transiciones mediante un adecuado cambio de colores.
Los tokens se muestran como pequeños círculos, si la cantidad de ellos en un sitio es menor o igual a 9, o como un entero de hasta tres dígitos en otro caso.
La ejecución gráfica puede hacerse en modo "trace" (paso a paso) o automáticamente, con una demora elegida por el usuario entre un disparo y el siguiente.
En este caso una vez ejecutada la red, también puede re-entrarse en la etapa de dibujo de nodos (y consecuentemente de arcos) para introducir modificaciones de interés antes de una nueva ejecución.
También, opcionalmente, pueden salvarse en disco las estructuras soporte y las coordenadas de los nodos del grafo para un posterior re-dibujo automático y su correspondiente ejecución y análisis.
Por último, el sistema se provee con algunas redes previamente compiladas y almacenadas en disco que representan casos típicos de sincronización de procesos. Esta facilidad es de utilidad para el principiante en el tema.

4.3. EL ANALIZADOR

Este módulo responde a las preguntas del análisis, opcionalmente, mediante una de dos técnicas; el árbol de alcanzabilidad y las ecuaciones matriciales.

4.3.1. ARBOL DE ALCANZABILIDAD

El árbol de alcanzabilidad representa todos los estados posibles por los que transita la red y describe además las secuencias posibles de disparos que hacen alcanzables tales estados. Por ejemplo, el siguiente árbol corresponde a la red de la la Figura 1:

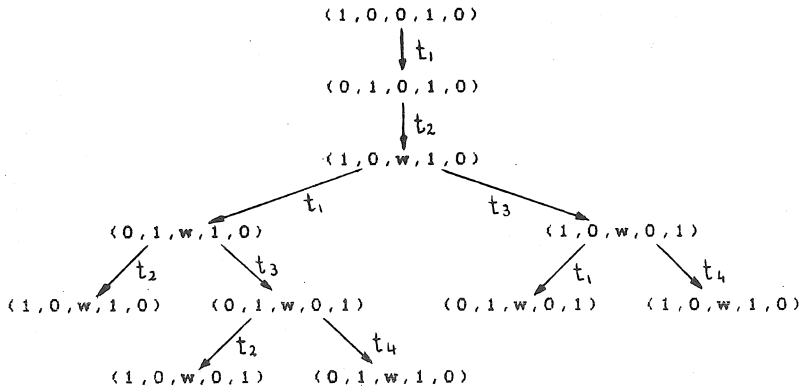

Figura 2

donde:

$R(S,u) = \{ (1,0,w,1,0), (0,1,w,1,0), (1,0,w,0,1), (0,1,w,0,1) \}$

con $w \geq 0$

En general el árbol de alcanzabilidad de una Red de Petri es un árbol donde el semigrado exterior de cada nodo es variable dependiendo del número de transiciones habilitadas en cada estado de la red.
A los efectos de economizar almacenamiento se ha recurrido al método de las transformadas de Knuth para encontrar una representación del árbol como árbol binario.

Por medio de la técnica del árbol de alcanzabilidad pueden responderse las preguntas del análisis en lo referente a: seguridad, acotabilidad y cobertura. En muchos casos también es posible responder a las preguntas de alcanzabilidad y vitalidad (Peterson [4]).

4.3.1.1. EXTENSION DE LA DEFINICION DE LA EJECUCION, SU IMPLEMENTACION VIA EL ARBOL DE ALCANZABILIDAD

Según el trabajo preliminar [8] se ha implementado esta extensión acorde a lo siguiente:

i)

En la rutina de generación del árbol de alcanzabilidad se extiende el concepto de nodo a fin de que cada uno de ellos esté constituido por el par:

(u,H) = (Marca corriente, Vector de Habilitación Corriente)

ii)

Se cuenta además con otra rutina que genera, recorriendo el árbol y en base al conocimiento del usuario, el correspondiente vector de ponderación W para ese estado de la red, a partir del cual se construirá el vector de probabilidades W' el que reemplazara en el nodo en cuestión al vector de habilitación, quedando ahora el nodo como sigue;

Nodo del árbol = (u,W')
= (Marca corriente, Vector de Prob. asociado con la marca)

iii)

Por último, existe una rutina de ejecución alternativa de la Red de Petri de modo que en cada estado de la misma, se recupere el vector W' asociado a la marca corriente para luego invocar a otra rutina que aplica el método de Montecarlo a fin de decidir que transición será disparada.

