*Bol. Soc. Esp. Mat. Apl.* n°29 (2004), 57–88

# Modelos de computación celular con membranas

M.J. Pérez, A. Romero y F. Sancho*

Dpto. Ciencias de la Computación e Inteligencia Artificial
E.T.S. Ingeniería Informática. Universidad de Sevilla

{marper,alvaro,fsancho}@us.es

**Resumen**

En este trabajo se presenta un nuevo modelo de computación no convencional (los sistemas celulares con membranas), inspirado en la estructura y funcionamiento de las células de los organismos vivos, y se desarrolla una teoría de la complejidad computacional en dicho modelo que permite dar una nueva caracterización de la relación $\mathbf{P} \neq \mathbf{NP}$.

**Palabras clave :** *Computación celular, Sistemas P, Sistemas celulares, Modelo no convencional, Complejidad computacional, Clases* **P** *y* **NP**.
**Clasificación por materias AMS :** *68Q05, 68Q10, 68Q15, 92B20*

## 1. Introducción

La búsqueda de procedimientos sistemáticos (*algoritmos*) que proporcionen soluciones de problemas, ha sido una constante del hombre a lo largo de la historia. G.W. Leibniz formuló la necesidad de disponer de un lenguaje universal (*lingua characteristica*) en el que poder expresar cualquier idea, así como la conveniencia de *mecanizar* cualquier tipo de razonamiento (*calculus ratiocinator*). En 1928, en el Congreso Internacional de Matemáticos, celebrado en Bolonia, D. Hilbert formuló tres cuestiones acerca de las Matemáticas que marcarían su devenir: ¿Son *completas*? ¿Son *consistentes*? ¿Son *decidibles*?

En 1931, K. Gödel presentó sus teoremas de incompletitud, mostrando que era irrealizable el Programa de Hilbert (que trataba de justificar el uso de la metodología transfinita y, al mismo tiempo, proporcionar una formalización completa de las Matemáticas). K. Gödel consideró un sistema axiomático que contenía la Aritmética elemental y construyó una proposición verdadera que no se podía probar en dicho sistema. Además, demostró que suponiendo la

*Este trabajo ha sido posible gracias a la ayuda recibida del proyecto de investigación TIC2002-04220-C03-01 del Ministerio de Ciencia y Tecnología, cofinanciado por fondos FEDER.

consistencia del mismo y la verificación de una serie de propiedades básicas, el sistema no era capaz de probar su propia consistencia. La esperanza de responder afirmativamente a la tercera cuestión planteada por D. Hilbert quedó truncada en 1936 cuando A. Church y A. Turing dieron una respuesta negativa al *Entscheidungsproblem* (problema de decisión relativo a un sistema axiomático: decidir, para cualquier aserto, si el sistema puede probarlo o no).

A lo largo del siglo XX se han desarrollado modelos abstractos para máquinas que tratan de implementar procedimientos de cálculo (máquinas de Turing, RAM, URM, etc.), para lenguajes formales (regulares, libres de contexto, recursivamente enumerables, etc.), para sistemas de programas (compiladores, sistemas operativos, etc.), para estructuras de datos (pilas, colas, montículos, etc.) y para bases de datos (relacionales, orientadas a objetos, etc.).

Definir un *modelo de computación* consiste, básicamente, en formalizar el concepto de *procedimiento mecánico*. Para ello, es necesario precisar sintácticamente qué es un tal procedimiento y definir con rigor la semántica del mismo. La semántica de un modelo de computación debe de ir acompañada del concepto de *resolución* de un problema, a través de distintos *modos* de computación (determinista, no determinista, secuencial, paralelo, aproximado, etc.).

Los trabajos de K. Gödel, A. Church, S. Kleene y A. Turing, entre 1931 y 1936, proporcionan las primeras formalizaciones del concepto de *algoritmo*, dando lugar a la aparición de los primeros *modelos de computación*. Concretamente, en 1931 K. Gödel define el concepto de *relación recursiva* e introduce la clase de funciones que denominó *recursivas* (y que hoy se conocen como *primitivas recursivas*). Posteriormente, en 1934, K. Gödel y S. Kleene extienden la clase anterior a las funciones *general recursivas* (que son las que conocemos hoy con el nombre de *recursivas*). En 1931, A. Church y S. Kleene desarrollan el concepto de *$\lambda$-cálculo* relacionándolo directamente con el concepto intuitivo de *función computable*. En 1936, A. Turing utiliza por primera vez el concepto abstracto de *máquina* a fin de formalizar la idea intuitiva de algoritmo.

