Sistemas de reescritura de redes*

M. Llorens†, J. Oliver†
†Univ. Politécnica de Valencia. Depto. Sistemas Informáticos y Computación.
Camino de Vera, s/n, E-46022 Valencia, Spain
{mllorens, fjoliver}@dsic.upv.es

Resumen

El artículo se centra en la modelización de sistemas concurrentes sujetos a cambios dinámicos utilizando extensiones de las redes de Petri. Comenzamos recordando la noción de sistema de reescritura de redes y probamos que tales sistemas tienen el mismo poder computacional que la máquina de Turing. Una subclase de los sistemas de reescritura de redes son las redes reconfigurables, para las cuales demostramos que su expresividad y la de las redes de Petri es equivalente, pero con las redes reconfigurables podemos modelizar más fácil y directamente sistemas que cambian su estructura dinámicamente.

Palabras clave: sistemas concurrentes, redes de Petri, reescritura de grafos, cambios dinámicos.

1. Introducción

En [1] y [6] introducimos los sistemas de reescritura de redes que mejoran la expresividad del modelo básico de las redes de Petri [7, 8] permitiendo representar cambios dinámicos en sistemas concurrentes. El modelo propuesto se basa en dos líneas de investigación: la primera estudia la manera de fusionar redes de Petri con gramáticas de grafos [4, 5, 9], mientras que la segunda, representada en particular por las redes automodificantes de R. Valk [10, 11], estudia redes de Petri cuyas relaciones de flujo pueden cambiar en tiempo de ejecución. Ambas propuestas dan lugar a modelos expresivos que aportan indudables ventajas con respecto a los modelos ya existentes. Sin embargo, la mayoría (si no todas) las propiedades básicas decidibles de las redes de Petri (acotabilidad de lugares, alcanzabilidad, interbloqueo y vivacidad) se pierden en estos modelos. Como consecuencia no pueden construirse herramientas automáticas de verificación para dichos modelos extendidos. Las redes reconfigurables [1, 2, 3, 6], una subclase de sistemas de reescritura de redes, intentan

*Este trabajo ha sido parcialmente financiado por *CICYT TIC 2001-2705-C03-01, por Acción Integrada Hispano-Alemana HA2001-0059 y por Proyecto de Investigación UPV 7176.*

combinar las características más significativas de las dos aproximaciones de manera que las propiedades fundamentales sean decidibles, con lo que en este modelo será factible la verificación automática.

En la Sección 2 describimos los sistemas de reescritura de redes de [1, 6] que combinan las redes de Petri con sistemas de reescritura de grafos. La idea es describir una configuración del sistema como una red de Petri y un cambio de configuración como una regla de reescritura de grafos que consiste en reemplazar parte del sistema (que hace *matching* con la parte izquierda de la regla de reescritura) por otra (que viene dada por la parte derecha de la regla). Una red reconfigurable es un sistema de reescritura de redes en el que el cambio de configuración se limita a la modificación de las relaciones de flujo de los lugares implicados en la regla de reescritura. La traducción de este modelo a redes de Petri es automática. Esta equivalencia asegura que todas las propiedades fundamentales de las redes de Petri son también decidibles para las redes reconfigurables, siendo posible la obtención de herramientas de verificación automática para este modelo. Por el contrario, la clase de los sistemas de reescritura de redes tiene el poder expresivo de la máquina de Turing con lo que la construcción de herramientas de verificación automática no es posible para dicha clase. En la Sección 3 presentamos una implementación de las redes reconfigurables mediante redes de Petri. La simulación de una máquina de Turing utilizando sistemas de reescritura de redes se presenta en la Sección 4. Finalmente, en la Sección 5 se muestran las conclusiones.

2. Sistemas de reescritura de redes

En esta sección presentamos el modelo de los sistemas de reescritura de redes de [1, 6]. Son necesarias las siguientes definiciones previas: Si $R \subseteq X \times Y$ es una relación binaria, se dice que $X'R = \{y \in Y \mid \exists x \in X' \ (x,y) \in R\}$ denota la *imagen* de $X' \subseteq X$ y $RY' = \{x \in X \mid \exists y \in Y' \ (x,y) \in R\}$ denota la *imagen inversa* de $Y' \subseteq Y$; el *dominio* de R es entonces $Dom(R) = RY$ y el *codominio* de R es $Cod(R) = XR$. Una *red de Petri* [7, 8] es una tupla $\mathcal{N} = (P, T, F)$ donde: $P = \{p_1, p_2, \ldots, p_m\}$ es un conjunto finito de lugares, $T = \{t_1, t_2, \ldots, t_n\}$ es un conjunto finito de transiciones ($P \cap T = \emptyset$, $P \cup T \neq \emptyset$) y $F : (P \times T) \cup (T \times P) \to \{0, 1, 2, 3, \ldots\}$ es un conjunto de arcos ponderados (relación de flujo). Una *red Petri marcada* es un par $(\mathcal{N}, M_0)$ donde $\mathcal{N}$ es una red de Petri y $M_0 : P \to \{0, 1, 2, 3, \ldots\}$ es el marcado inicial. Sean $\mathcal{N} = (P, T, F)$ y $\mathcal{N}' = (P', T', F')$ dos redes de Petri. Se dice que $\mathcal{N}$ y $\mathcal{N}'$ son *isomorfas* si existe una biyección $\varphi : (P \cup T) \to (P' \cup T')$ tal que $F(x, y) \in \mathcal{N}' = F'(\varphi(x'), \varphi(y')) \in \mathcal{N}$ para todo $x, y \in P \cup T$.

