Teoría de la Complejidad Computacional

Tema 7: Modelos de computación celular con membranas

David Orellana Martín

Grupo de Investigación en Computación Natural
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

dorellana@us.es

Máster Universitario en Lógica, Computación e Inteligencia Artificial
Curso 2023-2024

Índice

* **Computación natural**.
* **Modelos de computación celular con membranas**.
* **Sistemas P básicos de transición**.
* **Sistemas P con membranas activas**.
* **Sistemas P de tejidos con reglas symport/antiport**.
* **Complejidad computacional en sistemas de membranas**.

Modelos de computación convencionales versus no convencionales

Modelo de computación:

- ⋆ Formaliza el concepto de **procedimiento mecánico**.
- ⋆ Dispositivos del modelo : **máquinas**.

Máquina **convencional**: soporte electrónico.

Máquina **no convencional**: otro soporte distinto.

Limitaciones de las máquinas electrónicas

Máquinas: dispositivos finitos.

Chips electrónicos.

Recursos: **espacio** (memoria) y en **tiempo.**

- ⋆ Espacio: miniaturización (**R. Feymann, 1959**).
- ⋆ Tiempo: velocidad de cálculo de procesadores (**R. Churchhouse, 1983**).

Consecuencia:

- ⋆ Existen **problemas muy relevantes** de la vida real que **nunca** podrán ser resueltos por ordenadores electrónicos (a menos que . . .)

Computación Natural

Paradigma computacional inspirado en la Naturaleza viva.

Disciplina científica que trata de capturar la forma en que la Naturaleza lleva realizando *procesos* (procedimientos de cálculo) desde hace millones de años.

En este contexto, la Naturaleza viva puede ser considerada:

- **Como fuente de inspiración computacional**.
- **Como medio físico para realizar cálculos**.

Computación Natural

* **Redes neuronales artificiales**: W.S. McCulloch y W.H. Pitts (1943).
* **Teoría de la comunicación en el sistema nervioso humano**: J. von Neumann, H. Aiken, N. Wiener (1946).
* **Sistemas complejos auto-reproductivos**: J. von Neumann (1947).
* **Autómatas celulares**: J. von Neumann, S. Ulam (1951), inspirados en unas ideas de E.L. Post y A. Turing (1936).
* **Teoría de patrones**: A. Turing (1952).
* **Algoritmos genéticos**: J. Holland (1975).
* **Modelo Splicing**: T. Head (1987).
* **Computación molecular basada en ADN**: L. Adleman (1994).
* **Computación celular con membranas**: Gh. Păun (1998).

La célula (I)

Procesos esenciales para la **Vida**:

* Replicación del ADN.
* Producción de energía.
* Síntesis de proteínas.
* Procesos metabólicos.

Célula: entidad básica de la Vida.

* Estructura compleja y, a la vez, muy organizada.
* Permite ejecución simultánea de reacciones químicas.

La célula (II)

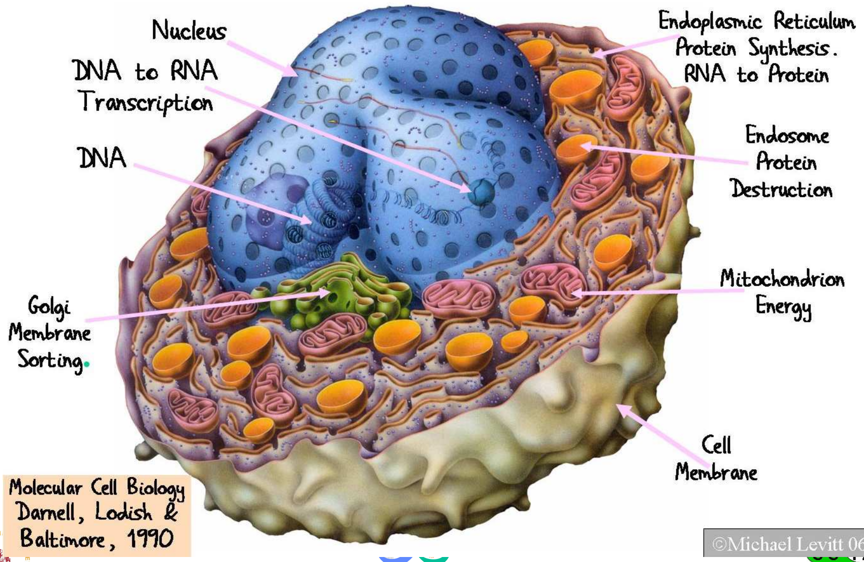
Partes de una célula:

- ⋆ Una especie de piel (**membrana plásmica**)
- ⋆ El corazón de la célula (**núcleo**), que almacena el ADN
- ⋆ El resto de la célula (**citoplasma**), que contiene:
  - La **mitocondria**: se encarga de producir energía.
  - El **aparato de Golgi**: fábrica de proteínas.
  - El **retículo endoplásmico**: red de membranas interconectadas.
  - Los **lisosomas**: estómagos de las células.

Existen dos tipos de células:

- ⋆ **Procariotas**: carecen de un núcleo bien definido (propias de los organismos unicelulares).
- ⋆ **Eucariotas**: poseen un núcleo rodeado por una doble membrana (específicas de animales y plantas).

La célula (III)



RGNC

Las membranas biológicas

⋆ Involucradas en la mayoría de reacciones químicas

⋆ Canales selectivos de comunicación (barreras **semipermeables**).

⋆ Controlan un flujo de datos; es decir, de **información**:

⋆ Premio Nobel de Química 2003: **P. Agre y R. MacKinnon** (canales proteínicos de las membranas).

Células versus máquinas

En una **célula viva**:

- ⋆ Cada **membrana** trabaja con **compuestos** químicos de acuerdo con unas **reacciones** específicas

En una **máquina paralela**:

- ⋆ Cada **procesador** trabaja con **datos** de acuerdo con un **programa** específico

Célula	**Máquina**
Membranas	Procesadores
Compuestos químicos	Datos
Reacciones químicas	Instrucciones

Paradigma de la computación celular con membranas

Membrane Computing: **Gh. Păun**, 1998–2000.

* Modelos de computación (**sistemas de membranas** o **sistemas P**):
  * Orientados a máquinas.
  * No deterministas.
  * Distribuidos.
  * Paralelos y maximales.
* Los modelos de computación celular pueden trabajar:
  * A modo de células (**cell-like**).
  * A modo de tejidos (**tissue-like**).
  * A modo de neuronas (**neural-like**).

Sistemas P básicos de transición (a modo de células)

Ingredientes **sintácticos**:

- ⋆ **Alfabeto de trabajo**(los elementos se denominan **objetos**).
- ⋆ Un subconjunto del alfabeto de trabajo (objetos de **entrada**).
- ⋆ Una **estructura de unidades de procesos** (membranas): **árbol enraizado**.
- ⋆ Un **multiconjunto** asociado a cada unidad de proceso.
- ⋆ Un conjunto finito de **reglas de evolución**.
- ⋆ Un **entorno** **pasivo**: **sólo recibe** **objetos**.
- ⋆ Una membrana distinguida (de **entrada**) y una zona distinguida (de **salida**), que puede ser una membrana o el entorno.

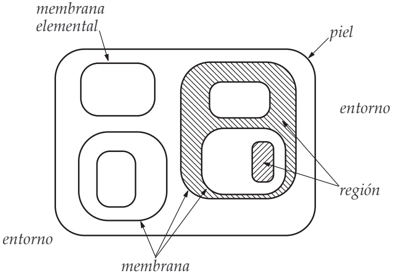
Sistemas P básicos de transición

Ingredientes **semánticos**:

- ⋆ **Configuración** o descripción instantánea del sistema.
  - ∗ **Configuraciones iniciales**.
- ⋆ **Aplicabilidad de las reglas** a una configuración.
- ⋆ **Transición** de una configuración a otra.
- ⋆ **Computación** a partir de una configuración inicial.

