Computación Bio–inspirada

Tema 9: Resolución eficiente de problemas NP-completos en Modelos Celulares

David Orellana Martín
Mario de J. Pérez Jiménez

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

dorellana@us.es (http://www.cs.us.es/∼dorellana/)
marper@us.es (http://www.cs.us.es/∼marper/)

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

Índice

- Una solución eficiente del problema SAT mediante sistemas P reconocedores con **membranas activas**.

- Una solución eficiente del problema 3-COL mediante sistemas P reconocedores de **tejidos con división celular**.

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

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

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

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

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

- $\star$ $\Sigma = \{x_{i,j}, \overline{x}_{i,j} : 1 \leq i \leq m, 1 \leq j \leq n\}$.
- $\star$ $\Gamma = \Sigma \cup \{c_k : 1 \leq k \leq m+2\} \cup \{d_k : 1 \leq k \leq 3n+2m+3\} \cup$
  $\cup \{r_{i,k} : 0 \leq i \leq m, 1 \leq k \leq 2n\} \cup \{e, t\} \cup \{\texttt{yes}, \texttt{no}\}$.
- $\star$ $\mu = [\ [\ \ ]_2\ ]_1$ (membrana interna: etiquetada por 2).
- $\star$ $\mathcal{M}_1 = \emptyset$ y $\mathcal{M}_2 = \{d_1\}$.
- $\star$ $i_{in} = 2$ 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 $\mathbf{\Pi} = \{\Pi(t) : \ t \in \mathbb{N}\}$ 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 $\mathbb{N} \times \mathbb{N}$ en $\mathbb{N}$ 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.

$$\mathcal{C}^{3n-1} \overset{\textit{Paso } 1}{\Longrightarrow} \mathcal{C}^{3n} \overset{\textit{Paso } 2}{\Longrightarrow} \mathcal{C}^{3n+1} \Longrightarrow \dots \overset{\textit{Paso } 2n}{\Longrightarrow} \mathcal{C}^{5n-1}$$

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.

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

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.

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

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 $(\mathtt{SAT}, cod, s)$

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

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

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

$\mathbf{\Pi}$ 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 $\Pi = (\Gamma, \Sigma, \mu, \mathcal{M}_1, \dots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$, 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 $\{1, \dots, q\}$ (supondremos que 1 es la etiqueta de la raíz);
* $\mathcal{M}_1, \dots, \mathcal{M}_q$ son los multiconjuntos iniciales sobre $\Gamma \setminus \Sigma$ colocados en las q membranas (regiones delimitadas por μ) etiquetadas biyectivamente por $\{1, \dots, q\}$;
* $i_{in} \in \{1, \dots, q\}$ representa la membrana de entrada e $i_{out} \in \{0, 1, \dots, q\}$ 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 $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{AM}(-\mathbf{d})}$.

Algunos resultados importantes:

* $\mathbf{PMC}_{\mathcal{AM}^{\mathbf{0}}(-\mathbf{d},+\mathbf{ne})} = \mathbf{P}$ [1] (técnica del **grafo de dependencia**).

* $\mathtt{Subset} - \mathtt{Sum} \in \mathbf{PMC}_{\mathcal{AM}^{\mathbf{0}}(+\mathbf{d},+\mathbf{ne})}$[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): $\mathbf{PMC}_{\mathcal{AM}^{\mathbf{0}}(+\mathbf{d},-\mathbf{ne})} = \mathbf{P}$.

[1] 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.

Una solución lineal del problema 3-COL mediante sistemas P reconocedores de tejidos con división celular

Se diseña una familia de sistemas P reconocedores de tejidos con división celular $\{\Pi(t) : \ t \in \mathbb{N}\}$ que resuelve el problema 3-COL en tiempo polinomial:

- ⋆ $\Pi(t)$ procesará todos los grafos no dirigidos con n nodos y m aristas, siendo $t = \langle m, n \rangle$.
  La aplicación de $\mathbb{N} \times \mathbb{N}$ en $\mathbb{N}$ definida por $\langle m, n \rangle = \frac{(m+n)(m+n+1)}{2} + m$ es biyectiva y computable en tiempo polinomial.