Definición 1 *Un* sistema de reescritura de redes *es una estructura* $N = (\mathcal{R}, \Gamma_0, M_0)$

donde $\mathcal{R} = \{r_1, \ldots, r_h\}$ *es un conjunto finito de reglas de reescritura,* $\Gamma_0 = (P_0, T_0, F_0)$ *es una red de Petri y* $M_0 : P_0 \rightarrow \mathbb{N}$ *es un marcado asociado a* Γ_0*.*

Una regla de reescritura $r \in \mathcal{R}$ *es una estructura* $r = (L, R, \tau, {}^\bullet\tau, \tau^\bullet)$ *donde*

1. $L = (P_L, T_L, F_L)$ *y* $R = (P_R, T_R, F_R)$ *son redes de Petri llamadas* parte izquierda *y* parte derecha *de* r*, respectivamente.*

2. $\tau \subseteq (P_L \times P_R) \cup (T_L \times T_R)$*, llamada* relación de transferencia *de* r*, es una relación binaria que relaciona lugares de* L *con lugares de* R *y transiciones de* L *con transiciones de* R*:* $P_L\tau \subseteq P_R$*,* $\tau P_R \subseteq P_L$*,* $T_L\tau \subseteq T_R$ *y* $\tau T_R \subseteq T_L$*.*

3. ${}^\bullet\tau \subseteq \tau$ *y* $\tau^\bullet \subseteq \tau$ *son subrelaciones de las relaciones de transferencia llamadas* relación interfaz de entrada *y* relación interfaz de salida*, respectivamente.*

Una configuración *de un sistema de reescritura de redes* N *es una red de Petri* $\Gamma = (P, T, F)$*. Un* estado *de un sistema de reescritura de redes* N *es un par* (Γ, M) *formado por una red de Petri* $\Gamma = (P, T, F)$ *(configuración) y un marcado* $M : P \rightarrow \mathbb{N}$ *para* Γ*, es decir, es una red de Petri marcada. El par* (Γ_0, M_0) *es el estado inicial del sistema de reescritura de redes. Un* evento *de un sistema de reescritura de redes es o bien una transición o bien una regla de reescritura:* $E = T \cup R$*.*

Para aplicar una regla de reescritura r en una configuración Γ debemos primero identificar una subred total Γ' de Γ isomorfa a la parte izquierda de la regla. Los elementos de Γ (lugares o transiciones) que no pertenezcan a Γ' constituyen el *contexto* de la regla. Para que una regla esté habilitada además se requiere que un elemento x' de Γ' tenga un elemento x de su *preset* que pertenezca al contexto sólo si x' pertenece al *interfaz de entrada* de la regla y, simétricamente, que un elemento x' de Γ' tenga un elemento x de su *postset* que pertenezca al contexto sólo si x' pertenece al *interfaz de salida* de la regla. Cuando estas condiciones se dan, la reescritura puede tener lugar y se reemplaza la subred Γ' por la parte derecha R de la regla, fijándose las conexiones entre los elementos de R y los del contexto de acuerdo con las relaciones interfaz. La relación de transferencia no sólo se utiliza para reescribir la parte izquierda por la parte derecha de la regla sino también para transferir los tokens de Γ' a R (de ahí su nombre). Nótese que, ya que la relación de transferencia puede ser cualquier tipo de relación, los tokens pueden duplicarse o pueden desaparecer.

La evolución dinámica de un sistema de reescritura de redes viene dada por su grafo de estados.

Definición 2 *El* grafo de estados *de un sistema de reescritura de redes* $N = (\mathcal{R}, \Gamma_0, M_0)$ *es el grafo dirigido etiquetado cuyos nodos son los estados de* N*, es* *decir, redes de Petri marcadas y cuyos arcos (etiquetados con eventos de N) son de dos clases:*

- Disparo de una transición: *arcos desde el estado* (Γ, M) *al estado* (Γ', M') *etiquetados con la transición t cuando la transición t puede dispararse en la red con marcado M y lleva al marcado M':*

$$(\Gamma, M) \xrightarrow{t} (\Gamma', M') \Longleftrightarrow (\Gamma = \Gamma' \quad y \quad M[t\rangle M' \ en\ \Gamma)$$