Hacia mediados de la década de los treinta, A. Church y A. Turing formulan de manera independiente que las definiciones que han dado del concepto de *procedimiento mecánico* capturan completamente la idea intuitiva del mismo (*tesis de Church–Turing*). En 1936, A. Turing establece la equivalencia de su modelo y el de A. Church. Así, la tesis citada puede expresarse así: *toda función computable de manera efectiva puede ser calculada por una máquina de Turing.*

En 1936, A. Church proporciona el primer ejemplo de un problema irresoluble algorítmicamente (según la formalización del $\lambda$-cálculo): el problema de *decisión* relativo a la *lógica de primer orden*, respondiendo negativamente a la tercera cuestión formulada por D. Hilbert. Pocos meses después, A. Turing establece de manera independiente el mismo resultado probándolo en su modelo.

El auge y desarrollo de dispositivos computacionales teóricos contribuirían de manera decisiva a la construcción de los primeros ordenadores electrónicos a finales de los cuarenta del pasado siglo, basados en el modelo conceptual de J. von Neumann, A. W. Burks y H. H. Goldstine descrito en 1946 [4], en donde se formulan las bases teóricas de los futuros ordenadores que son susceptibles de manipular *programas almacenados*.

Y ya en los comienzos se empezaron a utilizar algunas ideas de la biología en el marco computacional (por ejemplo, a través de las redes neuronales artificiales de McCulloch y Pitts, en 1943). Desde entonces se han realizado grandes esfuerzos para entender, desde una óptica computacional, los procesos que se dan en la naturaleza, encaminados a la obtención de algoritmos más eficientes o incluso para el diseño de nuevos tipos de ordenadores.

La *Bioinformática* es una disciplina que estudia la aplicación de herramientas y técnicas de la información para la manipulación de datos biológicos. Entre sus objetivos destacamos el desarrollo de una base de datos específica que permita almacenar y actualizar la gran cantidad de datos que han sido y están siendo generados de manera continua e ininterrumpida; el diseño de algoritmos eficientes para la comparación de sucesiones de cadenas; la investigación de métodos de predicción de la estructura tridimensional de las moléculas orgánicas, en general, y de las proteínas, en particular; y, finalmente, la definición de modelos físicos y/o matemáticos de sistemas biológicos. y Durante la última década, el campo de investigación denominado *Computación Natural* ha experimentado un enorme desarrollo. Se trata de una disciplina cuyo objetivo es la simulación e implementación de procesos dinámicos que se dan en la naturaleza y que son susceptibles de ser interpretados como procedimientos de cálculo. Esta disciplina surge de la investigación de modelos matemáticos y de las herramientas tecnológicas necesarias para la implementación de la computación bio–inspirada. Entre las áreas que se enmarcan dentro de la Computación Natural, destacamos las siguientes:

1. Los *algoritmos genéticos* (o más en general, la *computación evolutiva*), introducidos por J. Holland [7] en 1975, que hacen uso de algunas operaciones inspiradas en la evolución y en la selección natural a fin de encontrar una *buena* solución a partir de una gran cantidad de posibles soluciones candidatas.

2. Las *redes neuronales artificiales*, introducidas por W.S. McCulloch y W. Pitts [11] en 1943, que están inspiradas en las interconexiones y en el funcionamiento de las neuronas en el cerebro.

3. La *computación molecular*, cuyo objetivo consiste en usar moléculas orgánicas (ADN, ARN, proteínas, etc.) como hardware biológico que permite realizar computaciones. Esta disciplina nace a finales de 1994 con los trabajos de L. Adleman [1], si bien tiene precedentes en un trabajo de T. Head [6] en el que formula el *sistema splicing* (modelo teórico de procesamiento de moléculas de ADN con la participación de enzimas).

4. La *computación celular con membranas*, introducida por Gh. Păun [15] en 1998, que está inspirada en la estructura y el funcionamiento de las células de los organismos vivos, en cuanto a su capacidad para procesar y generar información.