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

- ⋆ $\Gamma = \{A_i, R_i, G_i, B_i, T_i, \overline{R}_i, \overline{G}_i \, \overline{B}_i, : 1 \leq i \leq n\} \cup \{c_i \ : \ 1 \leq i \leq 2n+1\} \cup$
  $\{a_i : 1 \leq i \leq 2n + m + \lceil \log m \rceil + 11\} \cup \{d_i \ : \ 1 \leq i \leq \lceil \log m \rceil + 1\} \cup$
  $\{z_i : 2 \leq i \leq m + \lceil \log m \rceil + 6\} \cup \{b, D, \overline{D}, e, T, S, N, \flat, \mathtt{yes}, \mathtt{no}\} \cup$
  $\{A_{ij}, P_{ij}, \overline{P}_{ij}, R_{ij}, G_{ij}, B_{ij} : 1 \leq i < j \leq n\}$
- ⋆ $\Sigma = \{A_{ij} \ : \ 1 \leq i < j \leq n\}$.
- ⋆ $\mathcal{E} = \Gamma \setminus \{\mathtt{yes}, \mathtt{no}\}$.
- ⋆ $\mathcal{M}_1 = \{a_1, b, c_1, \mathtt{yes}, \mathtt{no}\}$.
- ⋆ $\mathcal{M}_2 = \{D, A_1, \dots, A_n\}$.
- ⋆ $i_{in} = 2$ es la *célula de entrada*.

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

- Reglas de **división celular:**

  $r_{1,i} \equiv [A_i]_2 \rightarrow [R_i]_2[T_i]_2$ for $i = 1, \ldots, n$

  $r_{2,i} \equiv [T_i]_2 \ \rightarrow \ [G_i]_2[B_i]_2$ for $i = 1, \ldots, n$

- **Reglas de comunicación:**

  $r_{3,i} \equiv (1, a_i/a_{i+1}, 0)$ for $i = 1, \ldots, 2n + m + \lceil \log m \rceil + 10$
  $r_{4,i} \equiv (1, c_i/c_{i+1}^2, 0)$ for $i = 1, \ldots, 2n$
  $r_5 \equiv (1, c_{2n+1}/D, 2)$
  $r_6 \equiv (2, c_{2n+1}/d_1\overline{D}, 0)$
  $r_{7,i} \equiv (2, d_i/d_{i+1}^2, 0)$ for $i = 1, \ldots, \lceil \log m \rceil$
  $r_8 \equiv (2, \overline{D}/e\, z_2, 0)$
  $r_{9,i} \equiv (2, z_i/z_{i+1}, 0)$ for $i = 2, \ldots, m + \lceil \log m \rceil + 5$
  $r_{10,ij} \equiv (2, d_{\lceil \log m \rceil + 1}A_{ij}/P_{ij}, 0)$ for $1 \leq i < j \leq n$
  $r_{11,ij} \equiv (2, P_{ij}/R_{ij}\overline{P}_{ij}, 0)$ for $1 \leq i < j \leq n$
  $r_{12,ij} \equiv (2, \overline{P}_{ij}/B_{ij}G_{ij}, 0)$ for $1 \leq i < j \leq n$
  $r_{13,ij} \equiv (2, R_iR_{ij}/R_i\overline{R}_j, 0)$ for $1 \leq i < j \leq n$
  $r_{14,ij} \equiv (2, B_iB_{ij}/B_i\overline{B}_j, 0)$ for $1 \leq i < j \leq n$
  $r_{15,ij} \equiv (2, G_iG_{ij}/G_i\overline{G}_j, 0)$ for $1 \leq i < j \leq n$
  $r_{16,j} \equiv (2, \overline{R}_jR_j/\flat, 0)$ for $1 \leq j \leq n$
  $r_{17,j} \equiv (2, \overline{B}_jB_j/\flat, 0)$ for $1 \leq j \leq n$
  $r_{18,j} \equiv (2, \overline{G}_jG_j/\flat, 0)$ for $1 \leq j \leq n$
  $r_{19} \equiv (2, e\, \flat/\lambda, 0)$
  $r_{20} \equiv (2, e\ z_{m+\lceil \log m \rceil + 6}/T, 0)$
  $r_{21} \equiv (2, T/\lambda, 1)$
  $r_{22} \equiv (1, b\ T/S, 0)$
  $r_{23} \equiv (1, S\ \mathtt{yes}/\lambda, 0)$
  $r_{24} \equiv (1, b\ a_{2n+m+\lceil \log m \rceil + 11}/N, 0)$
  $r_{25} \equiv (1, N\ \mathtt{no}/\lambda, 0)$