- Cambio de configuración: *arcos desde el estado* (Γ, M) *al estado* (Γ', M') *etiquetados con la regla* $r = (L, R, \tau, {}^{\bullet}\tau, \tau^{\bullet}) \in \mathcal{R}$*, cuando existe un* embedding total *(ver [1])* $f : L \to \Gamma$ *tal que para todo* $x \notin f(L)$*,* $y \in L$ *se tiene*

$$x \in\ {}^{\bullet}f(y) \Rightarrow y \in Dom({}^{\bullet}\tau) \ \ y \ \ x \in\ f(y)^{\bullet} \Rightarrow y \in Dom(\tau^{\bullet})$$

y, siendo $\Gamma = (P, T, F)$ *y* $\Gamma' = (P', T', F')$*, se cumple lo siguiente:*

$$P' = P - f(P_L) + P_R \ \ tal\ que\ P_L\tau = P_R$$

$$T' = T - f(T_L) + T_R \ \ tal\ que\ T_L\tau = T_R$$

donde el significado de +(-) es añadir(eliminar) lugares/transiciones a(de) Γ*. Los nombres de los lugares* P_R*(transiciones* T_R*) añadidos a* Γ *deben ser nombres nuevos para evitar conflictos.*

$$M'(p) = \begin{cases} M(p) & si\ p \notin R \\ \sum_{p' \in\ \tau p} M(p') & si\ p \in R \end{cases}$$

$$F'(x, y) = \begin{cases} F(x, y) & si\ x \notin R\ \wedge\ y \notin R \\ F_R(x, y) & si\ x \in R\ \wedge\ y \in R \\ \sum_{y_i \in\ {}^{\bullet}\tau y} F(x, f(y_i)) & si\ x \notin R\ \wedge\ y \in R \\ \sum_{x_i \in\ \tau^{\bullet} x} F(f(x_i), y) & si\ x \in R\ \wedge\ y \notin R \end{cases}$$

En la sección que sigue presentamos una clase particular de sistemas de reescritura de redes, las redes reconfigurables, en las que la relación de transferencia es una biyección. Demostramos que este modelo es una forma abreviada del modelo básico de las redes de Petri, es decir, tiene el mismo poder expresivo pero la descripción del sistema a modelar es más breve y sencilla.

3. Redes reconfigurables. Implementación con redes de Petri

En esta sección se presenta una clase específica de sistemas de reescritura de redes que se corresponde con el modelo de redes reconfigurables que introdujimos en estudios previos [2, 3, 6]. Más exactamente, la definición que sigue es una reformulación de lo que allí se llamaba red reconfigurable *reversible*. Una *red reconfigurable* es un sistema de reescritura de redes tal que la relación de transferencia τ es una biyección y las interfaces $\tau^\bullet = {}^\bullet\tau$ son restricciones de la relación de transferencia τ al conjunto de transiciones, es decir, $\tau^\bullet = {}^\bullet\tau = T_L \times T_R \cap \tau$.

La primera condición afirma que el conjunto de lugares y transiciones no sufre cambios por la aplicación de reglas de reescritura, la segunda condición asegura que un cambio de configuración implica la modificación de las relaciones de flujo de los lugares en el dominio de la regla implicada de acuerdo a esa regla e independientemente del contexto en el que se aplique esta reescritura. Así pues, la definición anterior puede reformularse de manera directa (esto es, sin recurrir a sistemas de reescritura de redes) utilizando la siguiente definición que consideramos como la definición oficial de redes reconfigurables (ya que la reversibilidad se asumirá siempre implícitamente).

Definición 3 *Una* red reconfigurable *es una estructura* $N = (P, T, R, \gamma_0)$ *siendo* $P = \{p_1, \ldots, p_n\}$ *un conjunto finito y no vacío de lugares,* $T = \{t_1, \ldots, t_m\}$ *un conjunto finito y no vacío de transiciones disjunto de* P *(*$P \cap T = \emptyset$*),* $R = \{r_1, \ldots, r_h\}$ *un conjunto finito de reglas de reescritura, y* γ_0 *el estado inicial.*

Una regla de reescritura $r \in R$ *es una estructura* $r = (D, {}^\bullet r, r^\bullet)$ *donde* $D \subseteq P$ *es el dominio de* r*,* ${}^\bullet r : D \to (T \to \mathbb{N})^2$ *y* $r^\bullet : D \to (T \to \mathbb{N})^2$ *son la precondición y postcondición de* r*.*

Una configuración *de una red reconfigurable es una red de Petri* $\Gamma = (P, T, F)$*. Un* estado γ *de una red reconfigurable es un par* $\gamma = (\Gamma, M)$ *donde* Γ *es una configuración y* $M : P \to \mathbb{N}$*. Por tanto, un estado es una red de Petri marcada asociada con el conjunto de lugares y transiciones de la red reconfigurable. Los* eventos *de una red reconfigurable son sus transiciones junto con sus reglas de reescritura:* $E = T \cup R$*.*

Representamos una regla de reescritura utilizando la notación de sumas formales:

$$r = \sum_{p \in D} p((\sum_{t \in T} {}^\bullet r(p,t) \cdot t, \sum_{t \in T} {}^\bullet r(t,p) \cdot t)) \quad \rhd \sum_{p \in D} p((\sum_{t \in T} r^\bullet(p,t) \cdot t, \sum_{t \in T} r^\bullet(t,p) \cdot t))$$

Definición 4 *El* grafo de configuración $G(N)$ *de una red reconfigurable* $N = (P, T, R, \gamma_0)$ *es el grafo dirigido etiquetado cuyos nodos son las configuraciones, y tal que exis-*

te un arco desde la configuración Γ *a la configuración* Γ' *etiquetado con la regla* $r = (D, {}^\bullet r, r^\bullet) \in R$*, que denotamos* $\Gamma[r\rangle\Gamma'$*, si y solamente si se cumple lo siguiente:*

$$\forall p \in D, \quad \begin{cases} F(p,t) = {}^\bullet r(p,t) \text{ y } F(t,p) = {}^\bullet r(t,p) \\ F'(p,t) = r^\bullet(p,t) \text{ y } F'(t,p) = r^\bullet(t,p) \end{cases}$$

$$\forall p \notin D, \quad F(p,t) = F'(p,t) \text{ y } F(t,p) = F'(t,p)$$

La evolución dinámica de una red reconfigurable viene dada por su grafo de estados.