Sistemas P básicos de transición

Una **estructura de membranas** (diagrama de Venn):



Formalmente, una estructura de membranas es un **árbol enraizado**:

- ⋆ La raíz del árbol es la membrana piel.
- ⋆ Las hojas son las membranas elementales.
- ⋆ En este contexto, el "padre" de la membrana piel es el "entorno" del sistema.

Sistemas P básicos de transición: Sintaxis

Sistema P básico de transición de grado $q \geq 1$:

$$\Pi = (\Gamma, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, (\mathcal{R}_1, \rho_1), \ldots, (\mathcal{R}_q, \rho_q), i_{in}, i_{out})$$

en donde:

* Γ es un alfabeto (**objetos**) y $\Sigma \subsetneq \Gamma$.
* μ es una **estructura de membranas** de grado q: membranas etiquetadas biyectivamente con elementos de $\{1, \ldots, q\}$ (0 es la etiqueta del entorno).
* Para cada i, $1 \leq i \leq q$:
  * $\mathcal{M}_i$ es un multiconjunto finito sobre $\Gamma \setminus \Sigma$.
  * $\mathcal{R}_i$ es un conjunto de **reglas de evolución** asociadas a la membrana i, del tipo $u \to v$, con $v = (v_1, here)\,(v_2, out)\,(v_3, in_j)$ o bien $v = (v_1, here)\,(v_2, out)\,(v_3, in_j)\,\delta$, siendo u, v_1, v_2, v_3 multiconjuntos sobre Γ y δ es un símbolo distinguido.
  * ρ_i es un orden parcial estricto sobre $\mathcal{R}_i$ (**prioridades**: una regla tiene mayor prioridad que otra ...).
  * $i_{in} \in \{1, \ldots, q\}$ representa la **membrana de entrada** del sistema.
  * $i_{out} \in \{0, 1, \ldots, q\}$ representa la **zona de salida** del sistema.

Sistemas P básicos de transición: Semántica

Configuración inicial de Π: $\boxed{(\mu, \mathcal{M}_1, \dots, \mathcal{M}_q)}$

Configuración inicial de Π asociada a un multiconjunto m sobre Σ:

$$\boxed{(\mu, \mathcal{M}_1, \cdots, \mathcal{M}_{i_{in}} + m, \dots \mathcal{M}_q)}$$

Configuración de Π en un instante t: una tupla $C_t = (\mu', \mathcal{M}'_{i_1}, \dots, \mathcal{M}'_{i_k})$ tal que

- $\{i_1, \dots, i_k\}$ debe contener la etiqueta asociada a la membrana piel.
- μ': subárbol de μ obtenido al eliminar las membranas distintas de $i_1, \dots, i_k$.
- $\mathcal{M}'_{i_1}, \dots, \mathcal{M}'_{i_k}$ son multiconjuntos sobre $\Gamma \setminus \Sigma$.

Sistemas P básicos de transición: Semántica

* **Aplicabilidad** de una regla $u \to v$ de $\mathcal{R}_i$ a una configuración C_t.

  Condiciones necesarias:

  * En la estructura de membranas de C_t ha de aparecer una membrana etiquetada por i.
  * El multiconjunto u ha de estar contenido en esa membrana i de C_t.
  * Si (v_3, in_j) aparece en v, entonces j debe ser una hija de esa membrana i de C_t.
  * Si δ aparece en v, entonces i no puede ser la membrana piel ni, en su caso, la membrana de salida.
  * No existe una regla de $\mathcal{R}_i$ aplicable a C_t y con *mayor* prioridad que $u \to v$.

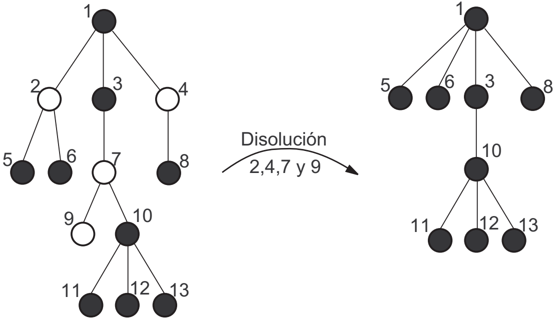
* Multiconjunto de reglas aplicables a una configuración.
* Las reglas se ejecutan en **paralelo**, de forma **maximal** y **no determinista**.

Sistemas P básicos de transición: Semántica

Sean $C' = (\mu', m'_{i_1}, \ldots, m'_{i_k}), C'' = (\mu'', m''_{j_1}, \ldots, m''_{j_l})$ configuraciones de Π:

- C'' se obtiene de C' en **un paso de transición** ejecutando las reglas de $\mathcal{R}_{i_1}, \ldots, \mathcal{R}_{i_k}$ aplicables a C' (de forma **paralela y maximal**) como sigue:
  - si $u \to v \in R_{i_s}$ y el multiconjunto u aparece en una membrana de μ' etiquetada por i_s, entonces
    - El multiconjunto u se elimina de la membrana i_s.
    - Si (v_1, here) aparece en v, se añade v_1 a la membrana i_s.
    - Si (v_1, out) aparece en v, se añade v_1 a la membrana *padre* de i_s (al entorno si i_s es la piel).
    - Si (v_1, in_j) aparece en v, se añade v_1 a la membrana j (hija de la membrana i_s).
    - Si $\delta \in v$, la membrana i_s se disuelve y su contenido pasa al primer antecesor no disuelto (la piel **no** se puede disolver ni, en su caso, la membrana de salida).
  - **Maximalidad**: tras la ejecución de las reglas no puede quedar un objeto por evolucionar al que se le pueda aplicar alguna regla.

Ilustración de la ejecución de las reglas de disolucion



Computaciones en sistemas P básicos de transición

Una **computación** $\mathcal{C}$ es una sucesión (finita o infinita) de configuraciones $(C_0, C_1, \ldots, C_r)$, con $r \in \mathbf{N} \cup \{+\infty\}$, tal que:

* C_0 es una configuración inicial de Π.
* Para cada $i < r$, C_{i+1} se obtiene de C_i por un paso de transición.

$\mathcal{C} = (C_0, C_1, \ldots, C_r)$ es una **computación de parada** si $r \in \mathbf{N}$. En ese caso, a C_r (**configuración de parada**) no se le puede aplicar ninguna regla.

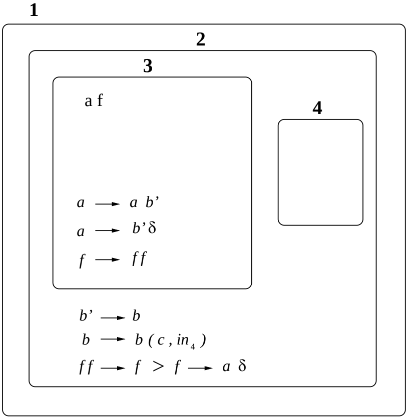
– El **resultado** de una computación de *parada* está codificado por el multiconjunto asociado a la membrana de salida de la configuración de parada.

Un sistema P básico de transición se puede considerar como:

* Una máquina *generadora* ($\Sigma = \emptyset$).
* Una máquina de *cálculo* ($\Sigma \neq \emptyset$).
* Una máquina de *decisión* o *reconocedora* ($\texttt{yes}, \texttt{no} \in \Gamma$ y $\Sigma \neq \emptyset$).