Veamos que $\mathbf{\Pi} = \{\Pi(t) : \ t \in \mathbf{N}\}$ proporciona una solución lineal de 3-COL.

Consideramos la siguiente codificación polinomial (cod, s):

- ⋆ Para cada grafo no dirigido $G = (V, E)$, con $V = \{A_1, \ldots, A_n\}$ y $|E| = m$:
  - $s(G) = \langle m, n \rangle$.
  - $cod(G) = \{A_{ij} \ : \ \{A_i, A_j\} \in E \wedge 1 \leq i < j \leq n\}$.

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

La función cod es computable en tiempo polinomial y $cod(G)$ es un multiconjunto de entrada del sistema $\Pi(s(G))$.

El grafo no dirigido $G = (V, E)$ será procesado por el sistema de membranas $\Pi(s(G))$ con multiconjunto de entrada $cod(G)$.

Si $\mathcal{C}$ es una computación del sistema $\Pi(s(G)) + cod(G)$, 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 $\Pi(s(G)) + cod(G)$ se estructura en cuatro fases:

- Fase de **generación**:
  - Se generan todas las posibles coloraciones del grafo G aplicando reglas de división en la célula etiquetada por 2. Simultáneamente, en la célula 1 existe un contador que determinará cuándo finaliza esta fase.
- Fase de **pre-chequeo**:
  - Tras generar todas las posibles coloraciones del grafo G, esta fase proporciona objetos R_{ij}, G_{ij}, B_{ij} en las células 2 (para cada arista A_{ij}).
- Fase de **chequeo**:
  - Los objetos R_{ij}, G_{ij}, B_{ij} se usan para chequear si la coloración codificada en cada célula 2 es o no válida.
- Fase de **salida**:
  - Se produce la respuesta correcta del sistema.

Fase de generación

Se hallan todas las posibles coloraciones de G (en las células etiquetadas por 2).

- En las células etiquetadas por 2, para cada A_i (que representa el i-ésimo nodo), en dos pasos se crean 3 células, cada una de las cuales corresponde a la asignación del color *rojo*, *verde* o *azul*, respectivamente, a través de los objetos R_i, G_i, B_i.

- Simultáneamente, en la célula 1 se produce la evolución de dos contadores: a_i (que se usará en la fase de salida) y c_i (duplicándose hasta obtener 4^n copias de c_{2n+1} en el paso $2n$).

- Esta fase consume $2n$ pasos y, además, en la configuración $\mathcal{C}^{2n}$ se tiene:
  - La célula 1 contiene el multiconjunto $\{a_{2n+1}, c_{2n+1}^{4^n}, b, \mathtt{yes}, \mathtt{no}\}$.
  - Existen 3^n células etiquetadas por 2 tales que cada una contiene un objeto D, los objetos A_i (representando los vértices), los objetos A_{ij} (representando las aristas), así como una coloración distinta del grafo G.

Fase de pre-chequeo

Esta fase parte de la configuración $\mathcal{C}^{2n}$ y su objetivo es "preparar" el sistema para iniciar la fase de chequeo.