Definición 5 *El* grafo de estados *de una red reconfigurable* $N = (P, T, R, \gamma_0)$ *es el grafo dirigido etiquetado cuyos nodos son los estados de* N *y cuyos arcos (etiquetados con eventos) son de dos clases:*

- Disparo de una transición: *arcos desde el estado* (Γ, M) *al estado* (Γ, M') *etiquetados con la transición* t *cuando la transición* t *puede dispararse en la red* Γ *con marcado* M *y que conduce al marcado* M'.
- Cambio de configuración: *arcos desde el estado* (Γ, M) *al estado* (Γ', M) *etiquetados con la regla* $r \in R$ *si* $\Gamma[r\rangle\Gamma'$ *es una transición de* $G(N)$.

Es decir, el conjunto de transiciones (arcos etiquetados) del grafo de estados de N *es:* $\{(\Gamma, M) \xrightarrow{t} (\Gamma, M') \mid M[t\rangle M' \text{ en } \Gamma\} \cup \{(\Gamma, M) \xrightarrow{r} (\Gamma', M) \mid \Gamma[r\rangle\Gamma' \text{ en } G(N)\}$.

A partir de la definición anterior, es obvio que la evolución de un sistema modelado mediante una red reconfigurable depende de qué evento ocurra. Si se dispara una transición, sólo se ve afectado el marcado de la red (como en las redes de Petri). En cambio, si se aplica una regla de reescritura, lo que se modifica es la estructura de la red manteniéndose el marcado original, es decir, los cambios de configuración son ortogonales al comportamiento de la red subyacente. Por otro lado, en el caso general de los sistemas de reescritura de redes, la aplicación de una regla de reescritura no sólo afecta a la estructura de la red sino también al marcado de la red y, por tanto, el comportamiento es distinto al de la red que subyace. No obstante, en la medida de que en este caso sólo se dispone de herramientas de modelización y simulación y que no es difícil restringir el comportamiento para condicionar los cambios de configuración a ciertas propiedades locales de los sistemas, será siempre posible tener en cuenta dichas restricciones, al menos a nivel de simulador.

Para definir las reglas de reescritura de una forma genérica, agrupamos lugares y transiciones de acuerdo a una decisión del diseñador. Esto no cambia la naturaleza del modelo considerado pero facilita su descripción. Así, decimos que dos lugares (transiciones) tienen el mismo *rol* si el diseñador las agrupa en la misma clase de lugar (transición).

Ejemplo 6 (Productores/Consumidores) *La Figura 1 representa dos líneas de comunicación half-duplex entre dos productores/consumidores.*

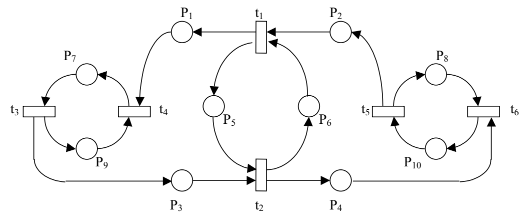


Figura 1: Dos líneas de comunicación *half-duplex* entre dos productores/consumidores

En esta red reconfigurable pueden tener lugar algunos cambios de configuración. Cada uno de ellos se representa mediante una regla de reescritura. En el ejemplo, distinguimos los roles P, Q y S entre lugares y los roles A y B entre transiciones: $P = \{P_1, P_2, P_3, P_4\}$, $Q = \{P_5, P_6\}$, $S = \{P_7, P_8, P_9, P_{10}\}$, $A = \{t_1, t_2\}$ y $B = \{t_3, t_4, t_5, t_6\}$. Definimos cuatro reglas de reescritura usando la notación de sumas formales. Mostramos en la Figura 2 la última regla gráficamente y en la Figura 3 parte del grafo de estados de la red reconfigurable.

1. *Cambio de modo secuencial a modo paralelo:*
   $R_1 : Q(A - A) + Q(-A + A) \rhd Q(\emptyset) + Q(\emptyset)$

2. *Cambio de sentido en una línea de comunicación:*
   $R_2 : P(-A) + P(A) \rhd P(A) + P(-A)$

3. *Cambio de productor/consumidor a consumidor:*
   $R_3 : S(B - B) + P(-B) + S(-B + B) + P(B)$
   $\rhd S(B - B) + P(B) + S(-B + B) + P(\emptyset)$

4. *Cambio de productor/consumidor a productor:*
   $R_4 : S(B - B) + P(-B) + S(-B + B) + P(B)$
   $\rhd S(B - B) + P(-B) + S(-B + B) + P(\emptyset)$