Ejemplo de un sistema P generador

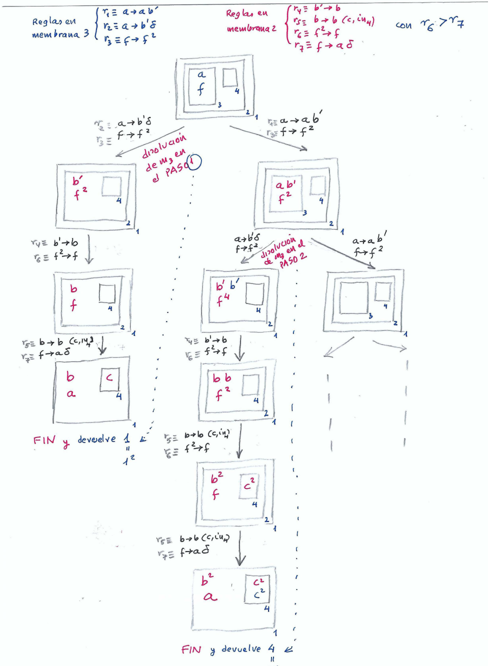
Un sistema celular con membranas que genera el conjunto $\{n^2 : \ n \geq 1\}$.



Membrana 4: membrana de salida.

- ⋆ Analizar las computaciones en función del paso de transición $m+1$ en el que se aplica la regla $a \rightarrow b'\,\delta$.

Traza de las computaciones



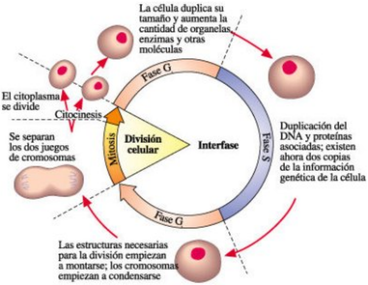
Contenidos de las membranas a lo largo de la evolución

Paso	Membrana 1	Membrana 2	Membrana 3	Membrana 4
0			af	
1			$ab' f^2$	
2			$ab'^2 f^{2^2}$	
3			$ab'^3 f^{2^3}$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
m			$ab'^m f^{2^m}$	
$m+1$		$b'^{(m+1)} f^{2^{m+1}}$	*disuelta*	
$m+2$		$b^{m+1} f^{2^m}$	*disuelta*	
$(m+2)+1$		$b^{m+1} f^{2^{m-1}}$	*disuelta*	c^{m+1}
$(m+2)+2$		$b^{m+1} f^{2^{m-2}}$	*disuelta*	$c^{2(m+1)}$
$(m+2)+3$		$b^{m+1} f^{2^{m-3}}$	*disuelta*	$c^{3(m+1)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$(m+2)+m$		$b^{m+1} f^{2^{m-m}}$	*disuelta*	$c^{m(m+1)}$
$2m+3$	ab^{m+1}	*disuelta*	*disuelta*	$c^{(m+1)(m+1)}$

El ciclo celular

El **ciclo celular** consta, básicamente, de cuatro fases.

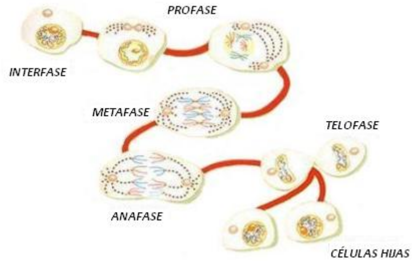
- Primera: etapa de crecimiento de la célula.
- Segunda: replicación del ADN.
- Tercera: nueva fase de crecimiento.
- Cuarta: mitosis celular.



La mitosis celular

Proceso que se produce en todo organismo vivo y que les permite crecer y reproducirse.

- ⋆ El resultado es la producción de dos células hijas a partir de una célula original.
- ⋆ Las células hijas son copias de la célula madre original.



Una abstracción de la mitosis celular ha sido incorporada a la Computación Celular con Membranas a través de una regla de división: **sistemas P con membranas activas**.

Sistemas P con membranas activas (I)

Sistema P con membranas activas $\Pi = (\Gamma, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$ de grado q:

* Γ es un alfabeto (de trabajo) y Σ es un alfabeto (de entrada) tal que $\Sigma \subsetneq \Gamma$;
* μ es una estructura de membranas de grado q (membranas etiquetadas biyectivamente por $\{1, \ldots, q\}$: la etiqueta de la raíz es i_s);
* $\mathcal{M}_1, \ldots, \mathcal{M}_q$ son multiconjuntos sobre $\Gamma \setminus \Sigma$ asociados a las membranas de μ;
* $i_{in} \in \{1, \ldots, q\}$ representa la membrana de entrada e $i_{out} \in \{0, 1, \ldots, q\}$ representa la **zona** de salida.
* $\mathcal{R}$ es un conjunto finito de reglas del siguiente tipo:

  (a) $[\, a \to u\,]_i^{\alpha}$, — evolución de objetos

  (b) $a\, [\ ]_i^{\alpha_1} \to [\, b\,]_i^{\alpha_2}$ $(i \neq i_s)$. — comunicación-in

  (c) $[\, a\,]_i^{\alpha_1} \to b\, [\ ]_i^{\alpha_2}$. — comunicación-out

  (d) $[\, a\,]_i^{\alpha} \to b$ $(i \neq i_s$, e $i \neq i_{out})$. — disolución

  (e) $[\, a\,]_i^{\alpha_1} \to [\, b\,]_i^{\alpha_2}\, [\, c\,]_i^{\alpha_3}$ $(i \neq i_s$, $i \neq i_{out}$ e i membrana elemental). — división (memb. elem)

  en donde $1 \leq i \leq q$, $\alpha, \alpha_i \in \{+, -, 0\}$, $a, b, c \in \Gamma$, u multiconjunto sobre Γ.

Sistemas P con membranas activas (II)

Las reglas son aplicadas de acuerdo con los siguientes principios:

- Las reglas están **asociadas a etiquetas** y pueden ser **usadas** por todas las membranas que tengan esa etiqueta.
- En cada paso de computación, a una membrana **sólo se le puede aplicar <u>una</u> regla** de los tipos (b), (c), (d) y (e). Y, en tal caso, sólo se aplica una vez.
- En cada paso de computación, a una membrana se le puede aplicar, simultáneamente, una regla del tipo (a) junto con una de los tipos (b), (c), (d) y (e). En tal caso, esa regla del tipo (a) deberá ser aplicada de manera <u>maximal</u>.
- La semántica de los sistemas P con membranas activas se define de manera similar a los sistemas P básicos de transición (configuración, paso de transición y computación).

Sistemas P con membranas activas (III)

Los sistemas P con membranas activas verifican:

- Usan tres cargas eléctricas.
- **No** usan cooperación ni prioridades.
- Las reglas están asociadas a etiquetas.
- La aplicación de una regla **puede modificar** la **carga** de una membrana.
- La aplicación de una regla **no** **modifica** la **etiqueta** de una membrana.

Sistemas P que trabajan a modo de tejidos

Unidades de proceso: **células**.

Ingredientes sintácticos:

- ⋆ Un **alfabeto**, cuyos elementos se denominan **objetos** (y dos subalfabetos: de entrada y del entorno).
- ⋆ Una **estructura** de **células**: un **grafo dirigido** (descrito implícitamente).
- ⋆ Un **entorno** **activo**: **recibe** objetos del sistema y **envía** objetos al sistema.
- ⋆ Unos **multiconjuntos de objetos** asociados a cada célula **y al entorno**.
- ⋆ Un conjunto de **reglas de evolución** asociado al sistema (que pueden ser de diversos tipos).
- ⋆ Dos "zonas" distinguidas: una **célula de entrada** y una **zona de salida** (célula o entorno).