- En el paso $2n+1$, se intercambian 3^n copias de c_{2n+1} de la célula 1 con 3^n objetos D de las células 2 (regla r_5).
- En $\mathcal{C}^{2n+1}$ los objetos c_{2n+1} han llegado a las células 2 y ya comienza el proceso de comunicación de esas células con el entorno (regla r_6).
- En cada célula 2 de $\mathcal{C}^{2n+2}$ aparecerán objetos d_1 y $\overline{D}$.
- En cada célula 2 de $\mathcal{C}^{2n+2+\lceil \log m \rceil}$ aparecerán m copias del objeto $d_{\lceil \log m \rceil + 1}$, tras aplicar las reglas $r_{7,i}$, para $i = 1, \ldots, \lceil \log m \rceil$. Además, cada tal célula 2 contendrá los objetos e y $z_{\lceil \log m \rceil + 1}$, así como una coloración diferente de G y los objetos A_{ij} que codifican las aristas de G. Notaremos $\gamma = 2n + 2 + \lceil \log m \rceil$.
- Aplicando las reglas del tipo $r_{10,ij}$, los objetos $d_{\lceil \log m \rceil + 1}$ cooperan con A_{ij} para intercambiar los objetos P_{ij} con el entorno. Los correspondientes objetos P_{ij} aparecerán en las células 2 de la configuración en $\mathcal{C}^{2n+2+\lceil \log m \rceil + 1} = \mathcal{C}^{\gamma+1}$.
- Los objetos P_{ij} producirán en las células 2 de $\mathcal{C}^{\gamma+2}$, objetos R_{ij} y $\overline{P}_{ij}$ aplicando reglas del tipo r_{11}, finalizando esta fase.
- Esta fase consume $\lceil \log m \rceil + 4$ pasos y, por tanto, finaliza al llegar a la configuración $\mathcal{C}^{2n+\lceil \log m \rceil + 4} = \mathcal{C}^{\gamma+2}$.

Fase de chequeo

Esta fase parte de la configuración $\mathcal{C}^{\gamma+2}$ y su objetivo es chequear si la coloración codificada en una célula etiquetada por 2 es válida.

– Para comprobar la validez de la coloración codificada por una célula etiquetada por 2, respecto de un color (por ejemplo, el color **rojo**) se procede como sigue:

  ⋆ Si el nodo i tiene color rojo en esa célula, R_i estará presente en ella.

  ⋆ Los objetos R_i y R_{ij} de esa célula 2 de $\mathcal{C}^{\gamma+2}$, producirán objetos $R_i\overline{R}_j$ (r_{13}) en $\mathcal{C}^{\gamma+3}$, y, a la vez, los $\overline{P}_{ij}$ producirán objetos B_{ij} y G_{ij} (r_{12}) en $\mathcal{C}^{\gamma+3}$.

  ⋆ Si el nodo j tiene color *rojo* en esa célula 2, los objetos $R_j\overline{R}_j$ se intercambiarán por un objeto $\flat$ del entorno (r_{16}). Luego, esa célula 2 de $\mathcal{C}^{\gamma+4}$ el objeto $\flat$ junto con un objeto e saldrá del sistema, en el siguiente paso.

  ⋆ Si la coloración que representa es válida, el objeto e permanecerá en esa célula al final de este proceso.

– Esta fase consume $m+3$ pasos, de los cuales $m+1$ se necesitan para chequear las m cláusulas (pues los objetos B_{ij}, G_{ij} aparecen un paso después). Luego la fase finaliza con la configuración $\mathcal{C}^{\gamma+m+5}$.

Fase de salida

Esta fase consume 4 o 5 pasos de computación, dependiento de si la respuesta es afirmativa o negativa.

– Si la **respuesta** es **afirmativa**, entonces en alguna célula 2 de $C^{\gamma+m+5}$ habrá un objeto e y el objeto $z_{\lceil \log m \rceil + m + 6}$.

  ⋆ Aplicando la regla r_{20}, en esa célula 2 se produce un objeto T que se enviará a la célula 1, aplicando r_{21} en el siguiente paso. Entonces, por la regla r_{22}, ese objeto T junto con el objeto b propicia que un objeto S llegue a la célula 1 de $\mathcal{C}^{\gamma+m+8}$. Finalmente, aplicando la regla r_{23} un objeto `yes` será enviado al entorno de la configuración $\mathcal{C}^{\gamma+m+9} = \mathcal{C}^{2n+\lceil \log m \rceil + m + 11}$.

– Si la **respuesta** es **negativa**, entonces el objeto e no aparece en ninguna célula 2 de $C^{\gamma+m+5}$.