Los algoritmos genéticos y las redes neuronales artificiales han sido implementadas a través de programas en ordenadores electrónicos convencionales. La

computación molecular basada en ADN ha sido implementada en medios bioquímicos (el experimento de L. Adleman permitió resolver una instancia concreta del problema del camino hamiltoniano, en su versión dirigida y con dos nodos distinguidos, a través de la manipulación de moléculas de ADN en el laboratorio). En cambio, la computación celular con membranas aún no ha sido implementada ni en medios electrónicos ni bioquímicos.

Recientemente, en octubre de 2003, el Institute for Scientific Information (ISI, USA), ha designado a la *computación celular con membranas* como *Fast Emerging Research Front* en el área de *Computer Science*.

## 2. Una visión computacional de las células

La célula es la unidad fundamental de todo organismo vivo. Posee una estructura compleja y, a la vez, muy organizada que permite la ejecución simultánea de un gran número de reacciones químicas.

Existen dos tipos de células: las *procariotas* (propias de los organismos unicelulares, como las bacterias) que carecen de un núcleo bien definido, y las *eucariotas* (específicas de los animales y plantas) que poseen un núcleo rodeado por una doble membrana. Ambas realizan de manera similar una serie de procesos que son esenciales para la vida, tales como la replicación del ADN, la producción de energía, la síntesis de proteínas y los procesos metabólicos.

En un primer análisis podemos distinguir tres partes claramente diferenciadas en una célula: una especie de piel (*membrana plásmica*) que delimita a la célula de su entorno; el corazón propiamente dicho (*núcleo*), que almacena la información genética a través de moléculas de ADN y de ARN; y el resto de la célula, denominado *citoplasma*.

A su vez, en el citoplasma de las células eucariotas podemos destacar las siguientes componentes fundamentales: *la mitocondria*, encargada de la producción de energía para la célula; el *aparato de Golgi*, que viene a ser una fábrica de proteínas y juega un papel esencial en el metabolismo de la célula; el *retículo endoplásmico*, que es una red de membranas interconectadas estructurada en dos partes: una que facilita el paso del ARN mensajero del núcleo a fin de producir la síntesis proteínica, y otra que se encarga del transporte de moléculas y la comunicación entre las componentes de la célula; y, finalmente, los *lisosomas*, que constan de una única membrana y se encargan de digerir sustancias que proceden del exterior y de ingerir restos celulares que han llegado a ser obsoletos para el organismo

A lo largo de este trabajo se van a estudiar dispositivos computacionales inspirados en la estructura y funcionamiento de las células eucariotas.

### 2.1. Las membranas biológicas

Las *membranas biológicas* son estructuras dinámicas básicas para la célula y juegan un papel esencial a la hora de definir el fenómeno que usualmente denominamos como *vida*.

Una membrana (denominada *plasma*) separa el espacio interno de la célula protegiéndolo de su entorno. Las restantes membranas (denominadas *internas*) proporcionan la estructura jerarquizada propia de la célula y ofrecen la adecuada protección al núcleo, que es el depositario de la información genética. Las membranas están involucradas de manera decisiva en la mayoría de reacciones químicas que tienen lugar en los compartimentos celulares y pueden considerarse como barreras semipermeables que, o bien permiten el paso de algunas sustancias químicas en cualquiera de los dos sentidos (dentro–fuera o fuera–dentro), o bien bloquean el paso de sustancias. Las membranas se comportan como canales selectivos de comunicación para la transferencia de compuestos químicos entre distintos compartimentos, así como entre la propia célula y su entorno, controlando así un cierto flujo de datos (es decir, de información).

El premio Nobel de química del año 2003, ha sido otorgado a los científicos P. Agre y R. MacKinnon por sus descubrimientos relacionados con los canales de las membranas celulares, que han supuesto un gran avance en los estudios de la química celular e ilustran su importancia en los procesos vitales.

Cada compartimento celular (delimitado y separado del resto por una membrana) puede considerarse como una unidad computacional (un *procesador*) con sus propios datos (compuestos químicos) y su propio programa local (reacciones químicas), de tal manera que el conjunto de compartimentos, considerado como una unidad global (la célula), puede ser interpretado como un *modelo de computación no convencional.*

A lo largo de la breve historia de la teoría de la computación, muchos avances, tanto en el plano teórico como en el de aplicaciones prácticas, se han producido bajo la inspiración de los procesos que se dan en la naturaleza. ¿Qué puede proporcionar la célula como fuente de inspiración computacional? ¿Se puede abstraer una especie de modelo de computación a partir de las células vivas? ¿Es posible implementar computaciones a través de las células, con la esperanza de disponer algún día de un *computador celular*, de propósito general?