Sistemas P que trabajan a modo de tejidos

Un **sistema P de tejidos con reglas de comunicación** de grado $q \geq 1$ es una tupla $\Pi = (\Gamma, \Sigma, \mathcal{E}, \mathcal{M}_1, \dots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$, en donde:

* Γ y Σ son alfabetos (de trabajo y de entrada, resp.) tales que $\Sigma \subsetneq \Gamma$.
* $\mathcal{E} \subsetneq \Gamma$ (alfabeto del entorno: etiquetado por 0) y tal que $\mathcal{E} \cap \Sigma = \emptyset$.
* $\mathcal{M}_1, \dots, \mathcal{M}_q$ son multiconjuntos sobre $\Gamma \setminus \Sigma$ asociados a cada una de las q células del sistema, etiquetadas biyectivamente por $\{1, \dots, q\}$.
* $\mathcal{R}$ es un conjunto finito de reglas del tipo **symport/antiport**: $(i, u/v, j)$, con $i, j \in \{0, 1, \dots, q\}, i \neq j$, u, v multiconjuntos sobre Γ, $|u| + |v| > 0$;
* $i_{in} \in \{1, \dots, q\}$ representa la célula de entrada.
* $i_{out} \in \{0, 1, \dots, q\}$ representa la **zona** de salida (una célula o el entorno).

Reglas symport/antiport

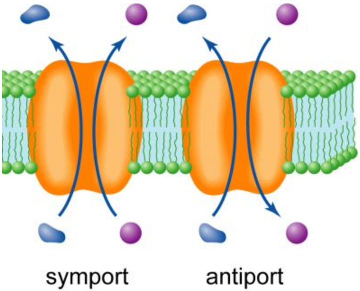
Una regla $(i, u/v, j)$ se dice que es del tipo **symport** si $u = \lambda$ o $v = \lambda$.

- Toda regla *symport* $(i, u/\lambda, j)$ proporciona un arco virtual desde i hasta j.

Una regla $(i, u/v, j)$ se dice que es del tipo **antiport** si $u \neq \lambda$ y $v \neq \lambda$.

- Toda regla *antiport* $(i, u/v, j)$ proporciona dos arcos: uno que va de i a j y otro que va de j a i.

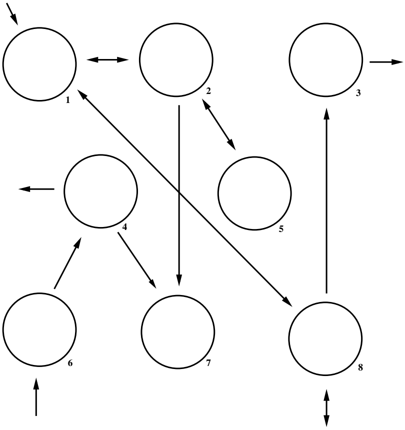
Longitud de la regla de comunicación $(i, u/v, j)$: $|u| + |v|$.



Grafo subyacente a un sistema P de tejidos

Todo sistema P de tejidos tiene asociado un grafo dirigido subyacente:

- **Nodos**: las células del sistema y el entorno.
- **Arcos**: se obtienen a partir reglas de la comunicación symport/antiport.



Communication rules: $\begin{cases} (0, ba^2/\lambda, 1), (0, \lambda/b^4cd, 3), (0, \lambda/ab^3, 4), (0, c/\lambda, 6), (0, a/b^2, 8), (1, c^3/b^2, 2) \\ (1, ad/a, 8), (2, ab/\lambda, 7), (2, b/b^2, 5), (3, \lambda, d^2, 8), (4, \lambda/a, 6), (4, b^2c^2/\lambda, 7) \end{cases}$

Semántica de los sistemas P de tejidos

Una **configuración** en un instante determinado es una tupla formada por:

- ⋆ Los multiconjuntos de objectos sobre Γ asociados a cada célula.
- ⋆ El multiconjunto de objectos sobre $\Gamma - \mathcal{E}$ asociados con el entorno.

Configuración inicial de $\Pi = (\Gamma, \Sigma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$:

$$\boxed{(\mathcal{M}_1, \cdots, \mathcal{M}_q; \emptyset)}$$

En esta configuración, los objetos del entorno aparecen con infinitas copias.
Configuración inicial de Π **asociada a un multiconjunto** m sobre Σ:

$$\boxed{(\mathcal{M}_1, \cdots, \mathcal{M}_{i_{in}} + m, \ldots \mathcal{M}_q; \emptyset)}$$

Una regla $(i, u/v, j)$ es **aplicable** a una configuración del sistema si u está contenido en la zona i y v está contenido en la zona j.

Cuando se aplica $(i, u/v, j)$, los objetos de u son enviados de i a j y, al mismo tiempo, los objetos de v son enviados de j a i.

Configuración de **parada**: ninguna regla del sistema es aplicable a dicha configuración.

Semántica de los sistemas P de tejidos

Sean C y C' dos configuraciones:

- ⋆ C' se obtiene de C mediante un **paso de transición** ($C \Rightarrow_{\Pi} C'$) si se puede pasar de C a C' aplicando las reglas del sistema (de forma no determinista, paralela y maximal).

Una **computación** es una sucesión (finita o infinita) de configuraciones tal que:

- ⋆ El primer término es una configuración inicial del sistema y cada uno de los restantes se obtiene del anterior mediante un paso de transición.
- ⋆ Si la sucesión es finita (*computación de parada*), el último término es una *configuración de parada*.
- ⋆ Resultado de una computación de parada: codificado por los objetos presentes en la zona de salida.

Sistemas P de tejidos con división celular

Un **sistema P de tejidos con división celular** de grado $q \geq 1$ es una tupla $\Pi = (\Gamma, \Sigma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$, en donde:

* Γ y Σ son alfabetos (de trabajo y de entrada, resp.) tales que $\Sigma \subsetneq \Gamma$;
* $\mathcal{E} \subsetneq \Gamma$ (alfabeto del entorno: etiquetado por 0) y tal que $\mathcal{E} \cap \Sigma = \emptyset$.
* $\mathcal{M}_1, \ldots, \mathcal{M}_q$ son cadenas sobre $\Gamma \setminus \Sigma$ asociadas a cada una de las q células del sistema, etiquetadas biyectivamente por $\{1, \ldots, q\}$.
* $\mathcal{R}$ es un conjunto finito de reglas del tipo:
  - (a) **Reglas symport/antiport**: $(i, u/v, j)$, con $i, j \in \{0, 1, \ldots, q\}$, $i \neq j$, $u, v \in \Gamma^*$, $|u + v| \neq 0$.
  - (b) **Reglas de División**: $[a]_i \rightarrow [b]_i[c]_i$, donde $i \in \{1, \ldots, q\}$, $i \neq i_{out}$ y $a, b, c \in \Gamma$.
* $i_{in} \in \{1, \ldots, q\}$ representa la célula de entrada.
* $i_{out} \in \{0, 1, \ldots, q\}$ representa la <u>zona de salida</u> (una célula o el entorno).

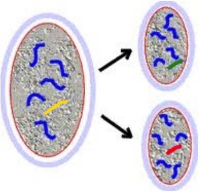
Reglas de división celular

Una regla $[\,a\,]_i \rightarrow [\,b\,]_i[\,c\,]_i$ es aplicable a una célula i de una configuración C si:

- ⋆ Dicha célula no es la de salida y, además, contiene al objeto a.