  ⋆ Aplicando la regla r_3 en los siguientes tres pasos, en la célula 1 de $C^{\gamma+m+8}$ tendremos los objetos $a_{\gamma+m+9} = a_{2n+\lceil \log m \rceil + m + 11}$ y b. Entonces, aplicando r_{24} se incorpora a la célula 1 del sistema, un objeto N del entorno. Finalmente, aplicando r_{25}, ese objeto N coopera con un objeto `no` de la célula 1 para enviar un objeto `no` al entorno de la configuración $\mathcal{C}^{\gamma+m+10} = \mathcal{C}^{2n+\lceil \log m \rceil + m + 12}$.

Verificación formal

* **Π** es polinomialmente uniforme por MT pues las reglas están descritas recursivamente y la cantidad de recursos usados para su construcción es:
  - Número total de objetos: $3n^2 + 9n + 2m + 3 \cdot \lceil \log m \rceil + 28 \in \Theta(m + n^2)$
  - Número de células iniciales 2.
  - Número inicial de objetos: $n + m + 6 \in \Theta(m + n)$.
  - Número total de reglas: $18n^2 - 9n + 2m + 3 \cdot \lceil \log m \rceil + 24 \in \Theta(m + n^2)$.
  - Máxima longitud de una regla: 4.

* **Π** es polinomialmente acotada con respecto a (cod, s): el tiempo de ejecución de cualquier computación de $\Pi(s(\varphi)) + cod(\varphi)$ es:
  - Fase de generación: $2n$ pasos.
  - Fase de pre-chequeo: $\gamma + 2$ (es decir, $2n + \lceil \log m \rceil + 4$) pasos.
  - Fase de chequeo: $m + 3$ pasos.
  - Fase de salida: 4 o 5 pasos.

Tiempo total de ejecución: $2n + \lceil \log m \rceil + m + 11$ ó $2n + \lceil \log m \rceil + m + 12$.

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

- En cualquier computación $\mathcal{C}$ de $\Pi(s(G)) + cod(G)$: para cada **coloración válida** del grafo, existe una célula 2 en $\mathcal{C}^{2n+m\lceil \log m \rceil+7}$ que codifica esa valoración y contiene e, $z_{m+\lceil \log m \rceil+6}$. Esa célula, en $\mathcal{C}^{2n+m\lceil \log m \rceil+8}$, contiene el objeto T pero no el objeto e. Por tanto, el objeto `yes` está en el entorno asociado a $\mathcal{C}^{2n+m\lceil \log m \rceil+11}$, pero no estará en ese entorno el objeto `no`.
- En cualquier computación $\mathcal{C}$ de $\Pi(s(G)) + cod(G)$: para cada **coloración no válida** del grafo, existe una célula 2 en $\mathcal{C}^{2n+m\lceil \log m \rceil+7}$ que codifica esa valoración y contiene a $z_{m+\lceil \log m \rceil+6}$ pero no al objeto e. Esa célula, en $\mathcal{C}^{2n+m\lceil \log m \rceil+8}$, contiene el objeto $z_{m+\lceil \log m \rceil+6}$ pero no el objeto T. Por tanto, el objeto `no` está en el entorno asociado a $\mathcal{C}^{2n+m\lceil \log m \rceil+12}$, pero no estará en ese entorno el objeto `yes`.
- Si una computación de $\Pi(s(G)) + cod(G)$ es de aceptación, entonces el grafo G tiene, al menos, una coloración válida con tres colores (**Adecuación**).
- Si una computación de $\Pi(s(G)) + cod(G)$ es de rechazo, entonces el grafo G carece de coloraciones válidas con tres colores (**Completitud**).

Teorema: $3 - \mathtt{COL} \in \mathbf{PMC}_{\mathcal{TDC}(4)}$.

Corolario: $\mathbf{NP} \cup \mathbf{co-NP} \subseteq \mathbf{PMC}_{\mathcal{TDC}(4)}$.

Recordando que $\mathbf{PMC}_{\mathcal{TC}} = \mathbf{P}$, se deduce que pasar de $\mathcal{TC}$ a $\mathcal{TDC}(4)$ 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 de tejidos, las reglas de **división celular** proporciona una **frontera** entre la no eficiencia y la presumible eficiencia.