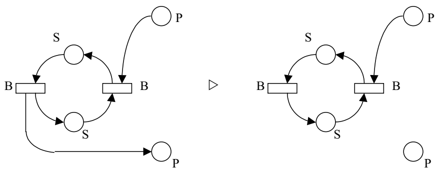


Figura 2: Regla de reescritura que cambia de productor/consumidor a productor

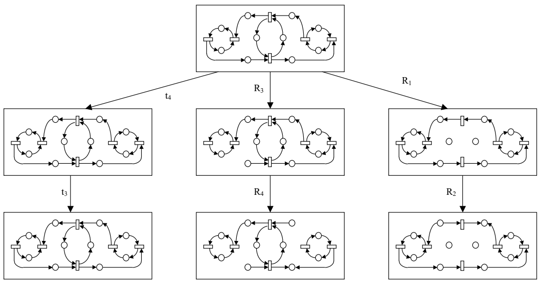


Figura 3: (Parte del) grafo de estados de la red reconfigurable

Llamamos *configuración local* a una función $T \to \mathbb{N}\times\mathbb{N}$ que representa las relaciones de flujo entrantes y salientes con respecto a un conjunto fijo de transiciones T de algún lugar de una configuración de la red reconfigurable. Una configuración de una red reconfigurable consiste en asociar cada lugar (del conjunto fijo de lugares) con la correspondiente configuración local. Así pues, una configuración local de una red reconfigurable podrá ser bien una configuración local de la configuración inicial o bien una configuración local de la parte derecha de una regla. Por tanto, sólo existen un número finito de configuraciones locales y (puesto que el conjunto de lugares es, a su vez, fijo), sólo un número finito de configuraciones. Por $Conf(N) = \{\Gamma_0, \Gamma_1, \ldots, \Gamma_k\}$ denotamos el conjunto de configuraciones de una red reconfigurable N, donde Γ_0 es la *configuración inicial* o configuración del estado inicial $\gamma_0 = (\Gamma_0, M_0)$. Por $\Gamma_i[r\rangle\Gamma_j$ denotamos el cambio de configuración debido a la regla de reescritura r de Γ_i a Γ_j. Podemos construir fácilmente una red de Petri equivalente $\widetilde{N} = (\widetilde{P}, \widetilde{T}, \widetilde{F}, \widetilde{M}_0)$ cuyo conjunto de lugares es $\widetilde{P} = P \cup \{q_0, \ldots, q_k\}$, es decir, los lugares de la red reconfigurable original junto con un lugar específico añadido a cada posible configuración. Tendremos tantas copias del conjunto de transiciones como número de configuraciones, $\{q_0, \ldots, q_k\} \times T$. Podemos imaginar los lugares y las transiciones de una configuración $\Gamma_i = (P, T, F_i)$ como si estuvieran situados en dos planos paralelos diferentes, es decir, un plano con el conjunto de lugares y un plano con el conjunto de transiciones conectados mediante las relaciones de flujo de la configuración representada. Las relaciones de flujo se establecen de forma que la configuración Γ_i estará representada por el plano de transiciones $\{q_i\} \times T$ y el plano (compartido) de lugares P, así: $\widetilde{F}(p, (q_i, t)) = F_i(p, t)$ y $\widetilde{F}((q_i, t), p) = F_i(t, p)$. Además, el lugar q_i asociado con la configuración Γ_i contiene como mucho un token, y está marcado en los estados asociados con su configuración, siendo

$$\widetilde{F}(q_i, (q_j, t)) = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{en otro caso} \end{cases} \qquad \widetilde{F}((q_j, t), q_i) = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{en otro caso} \end{cases}$$

Sólo nos queda representar el cambio de configuración. Para este propósito es suficiente añadir una transición extra $r_{i,j}$ por cada cambio de configuración $\Gamma_i[r\rangle\Gamma_j$ cuya relación de flujo viene dada por $\widetilde{F}(q_i, r_{i,j}) = \widetilde{F}(r_{i,j}, q_j) = 1$ y $\widetilde{F}(p, r_{i,j}) = \widetilde{F}(r_{i,j}, p) = 0$ en otro caso. Así pues, el conjunto de transiciones es $\widetilde{T} = (\{q_0, \ldots, q_k\} \times T) \cup \widetilde{R}$ donde $\widetilde{R}$ es el conjunto de transiciones tal que $r_{i,j} \in \widetilde{R}$ si $\exists r \in R$ tal que $\Gamma_i[r\rangle\Gamma_j$ en $G(N)$. La red de Petri resultante inicialmente está marcada como: $\widetilde{M}_0(p) = M_0(p)$ siendo $p \in P$ y $\widetilde{M}_0(q_i) = \begin{cases} 1 & \text{si } i = 0 \\ 0 & \text{en otro caso} \end{cases}$

Teorema 7 *Toda red reconfigurable $N = (P, T, \mathcal{R}, \gamma_0)$ es equivalente a una red de Petri $\widetilde{N} = (\widetilde{P}, \widetilde{T}, \widetilde{F}, \widetilde{M}_0)$. Además, el grafo de marcado de $\widetilde{N}$ es isomorfo (ver Sección 2) al de N.*