Cuando se aplica la regla $[\,a\,]_i \rightarrow [\,b\,]_i[\,c\,]_i$:

- ⋆ La célula i se divide en dos células con la misma etiqueta.
- ⋆ En la primera copia, el objeto a es reemplazado por el objeto b.
- ⋆ En la segunda copia, el objeto a es reemplazado por el objeto c.
- ⋆ Todos los objetos restantes se replican y se copian en las dos células nuevas.



En un sistema P de tejidos con división celular, las reglas se aplican de forma no determinista, paralela y maximal, pero con una restricción importante:

- ⋆ Si en una célula se aplica una regla de división, sólo dicha regla se aplicará en ese paso (bloqueo de todas las comunicaciones con otras células y con el entorno).

Complejidad computacional en sistemas de membranas

Resolver un problema de decisión X equivale a **reconocer** o **decidir** el lenguaje L_X asociado a dicho problema.

Se introducen los **sistemas de membranas reconocedores** a fin de:

- ⋆ Proporcionar **soluciones eficientes** de problemas de decisión en el marco de la computación celular con membranas (en particular, de problemas computacionalmente duros: **NP**-completos).

Ahora bien:

- ⋆ Los sistemas de membranas son **dispositivos no deterministas**.
- ⋆ Las MTND resuelven problemas **NP**-completos en **tiempo polinomial**.
- ⋆ Las MTND **no son fiables** ya que:
  - No capturan el **verdadero concepto de algoritmo**.
  - El resultado de <u>una</u> ejecución de una MTND puede conducir a <u>error</u>.

Sistemas de membranas reconocedores

Establecer "distancias" entre las MTNDs y los sistemas de membranas.

- ⋆ Las soluciones **eficientes** que proporcionen los sistemas de membranas han de ofrecer ventajas **cualitativas** respecto a las obtenidas por MTNDs.

- ⋆ Esas soluciones deben capturar el **verdadero concepto de algoritmo**; es decir, para <u>cada dato de entrada</u> del problema:
    - El resultado de **cualquier** computación asociada a <u>esa entrada</u>, debe codificar la respuesta correcta.

Sistemas de membranas reconocedores

Un **sistema de membranas reconocedor**[1] es un sistema de membranas tal que:

- ⋆ El alfabeto de trabajo contiene dos elementos distinguidos: **yes**, **no**.
- ⋆ Existe una membrana/celula de entrada (i_{in}).
- ⋆ La zona de salida del sistema es el entorno (i_{out}: etiqueta del entorno).
- ⋆ **Todas las computaciones** son de **parada**.
- ⋆ Si $\mathcal{C}$ es una computación del sistema, entonces o bien un objeto `yes` o un objeto `no` (pero no ambos) se envían al entorno y, además, **sólo en el último paso de la computación**.

En un sistema de membranas reconocedor, diremos que

- ⋆ Una computación es de *aceptación* (resp. de *rechazo*) si el objeto `yes` (resp., `no`) aparece en el entorno asociado a la configuración de parada.

[1] M.J. Pérez-Jiménez, A. Romero, F. Sancho. Complexity classes in models of cellular computing with membranes. *Natural Computing*, 2, 3 (2003), 265-285.

Codificación polinomial

Sea $X = (\Sigma_X, I_X, \theta_X)$ un problema de decisión y $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbf{N}\}$ una familia de sistemas de membranas reconocedores.

Codificación polinomial de X en $\mathbf{\Pi}$:

- Es un par (cod, s) de funciones sobre I_X computables en tiempo polinomial tales que para cada $u \in I_X$:
  - $s(u)$ es un número natural.
  - $cod(u)$ es un multiconjunto de entrada del sistema $\Pi(s(u))$.
  - $s^{-1}(n)$ es un conjunto finito, para cada $n \in \mathbf{N}$.

Adecuación y completitud

Sea (cod, s) una **codificación polinomial** de X en $\mathbf{\Pi}$.

⋆ $\mathbf{\Pi}$ es **adecuada** respecto de (X, cod, s) si para cada $u \in I_X$ se tiene:

- Si **existe una computación** de **aceptación** de $\Pi(s(u)) + cod(u)$, entonces $\theta_X(u) = 1$.

⋆ $\mathbf{\Pi}$ es **completa** respecto de (X, cod, s) si para cada $u \in I_X$ se tiene:

- Si $\theta_X(u) = 1$ entonces **toda computación** de $\Pi(s(u)) + cod(u)$ es de **aceptación**.

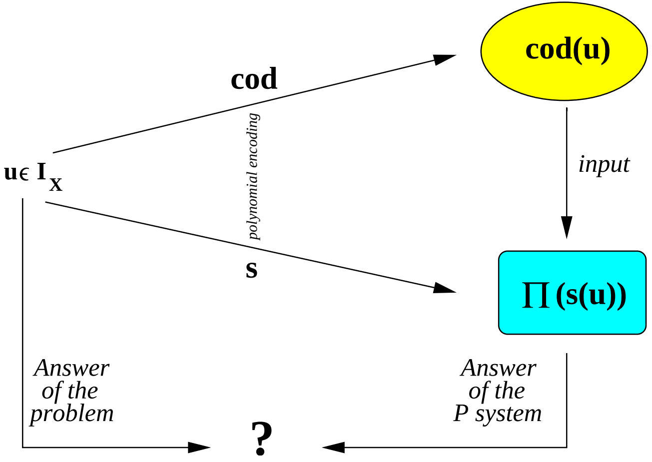
Resolución de problemas en tiempo polinomial (I)

Sea $X = (\Sigma_X, I_X, \theta_X)$ un problema de decisión y $\mathbf{\Pi} = \{\Pi(n) : \ n \in \mathbf{N}\}$ una familia de sistemas de membranas reconocedores.

X es **resoluble en tiempo polinomial** por la familia $\mathbf{\Pi}$ ($X \in \mathbf{PMC}_{\mathcal{R}}$) sii:

- ⋆ $\mathbf{\Pi}$ es **polinomialmente uniforme** por MT: **Existe <u>una</u> MTD** que trabaja en tiempo polinomial y construye $\Pi(n)$ a partir de $n \in \mathbf{N}$.
- ⋆ Existe una **codificación polinomial** (cod, s) de X en $\mathbf{\Pi}$ tal que:
  - $\mathbf{\Pi}$ es **polinomialmente acotada**: Existe un polinomio $p(n)$ tal que para cada $u \in I_X$, <u>todas</u> las computaciones de $\Pi(s(u)) + cod(u)$ paran en, a lo sumo, $p(|u|)$ pasos.
  - $\mathbf{\Pi}$ es **adecuada** y **completa** respecto a (X, cod, s).

Resolución de problemas en tiempo polinomial (II)



Estabilidad de las clases de complejidad polinomial

Sea $\mathcal{R}$ una clase de sistemas de membranas reconocedores.

La clase de complejidad $\mathbf{PMC}_{\mathcal{R}}$:

- ⋆ Es **estable bajo reducibilidad** en tiempo polinomial.
  - Si $X_1 \leq^p X_2$ y $X_2 \in \mathbf{PMC}_{\mathcal{R}}$, entonces $X_1 \in \mathbf{PMC}_{\mathcal{R}}$.
- ⋆ Es **cerrada bajo complementario**.
  - Si $X \in \mathbf{PMC}_{\mathcal{R}}$, entonces $\overline{X} \in \mathbf{PMC}_{\mathcal{R}}$.

De las propiedades anteriores se deduce lo siguiente:

- ⋆ Si un problema **NP**-completo pertenece a la clase $\mathbf{PMC}_{\mathcal{R}}$, entonces se verifica que $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{R}}$.