4.3.2. ECUACIONES MATRICIALES

Por medio de la búsqueda de soluciones a ecuaciones matriciales, que describen la estructura de una Red de Petri, pueden abordarse los problemas del análisis en lo concerniente a la alcanzabilidad y la conservatividad estricta y no estricta.
En este último caso, si existe solución, se provee un vector no nulo de ponderación asociado a los sitios para el cual la red se mantiene conservativa ([4] Peterson).

Esta técnica ha sido implementada utilizando una variante del método de Gauss.

5. CONCLUSIONES

El presente artículo resume las características esenciales del Trabajo Final, para optar por el Título de Licenciado en Cs de la Computación, de los autores.

Vale aqui remarcar otros puntos de interés:

1)

El sistema constituye una facilidad versatil para encarar la modelización de sistemas, computacionales o no, que exhiban actividades asincrónicas y concurrentes.
Esto se logra a través de salidas gráficas, si la cantidad de nodos de cada tipo no es mayor que 50.
Las salidas consisten en la visualización de la ejecución de la red y en el diagnóstico de la misma a través del análisis.

Otra opción permite la ejecución en memoria, principal o secundaria, no visualizada de la red. Esto se obtiene, cuando la misma se entra como ente matemático, permitiéndose entonces ejecutar redes de mayor envergadura, liberando al usuario de las restricciones impuestas por el tamaño de la pantalla.

En ambos casos podrán obtenerse listados sobre los resultados de la ejecución y del análisis.

2)

A través de la implementación de la extensión tal como lo plantea el autor del artículo original [8], se permite un comportamiento del modelo más granularmente afín con el estado del sistema modelizado en cada instancia de la red.

En consecuencia una acumulación de tokens en un sitio de la red, modelizaría ahora más fielmente la formación de una cola cuyos parámetros pueden registrarse para análisis posteriores.

Además, La asignación de una probabilidad de disparo igual a 1 ó 0 para una de las transiciones habilitadas, describiría naturalmente una política de prioridades sin necesidad de recurrir al uso de arcos inhibidores.

Con respecto al problema de la alcanzabilidad se está ahora en condiciones de responder no sólo que cierto estado es alcanzable, sino que además se precisaría más la respuesta diciendo con qué probabilidad se arribaría a ese estado luego de k disparos.
Esto es asi pues,

si u' = d(u,Sigma)

donde d es la función extendida de próximo estado y Sigma es una secuencia de disparos de longitud k, la probabilidad de arribar a u' estaría dada por el producto de las probabilidades asociadas con cada transición disparada, en las sucesivas marcas (estados) por los que pasa la red.

6. RECONOCIMIENTOS

El presente trabajo no hubiera sido posible sin la cuota de motivación que sobre el mismo aportara el Dr Alan J Harget de la University of Aston in Birmingham (U.K.) quién en 1982 introdujo al director en el tema.

Se agradece también en especial a las Licenciadas Susana Graciela Barbenza y Mirtha Susana Escudero quienes desarrollaron una versión inicial de Software de Redes Petri para la PDP-11/34 que los autores han extendido y convertido a IBM PC compatibles.

En general se agradece también las sugerencias aportadas por distintos integrantes del grupo de proyecto en Sistemas Operativos de la Universidad Nacional de San Luis.

7. BIBLIOGRAFIA

[1] C. Petri, "Kommunikation mit Automaten" , Ph. D. dissertation, University of Bonn, West Germany, 1962.

[2] J. Sifakis, "Use of Petri Nets for Performance Evaluation",Procedures of the Third International Workshop on Modelling and Performance Evaluation of Computer Systems, North Holland 1977, pages 75-93.

[3] J. Peterson, "Petri Nets", Computer Surveys, Volume 9, Number 3, September 1977.

[4] J. Peterson, "Petri Net Theory and the Modeling of Systems", Prentice Hall, 1981.

[5] H. J. Genrich and K. Lautenbach, "System Modelling with High-Level Petri Nets", Theoretical Computer Science 13, 1981, pages 109-136.

[6] R. Gallard, "A Set of Petri Net Routines for System Modeling", MSc dissertation, University of Aston in Birmingham, Birmingham, U.K., July 1982.

[7] M. Escudero, S. Barbenza, R. Gallard, "Un Sistema para la Definición, Ejecución y Análisis de Redes de Petri" 15 JAIIO, U.N.S, Nov 1985.

[8] R. Gallard, "An Extension in the Definition of a Petri Net Execution", Computer Journal, Vol. 30, Nro 1, Feb. 1987.