Así pues, las redes reconfigurables son equivalentes a las redes de Petri pero proporcionan representaciones más compactas de sistemas concurrentes cuya estructura evoluciona en tiempo de ejecución. Esto se puede apreciar en la Figura 4 que ilustra un fragmento de la red de Petri equivalente a la red reconfigurable del Ejemplo 6. Para facilitar la comprensión de la implementación de una red reconfigurable con una red de Petri, sólo mostramos cómo representar un cambio de configuración (el cambio de modo secuencial a paralelo) y solamente para los lugares y las transiciones implicados. Con este pequeño fragmento podemos hacernos una idea del tamaño de la red de Petri completa y de la manera fácil y concisa en la que podemos modelizar el sistema con una red reconfigurable. En general, si la red reconfigurable $N = (P, T, \mathcal{R}, \gamma_0)$ de la que partimos tiene n lugares, m transiciones, r reglas de reescritura y el número de configuraciones es $k+1$, la red de Petri equivalente $\widetilde{N} = (\widetilde{P}, \widetilde{T}, \widetilde{F}, \widetilde{M}_0)$ que obtendremos consta de: $n + (k+1)$ lugares y $((k+1) * m) + z$ transiciones, siendo $z = \sum_{i=0}^{k}$número de disparos de reglas de $\mathcal{R}$ en la configuración Γ_i, es decir, z es el número de transiciones de $\widetilde{R}$. Así pues, las redes reconfigurables pueden verse como un modelo abreviado de las redes de Petri; es decir, la expresividad es la misma pero con las redes reconfigurables podemos modelizar más fácil y directamente sistemas que cambian su estructura dinámicamente.

4. Simulación de máquinas de Turing con sistemas de reescritura de redes

En esta sección mostramos que sólo con que las relaciones de transferencia sean biyecciones parciales, ya obtenemos una subclase de sistemas de reescritura de redes

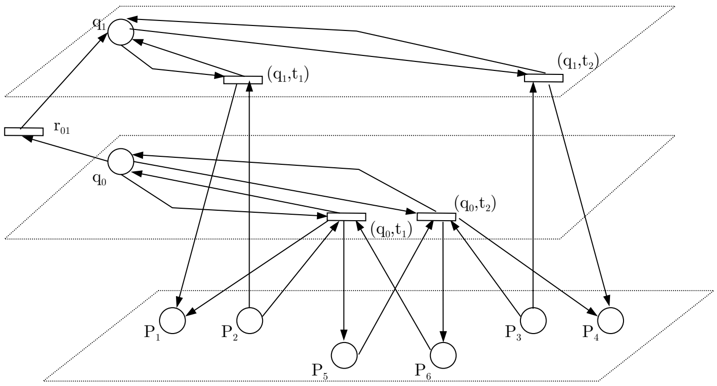
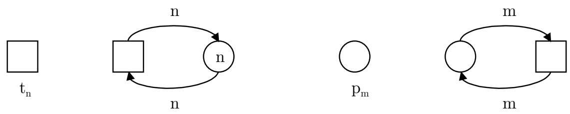


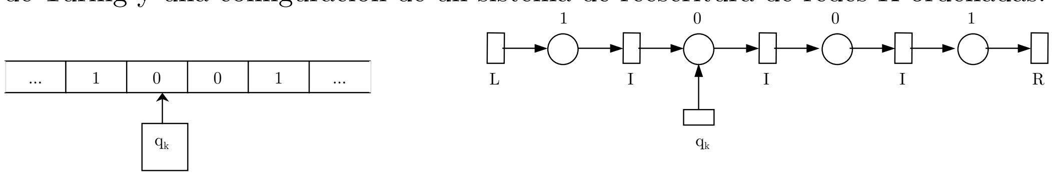
Figura 4: Parte de red de Petri equivalente a la red reconfigurable del Ejemplo 6

con la misma expresividad que las máquinas de Turing. Se dice que una relación $\rho \subseteq X \times Y$ es una *biyección parcial* si induce una biyección cuando se restringe a su dominio y codominio, es decir, cada elemento de X tiene como mucho una imagen en Y y cada elemento de Y tiene como mucho una co-imagen en X.

El hecho de que las máquinas de Turing puedan simularse mediante sistemas de reescritura de redes cuyas relaciones de transferencia son biyecciones parciales es directo puesto que el modelo de los sistemas de reescritura de redes no cambia al añadir restricciones de orden a sus lugares y transiciones. Formalmente, si K es un conjunto finito de órdenes, una *red de Petri K-ordenada* es una red de Petri (P, T, F) junto con una función $\kappa : P \cup T \to K$ que asocia todo elemento de la red con un orden en K. Un *sistema de reescritura de redes K-ordenadas* es, por tanto, un sistema de reescritura de redes en el que todos los componentes de las redes de Petri que lo constituyen (configuración inicial y partes izquierda y derecha de las reglas) están K-ordenados. El *grafo de estados de un sistema de reescritura de redes K-ordenadas* no es más que el grafo de estados del sistema de reescritura de redes subyacente con la excepción de que para poder aplicar un cambio de configuración se requiere, además, que el *embedding* (de la parte izquierda de la regla en la configuración actual) sea compatible con los órdenes, es decir, que haga corresponder un elemento (lugar o transición) con un elemento con el mismo orden. Podemos asociar todo sistema de reescritura de redes K-ordenadas cuyas relaciones de transferencia sean biyecciones parciales con un sistema de reescritura de redes equivalente cuyas relaciones de transferencia sean también biyecciones parciales. Para este propósito, notamos que, ya que las relaciones de transferencia son biyecciones parciales, las relaciones de flujo de sus distintas configuraciones estarán acotadas superiormente por el máximo entre los valores de las relaciones de flujo de la configuración inicial y los valores de las relaciones de flujo de la parte derecha de las reglas. Representamos que una transición t tiene un orden n y que un lugar p tiene un orden m mediante subredes concretas añadidas a las redes originales. Estos enteros se elegirán lo suficientemente grandes para que no se confundan con los valores de las relaciones de flujo de alguna configuración alcanzable. Es decir, vamos a exigir que todas las relaciones de flujo del sistema tengan un peso inferior al peso de las relaciones de flujo de los órdenes.