Una solución lineal del problema SAT mediante sistemas P reconocedores con membranas activas

Se diseña una familia de sistemas P reconocedores con membranas activas $\{\Pi(t) : \ t \in \mathbf{N}\}$ que resuelve SAT en tiempo polinomial:

- ⋆ $\Pi(t)$ procesará todas las fórmulas proposicionales en FNC con n variables y m cláusulas, siendo $t = \langle m, n \rangle$.

  Recuérdese que la aplicación de $\mathbf{N} \times \mathbf{N}$ en $\mathbf{N}$ definida por $\langle m, n \rangle = \frac{(m+n)(m+n+1)}{2} + m$ es biyectiva y computable en tiempo polinomial.

Para cada $(m, n) \in \mathbf{N} \times \mathbf{N}$ se define $\Pi(\langle m, n \rangle) = (\Gamma, \Sigma, \mu, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, i_{in})$:

- ⋆ $\Sigma = \{x_{i,j}, \overline{x}_{i,j} : 1 \leq i \leq m, 1 \leq j \leq n\}$.
- ⋆ $\Gamma = \Sigma \cup \{c_k : 1 \leq k \leq m+2\} \cup \{d_k : 1 \leq k \leq 3n+2m+3\} \cup$
  $\cup \{r_{i,k} : 0 \leq i \leq m, 1 \leq k \leq 2n\} \cup \{e, t\} \cup \{\mathtt{yes}, \mathtt{no}\}$.
- ⋆ $\mu = [\ [\ \ ]_2\ ]_1$ (membrana interna: etiquetada por 2).
- ⋆ $\mathcal{M}_1 = \emptyset$ y $\mathcal{M}_2 = \{d_1\}$.
- ⋆ $i_{in} = 2$ es la *membrana de entrada*.

El conjunto de reglas $\mathcal{R}$ es el siguiente:

(a) $\{[\, d_k\,]_2^0 \rightarrow [\, d_k\,]_2^+ \; [\, d_k\,]_2^- \; : 1 \leq k \leq n\}$.

(b) $\{[\, x_{i,1} \rightarrow r_{i,1}\,]_2^+, [\overline{x}_{i,1} \rightarrow r_{i,1}\,]_2^-, [\, x_{i,1} \rightarrow \lambda\,]_2^-, [\,\overline{x}_{i,1} \rightarrow \lambda\,]_2^+ : 1 \leq i \leq m\}$.

(c) $\{[\, x_{i,j} \rightarrow x_{i,j-1}\,]_2^+, [\, x_{i,j} \rightarrow x_{i,j-1}\,]_2^- \; : 1 \leq i \leq m, 2 \leq j \leq n\}$.
$\{[\,\overline{x}_{i,j} \rightarrow \overline{x}_{i,j-1}\,]_2^+, [\,\overline{x}_{i,j} \rightarrow \overline{x}_{i,j-1}\,]_2^- \; : 1 \leq i \leq m, 2 \leq j \leq n\}$.

(d) $\{[\, d_k\,]_2^+ \rightarrow [\;\;]_2^0\, d_k, [\, d_k\,]_2^- \rightarrow [\;\;]_2^0\, d_k \; : 1 \leq k \leq n\}$.
$\{d_k\, [\;\;]_2^0 \rightarrow [\, d_{k+1}\,]_2^0 : 1 \leq k \leq n-1\}$.

(e) $\{[\, r_{i,k} \rightarrow r_{i,k+1}\,]_2^0 : 1 \leq i \leq m, 1 \leq k \leq 2n-1\}$.

(f) $\{[\, d_k \rightarrow d_{k+1}\,]_1^0 : n \leq k \leq 3n-3\}; [\, d_{3n-2} \rightarrow d_{3n-1} e\,]_1^0$.

(g) $e\, [\;\;]_2^0 \rightarrow [\, c_1\,]_2^+; [\, d_{3n-1} \rightarrow d_{3n}\,]_1^0$.

(h) $\{[\, d_k \rightarrow d_{k+1}\,]_1^0 : 3n \leq k \leq 3n+2m+2\}$.

(i) $[\, r_{1,2n}\,]_2^+ \rightarrow [\;\;]_2^- \; r_{1,2n}$.

(j) $\{[\, r_{i,2n} \rightarrow r_{i-1,2n}\,]_2^- \; : 1 \leq i \leq m\}$.

(k) $r_{1,2n}\, [\;\;]_2^- \rightarrow [\, r_{0,2n}\,]_2^+$.

(l) $\{[\, c_k \rightarrow c_{k+1}\,]_2^- \; : 1 \leq k \leq m\}$.

(m) $[\, c_{m+1}\,]_2^+ \rightarrow [_2\;]_2^+ \; c_{m+1}$.

(n) $[\, c_{m+1} \rightarrow c_{m+2}\; t\,]_1^0$.

(o) $[\, t\,]_1^0 \rightarrow [\;\;]_1^+ \; t$.

(p) $[\, c_{m+2}\,]_1^+ \rightarrow [\;\;]_1^- \; \texttt{yes}$.

(q) $[\, d_{3n+2m+3}\,]_1^0 \rightarrow [\;\;]_1^+ \; \texttt{no}$.

Obsérvese que para cada $m, n \geq 1$, el sistema de membranas $\Pi(\langle m, n \rangle)$:

- ⋆ Tiene una estructura de membranas muy sencilla: sólo contiene dos membranas: la membrana piel y una membrana hija (que llamaremos **membrana interna**).
- ⋆ **No usa reglas de disolución** de membranas.
- ⋆ Las reglas de evolución de objetos $[\, a \rightarrow u \,]_i$ son **simples** (**mínima producción**; es decir, la parte derecha u consta de un sólo símbolo (con multiplicidad 1).

Veamos que $\mathbf{\Pi} = \{\Pi(t) : \ t \in \mathbf{N}\}$ proporciona una solución lineal de SAT.

Consideramos la siguiente codificación polinomial (cod, s) de SAT en $\mathbf{\Pi}$:

⋆ Para cada fórmula $\varphi = C_1 \wedge \cdots \wedge C_m$ en FNC con $Var(\varphi) = \{x_1, \ldots, x_n\}$:

- $s(\varphi) = \langle m, n \rangle = \frac{(m+n)\cdot(m+n+1)}{2} + m$.
- $cod(\varphi) = \{x_{i,j} : x_j \in C_i\} \cup \{\overline{x}_{i,j} : \neg x_j \in C_i\}$.

La función s de $\mathbf{N} \times \mathbf{N}$ en $\mathbf{N}$ es biyectiva y computable en tiempo polinomial.

La función cod es computable en tiempo polinomial y $cod(\varphi)$ es un multiconjunto de entrada del sistema $\Pi(s(\varphi)) = \Pi(\langle m, n \rangle)$.

La fórmula φ (instancia de SAT) será procesada por el sistema de membranas $\Pi(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$.

Si $\mathcal{C}$ es una computación del sistema $\Pi(s(\varphi)) + cod(\varphi)$, notaremos por $\mathcal{C}^i$ la configuración obtenida tras ejecutar i pasos de $\mathcal{C}$.

$$\mathcal{C} \equiv \mathcal{C}^0 \Rightarrow \mathcal{C}^1 \Rightarrow \mathcal{C}^2 \Rightarrow \cdots \Rightarrow \mathcal{C}^r$$

La ejecución del sistema $\Pi(s(\varphi)) + cod(\varphi)$ se estructura en cuatro fases:

- Fase de **generación y primer chequeo**:
  - Se hallan todas las posibles valoraciones relevantes para φ (codificadas en las membranas internas). A la vez que se van generando los valores de una variable, se chequean qué cláusulas son verdaderas por esos valores.