En las figuras mostramos la equivalencia entre una configuración de una máquina de Turing y una configuración de un sistema de reescritura de redes K-ordenadas.



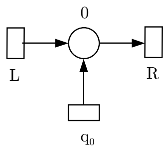
Una celda de la máquina es un lugar con el orden 0 ó 1 representando su contenido. Cada estado de la máquina de Turing también se considera un orden y la celda actual se representa por el lugar que tiene una transición de entrada con el estado actual de la máquina como orden. Representamos que una celda es la más a la izquierda, interna o la más a la derecha en función del orden de las transiciones de entrada y de salida del lugar que interviene. Las relaciones de transición entre las configuraciones de la máquina son las reglas de reescritura del sistema de reescritura.

Formalmente, una máquina de Turing determinista se representa por un sistema de reescritura de redes K-ordenadas cuya relación de transferencia τ es una biyección parcial como sigue. Sea $N = (\mathcal{R}, \Gamma_0, M_0)$ un sistema de reescritura de redes K-ordenadas en el que τ es una biyección parcial y donde

- $K = \{0, 1\} \cup Q \cup \{L, R, I\}$ es el conjunto de órdenes tal que $\{0, 1\}$ son los órdenes de los lugares e indican el contenido de la celda que cada lugar representa, y las transiciones pueden estar ordenadas con $Q \cup \{L, R, I\}$ indicando:
  - $L(R)$ es el orden de la transición de entrada (salida) al lugar que representa la celda más a la izquierda (derecha) de la cinta,
  - I es el orden de las transiciones entre los lugares que representan celdas internas, y

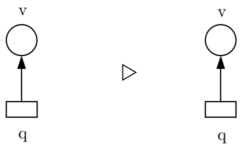
  - Q es un conjunto finito de órdenes que representan los estados internos de la máquina. El estado de la configuración actual q_k está representado por una transición de entrada al lugar de la celda actual con el orden $q_k \in Q$;

- $\Gamma_0 = (P_0, T_0, F_0)$ es la configuración inicial donde $P_0 = \{p_0\}$, $T_0 = \{q_0, L, R\}$ y F_0 se define como sigue: $F_0(p_0, q_0) = 0$, $F_0(q_0, p_0) = 1$, $F_0(p_0, L) = 0$, $F_0(L, p_0) = 1$, $F_0(p_0, R) = 1$ y $F_0(R, p_0) = 0$. Gráficamente,



- M_0 es el marcado inicial tal que $M_0(p_0) = 0$,

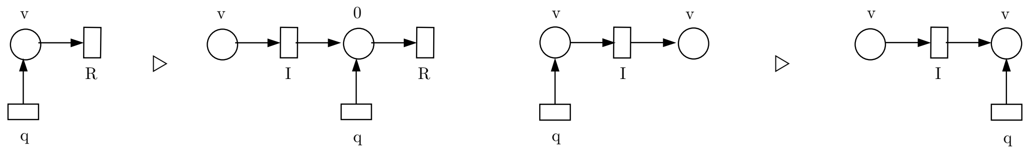
- $\mathcal{R}$ es el conjunto de reglas de reescritura y representa las relaciones de transición entre configuraciones de la máquina de Turing. Las reglas son las siguientes:
  - **Escritura de un valor en la celda actual**. Tenemos una regla de reescritura por cada instrucción de la máquina de Turing de la forma $[q, v, v', q'] \in Q \times \{0,1\}^2 \times Q$.



  - **Movimiento de la cabeza a derecha**. Tenemos dos reglas de reescritura por cada instrucción de la máquina de Turing de la forma $[q, v, R, q']$ según si la celda actual es la más a la derecha (Fig. 5(a)) o no (Fig. 5(b)).
  - **Movimiento de la cabeza a izquierda**. Tenemos dos reglas de reescritura por cada instrucción de la máquina de Turing de la forma $[q, v, L, q']$ según si la celda actual es la más a la izquierda (Fig. 6(a)) o no (Fig. 6(b)).

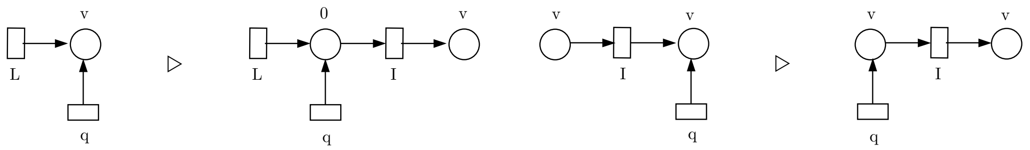
Ya hemos visto cómo simular una máquina de Turing determinista a partir de un sistema de reescritura de redes K-ordenadas cuya relación de transferencia τ es una biyección parcial. Esta simulación puede extenderse a la clase general de sistemas de reescritura de redes y, por tanto, no podremos obtener herramientas de verificación automáticas para la clase general de sistemas de reescritura de redes.

Proposición 8 *Los sistemas de reescritura de redes tienen el mismo poder computacional que las máquinas de Turing.*



(a) La celda activa es la más a la derecha (b) La celda activa no es la más a la derecha

Figura 5: Movimiento de la cabeza a derecha



(a) La celda activa es la más a la izquierda (b) La celda activa no es la más a la izquierda

Figura 6: Movimiento de la cabeza a izquierda

5. Conclusiones

Hemos distinguido dos clases particulares de sistemas de reescritura de redes de acuerdo con su relación de transferencia τ. El modelo de las redes reconfigurables es un sistema de reescritura de redes en el que la relación de transferencia τ es una biyección. Este modelo, aunque formalmente es equivalente a las redes de Petri, nos permite expresar de una forma más fácil y concisa sistemas en los que pueden ocurrir cambios estructurales dinámicamente. No obstante, la traducción automática a redes de Petri asegura que todas las propiedades fundamentales de las redes de Petri siguen siendo decidibles para las redes reconfigurables. Para este modelo podemos encontrar, por tanto, herramientas de verificación automáticas. La traducción a redes de Petri equivalentes incrementa considerablemente el tamaño de la red y, por ello, resulta más eficiente implementar los métodos de verificación de propiedades de las redes de Petri directamente en el modelo original. Esto supone que las nociones de grafo de cobertura, invariantes lineales, sifones y *traps* pueden definirse directamente para una red reconfigurable, haciendo uso de las simetrías inducidas por el grupo generado por la relación de transferencia. Estos son los tópicos de nuestra investigación actual.

Por el contrario, la clase de los sistemas de reescritura de redes tiene el poder computacional de la máquina de Turing, con lo que no es posible la verificación automática en este caso. Sin embargo, este modelo sigue siendo interesante como herramienta de modelización y simulación. Uno de nuestros objetivos futuros es implementar un simulador para sistemas de reescritura de redes.

Referencias

[1] E. Badouel, M. Llorens, and J. Oliver. Modelling Concurrent Systems using Reconfigurable Nets. In *Actes du 6eme Seminaire Atelier en Algèbre, Logique, et ses Applications*, Yaoundé, Cameroon, 2002.

[2] E. Badouel and J. Oliver. Reconfigurable Nets, a Class of High Level Petri Nets Supporting Dynamic Changes Within Workflow Systems. In *Proc. of Workflow Management: Net-based concepts, models, techniques and tools* (WFM'98), CSR 98/07, pp. 129–145, Lisbon, Portugal, 1998.

[3] E. Badouel and J. Oliver. Dynamic Changes in Concurrent Systems: Modelling and Verification. Technical report, Inria Research Report PI-3708, France, 1999.

[4] P. Baldan. Modelling Concurrent Computations: From Contextual Petri Nets to Graph Grammars. PhD Thesis, University of Pisa TD-1/00, 2000.

[5] A. Corradini. Concurrent Computing: From Petri Nets to Graph Grammars. Invited talk at the *Joint COMPUGRAPH/SEMAGRAPH Workshop on Graph Rewriting and Computation*, Elsevier, *ENTCS*, vol. 2, 1995.

[6] M. Llorens, and J. Oliver. Modelización de Sistemas Concurrentes mediante Redes Reconfigurables. In *IX Jornadas de Concurrencia*, pp. 213–224, Sitges, Barcelona, 2001.

[7] T. Murata. Petri Nets: Properties, Analysis and Applications. In *Proc. of the IEEE*, vol. 77, no.4, pp. 541–580, 1989.

[8] J.L. Peterson. Petri Net Theory and the Modeling of Systems. Englewood Cliffs, NJ. Prentice-Hall, 1981.

[9] H. Schneider. Graph Grammars as a Tool to Define the Behavior of Processes Systems: From Petri Nets to Linda. In the *4th Int. Conf. on Graph Grammars*, Williamsburg, USA, 1993.

[10] R. Valk. Self-modifying Nets, a Natural Extension of Petri Nets. In *Proc. of Int. Coll. on Automata, Languages and Programming* (ICALP'78), Springer-Verlag, *LNCS*, vol. 62, pp. 464–476, Udine, Italy, 1978.

[11] R. Valk. Generalizations of Petri Nets. In *Proc. of 10th Symp. on Mathematical Foundations of Computer Science* (MFCS'81), Springer-Verlag, *LNCS*, vol. 118, pp. 140–155, Strbske Pleso, Czechoslovakia, 1981.