- Fase de **sincronización**:
  - Tiene como objetivo fijar el instante en que finaliza la fase anterior, así la siguiente fase comenzará, simultáneamente, en todas las membranas.

- Fase de **segundo chequeo**:
  - Se determina el número de cláusulas verdaderas (distintas) que hay en cada membrana interna.

- Fase de **salida**:
  - Se produce la respuesta correcta del sistema, de acuerdo con lo obtenido en la fase anterior.

Fase de generación y primer chequeo

Se hallan todas las valoraciones relevantes para φ y, a la vez, se hallan las cláusulas que son verdaderas por esos valores.

Se estructura en un **bucle** que da, exactamente, n vueltas.

- La vuelta k-ésima $(1 \leq k \leq n-1)$ consume 3 pasos de computación.
  - Primer paso: La presencia del objeto d_k en una membrana interna con carga neutra dispara la correspondiente regla de división creando dos membranas (una con carga positiva, que significa asignar 1 a la variable x_k, y otra con carga negativa, que significa asignar 0 a la variable x_k).
  - Segundo paso: En cada membrana interna se hallan las clausulas verdaderas por el valor asignado a x_k (si C_i es verdadera entonces se produce un objeto $r_{i,1}$ en esa membrana). Simultáneamente, los objetos $x_{i,j}$ y $\overline{x}_{i,j}$, con $j \geq 2$, se transforman en $x_{i,j-1}$ y $\overline{x}_{i,j-1}$, respectivamente. Así, los objetos $x_{i,1}$ y $\overline{x}_{i,1}$ en esta vuelta representarán a x_k y $\overline{x}_k$, respectivamente. Además, los objetos d_k salen a la piel cambiando a neutra la carga de la membrana interna.
  - Tercer paso: El objeto d_k pasa de la piel a una membrana interna transformado en d_{k+1}, a fin de comenzar la siguiente vuelta. Simultáneamente, el segundo subíndice de $r_{i,l}$ se aumenta una unidad, por cuestiones de sincronización.

Fase de generación y primer chequeo

* La vuelta n-ésima consume 2 pasos de computación.
  * Primer paso: La presencia del objeto d_n en una membrana interna con carga neutra dispara la correspondiente regla de división creando dos membranas (una con carga positiva, que significa asignar 1 a la variable x_n, y otra con carga negativa, que significa asignar 0 a la variable x_n). Simultáneamente, los objetos $r_{i,l}$ aumentan el segundo subíndice en una unidad.
  * Segundo paso: En cada membrana interna se hallan las clausulas verdaderas por el valor asignado a x_n (si C_i es verdadera entonces se produce un objeto $r_{i,1}$ en esa membrana). Simultáneamente, los objetos d_n salen a la membrana piel.

Por tanto, la fase de generación y primer chequeo consume, exactamente, **$3n - 1$ pasos** de computación, finalizando cuando los objetos d_n aparecen en la membrana piel.

$$\mathcal{C}^0 \overset{Paso\ 1}{\Longrightarrow} \mathcal{C}^1 \overset{Paso\ 2}{\Longrightarrow} \mathcal{C}^2 \Longrightarrow \dots \overset{Paso\ 3n-1}{\Longrightarrow} \mathcal{C}^{3n-1}$$

Fase de sincronización

Como se ha indicado, esta fase comienza cuando cuando los objetos d_n aparecen en la membrana piel y consume $2n$ pasos de computación.

- En cada paso de esta etapa, d_i $(n \leq i \leq 3n-1)$ evoluciona a d_{i+1}. Simultáneamente, los objetos $r_{i,l}$ van aumentado su segundo subíndice, de uno en uno, hasta llegar a $2n$.

- Esta fase finaliza cuando los objetos d_{3n} aparecen en la piel (momento en el que cada membrana interna posee carga positiva, contiene c_1 y objetos del tipo $r_{i,2n}$).

Por tanto, la fase de sincronización consume, exactamente, **$2n$ pasos** de computación.

$$\mathcal{C}^{3n-1} \stackrel{Paso\,1}{\Longrightarrow} \mathcal{C}^{3n} \stackrel{Paso\,2}{\Longrightarrow} \mathcal{C}^{3n+1} \Longrightarrow \ldots \stackrel{Paso\,2n}{\Longrightarrow} \mathcal{C}^{5n-1}$$

Fase de segundo chequeo

Esta fase comienza cuando los objetos d_{3n} aparecen en la membrana piel.

El objetivo de esta fase consiste en determinar el número de cláusulas verdaderas (distintas) que hay en cada membrana interna.

- La presencia del objeto c_{i+1}, $i \geq 1$, en una membrana interna prueba que las cláusulas $C_1, \ldots, C_i$ son verdaderas por la valoración codificada.
- Para cada c_i, $1 \leq i \leq m$, el objeto c_{i+1} aparece en *alguna* membrana (si corresponde) tras ejecutar 2 pasos.
- Esta fase finaliza cuando el objeto $d_{3n+2m+1}$ aparece en la piel, junto con un objeto c_{m+1} (en el caso en que la fórmula de entrada sea satisfactible).

Por tanto, la fase de segundo chequeo consume, exactamente, $2m+1$ **pasos** de computación.

$$\mathcal{C}^{5n-1} \overset{Paso\ 1}{\Longrightarrow} \mathcal{C}^{5n} \overset{Paso\ 2}{\Longrightarrow} \mathcal{C}^{5n+1} \Longrightarrow \ldots \overset{Paso\ 2m+1}{\Longrightarrow} \mathcal{C}^{5n+2m}$$

Fase de salida

Se produce la respuesta correcta del sistema en, exactamente, **3 pasos** de computación.

* **Primer caso**: La formula φ **es satisfactible**.

  En este caso, la membrana piel de la configuración $\mathcal{C}^{5n+2m}$ contiene 2^n objetos $d_{3n+2m+1}$ y algún objeto c_{m+1}.

  - Primer paso: Mediante el esquema de reglas (h) el objeto $d_{3n+2m+1}$ produce $d_{3n+2m+2}$ y mediante la regla (n) los objetos c_{m+1} producen los objetos c_{m+2} y t.
  - Segundo paso: Mediante el esquema de reglas (h) el objeto $d_{3n+2m+2}$ produce $d_{3n+2m+3}$ y mediante la regla (o) un objeto t sale al entorno, provocando que la membrana piel pase a tener carga positiva.
  - Tercer paso: Puesto que la piel tiene carga positiva y contiene al objeto c_{m+2}, aplicando la regla (p), un objeto **yes** se envía al entorno, la carga de la piel pasa a negativa y la computación finaliza.

$$\mathcal{C}^{5n+2m} \overset{Paso\ 1}{\Longrightarrow} \mathcal{C}^{5n+2m+1} \overset{Paso\ 2}{\Longrightarrow} \mathcal{C}^{5n+2m+2} \overset{Paso\ 3}{\Longrightarrow} \mathcal{C}^{\mathbf{5n+2m+3}}$$

Fase de salida

* **Segundo caso**: La formula φ **no es satisfactible**.

  En este caso, la membrana piel de la configuración $\mathcal{C}^{5n+2m}$ contiene únicamente 2^n objetos $d_{3n+2m+1}$.

  - Primer paso: Mediante el esquema de reglas (h), los objetos $d_{3n+2m+1}$ producen objetos $d_{3n+2m+2}$.
  - Segundo paso: Mediante el esquema de reglas (h), los objetos $d_{3n+2m+2}$ producen objetos $d_{3n+2m+3}$.
  - Tercer paso: Aplicando la regla (q), un objeto $d_{3n+2m+3}$ envia un objeto **no** al entorno y, además, la carga de la membrana piel pasa a ser positiva, con lo cual la computación finaliza.

$$\mathcal{C}^{5n+2m} \overset{Paso\ 1}{\Longrightarrow} \mathcal{C}^{5n+2m+1} \overset{Paso\ 2}{\Longrightarrow} \mathcal{C}^{5n+2m+2} \overset{Paso\ 3}{\Longrightarrow} \mathcal{C}^{\mathbf{5n+2m+3}}$$

Π es polinomialmente uniforme por MT

Las reglas de los sistemas $\Pi(\langle m, n \rangle)$ están descritas recursivamente y la cantidad de recursos usados para su construcción es:

- Número total de **objetos**: $4mn + 3m + 5n + 9 \in \Theta(m \cdot n)$
- Número de **membranas iniciales**: 2.
- Máximo **cardinal** de los **multiconjuntos iniciales**: 1.
- Número total de **reglas**: $6mn + 3m + 6n + 10 \in \Theta(m \cdot n)$.
- Máxima **longitud** de una **regla**: 3.

Π es polinomialmente acotada respecto a $(\texttt{SAT}, cod, s)$

El tiempo de ejecución de $\Pi(s(\varphi)) + cod(\varphi)$ es:

- $\star$ $3n - 1$ pasos en la fase de **generación y primer chequeo**.
- $\star$ $2n$ pasos en la fase de **sincronización**.
- $\star$ $2m + 1$ pasos en la fase de **segundo chequeo**.
- $\star$ 3 pasos en la fase de **salida**.

Tiempo total de ejecución: $5n + 2m + 3 \in \Theta(n + m)$.

Π es adecuada y completa respecto a $(\mathtt{SAT}, cod, s)$

- Cada sistema de la familia $\mathbf{\Pi}$ es determinista.

- $\mathbf{\Pi}$ es adecuada y completa respecto a $(\mathtt{SAT}, cod, s)$,

  - Si alguna membrana interna de $\mathcal{C}^{5n+2m-1}$ contiene un objeto c_{m+1}, entonces, en el último paso, el sistema envía un objeto **yes** al entorno.

  - Si ninguna membrana interna de $\mathcal{C}^{5n+2m-1}$ contiene un objeto c_{m+1}, entonces, en el último paso, el sistema envía un objeto **no** al entorno.

  - Si la computación de $\Pi(s(\varphi)) + cod(\varphi)$ es de aceptación, entonces la fórmula φ es satisfactible (**Adecuación**).

  - Si la la fórmula φ es satisfactible, entonces la computación de $\Pi(s(\varphi)) + cod(\varphi)$ es de aceptación (**Completitud**).

Conclusión

La familia $\mathbf{\Pi}$ de sistemas de membranas de $\mathcal{AM}$ resuelve el problema $\mathtt{SAT}$ en tiempo lineal. Es decir:

Teorema: $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{AM}}$.

Corolario: $\mathbf{NP} \cup \mathbf{co-NP} \subseteq \mathbf{PMC}_{\mathcal{AM}}$.

Recordando que $\mathbf{PMC}_{\mathcal{NAM}} = \mathbf{P}$, se deduce que pasar de $\mathcal{NAM}$ a $\mathcal{AM}$ equivale a pasar de la **no eficiencia** (capacidad para resolver sólo problemas tratables) a la **presumible eficiencia** (capacidad para resolver problemas **NP**-completos).

En consecuencia, suponiendo que $\mathbf{P} \neq \mathbf{NP}$, en el marco de los sistemas P reconocedores con membranas activas, las reglas de **división de membranas** proporciona una **frontera** entre la no eficiencia y la presumible eficiencia.

Sistemas P con membranas activas y sin polarizaciones

Un sistema P con membranas activas y sin polarizaciones de grado $q \geq 1$ es una tupla $\Pi = (\Gamma, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$, en donde:

* Γ es un alfabeto (de trabajo) y Σ es un alfabeto (de entrada) tal que $\Sigma \subsetneq \Gamma$;
* μ es una estructura de membranas de grado q: membranas etiquetadas biyectivamente en $\{1, \ldots, q\}$ (supondremos que 1 es la etiqueta de la raíz);
* $\mathcal{M}_1, \ldots, \mathcal{M}_q$ son los multiconjuntos iniciales sobre $\Gamma \setminus \Sigma$ colocados en las q membranas (regiones delimitadas por μ) etiquetadas biyectivamente por $\{1, \ldots, q\}$;
* $i_{in} \in \{1, \ldots, q\}$ representa la membrana de entrada e $i_{out} \in \{0, 1, \ldots, q\}$ representa la **zona** de salida;
* $\mathcal{R}$ es un conjunto finito de reglas del siguiente tipo:

(a)	$[\, a \to u \,]_i$	**evolución**
(b)	$a\, [\ \ ]_i \to [\, b \,]_i$ $(i \neq 1)$	**comunicación-in**
(c)	$[\, a \,]_i \to b\, [\ \ ]_i$	**comunicación-out**
(d)	$[\, a \,]_i \to b$ $(i \neq 1$ e $i \neq i_{out})$	**disolución**
(e)	$[\, a \,]_i \to [\, b \,]_i\, [\, c \,]_i$ $(i \neq 1,\ i \neq i_{out}$ e i elemental$)$	**división elemental**
(f)	$[\, [\ ]_j\, [\ ]_k \,]_i \to [\, [\ ]_j \,]_i\, [\, [\ ]_k \,]_i$ $(i \neq 1,\ i \neq i_{out}$ e i no elemental$)$	**división no elemental**

en donde $1 \leq i \leq q$, $\alpha, \alpha_i \in \{+, -, 0\}$, $a, b, c \in \Gamma$, u multiconjunto sobre Γ.

Sistemas P con membranas activas y sin polarizaciones

Notaremos por $\mathcal{AM}^0$ el modelo de computación de los sistemas P reconocedores con membranas activas y sin polarizaciones.

- Las notaciones $\mathcal{AM}^0(\alpha, \beta)$, con $\alpha \in \{-d, +d\}$ y $\beta \in \{-ne, +ne\}$ significan lo siguiente:
  - $\alpha = -d$: el citado modelo <u>no usa</u> reglas de disolución.
  - $\alpha = +d$: el citado modelo <u>usa</u> reglas de disolución.
  - $\beta = -ne$: el citado modelo <u>usa</u> reglas de división <u>sólo</u> para membranas elementales.
  - $\beta = +ne$: el citado modelo <u>usa</u> reglas de división para membranas elementales y no elementales.

Sistemas P con membranas activas y sin polarizaciones

Recuérdese que $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{AM}(-\mathbf{d})}$.

Algunos resultados importantes:

* $\mathbf{PMC}_{\mathcal{AM}^{\mathbf{0}}(-\mathbf{d},+\mathbf{ne})} = \mathbf{P}$ [2] (técnica del **grafo de dependencia**).
* $\texttt{Subset}-\texttt{Sum} \in \mathbf{PMC}_{\mathcal{AM}^{\mathbf{0}}(+\mathbf{d},+\mathbf{ne})}$[1].
* ¿Se pueden resolver problemas **NP**-completos mediante familias de $\mathcal{AM}^{\mathbf{0}}(+\mathbf{d},-\mathbf{ne})$, en tiempo polinomial?
* Conjetura de Păun (2005): $\mathbf{PMC}_{\mathcal{AM}^{\mathbf{0}}(+\mathbf{d},-\mathbf{ne})} = \mathbf{P}$.

[2] M.A. Gutiérrez, M.J. Pérez-Jiménez, A. Riscos, F.J. Romero. On the power of dissolution in P systems with active membranes. **Lecture Notes in Computer Science**, **3850** (2006), 224-240.