En las ideas originales de Gh. Păun, los sistemas celulares con membranas no fueron introducidos propiamente para modelizar completamente la estructura y funcionamiento de una célula, sino más bien para analizar algunos hechos computacionalmente relevantes que pueden ser abstraídos de las mismas.

Los sistemas P (dispositivos computacionales de los modelos celulares) constituyen un modelo teórico de computación inspirados en la realidad de las células vivas. No obstante, existe la esperanza de que la computación celular con membranas proporcione un marco de simulaciones de fenómenos moleculares. En este contexto, se está trabajando en el desarrollo del hardware y simulaciones biológicas. Así el proyecto E-CELL, iniciado en 1996, tiene como objetivo la modelización y simulación de procesos celulares tales como trayectorias metabólicas, síntesis de proteínas y transporte a través de las membranas, con el fin de predecir el comportamiento de las células vivas.

Uno de los retos más importantes planteado en este nuevo modelo consiste en encontrar una posible implementación biológica del mismo.

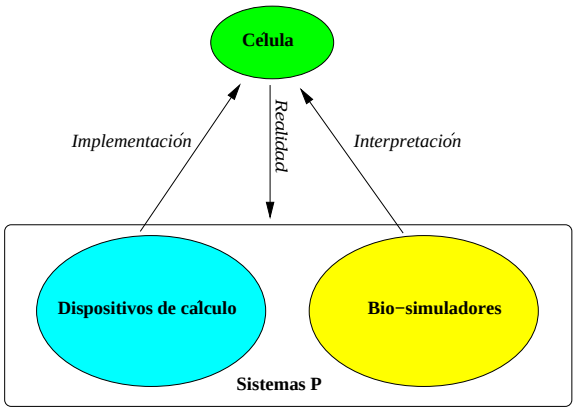


Figura 1: Sistemas celulares

## 3. Sistemas celulares: Una descripción informal

En este trabajo se trata de modelizar las células vivas y su funcionamiento a través de los *sistemas celulares con membranas*. En esta sección vamos a dar una descripción informal de los sistemas celulares básicos de transición.

Una *estructura de membranas* consta de un conjunto finito de membranas que se encuentran dentro de una membrana principal (denominada *piel*) y delimita una serie de *regiones* o compartimentos formados por el espacio comprendido entre una membrana y las situadas inmediatamente dentro de ella (si las hay). Una membrana es *elemental* si no contiene ninguna membrana.

Una estructura de membranas puede ser descrita como una cadena sobre un alfabeto finito y se puede representar mediante diagramas de Venn a través una familia de subconjuntos de un conjunto base, tal que dos conjuntos cualesquiera de la familia o son disjuntos, o uno de ellos está contenido en el otro. También se puede considerar como un árbol cuya raíz es la membrana piel y las hojas son las membranas elementales.

Las sustancias químicas contenidas en las distintas regiones de las membranas celulares se representan mediante símbolos de un alfabeto finito. Por tanto, los objetos son símbolos atómicos, indivisibles, sin *estructura interna*, pero que pueden *evolucionar* a lo largo del tiempo.

Teniendo presente que las diferentes sustancias químicas pueden aparecer en una región en diferentes cantidades, representaremos el contenido de cada región a través de *multiconjuntos* de objetos (conjuntos cuyos elementos pueden aparecer repetidos, es decir, con multiplicidades asociadas a cada uno de ellos).

Las reacciones químicas que se producen en las regiones se representan a través de *reglas de evolución* que, básicamente, son reglas de reescritura de un sistema formal que actúan sobre los objetos modificándolos y/o trasladándolos de un compartimento a otro de acuerdo con una determinada semántica.

Una *configuración* de un sistema celular consta de una estructura de membranas y una familia de multiconjuntos de objetos asociados a cada región. En la descripción de un tal sistema, aparece siempre una configuración denominada *inicial*. En algunas variantes existe una *membrana de entrada* que permite añadir un multiconjunto de objetos antes de que se produzcan evoluciones.

Las reglas de evolución serán aplicadas de manera *no determinista, paralela y maximal*; es decir, las reglas a ser usadas y los objetos involucrados en la misma serán escogidos de tal manera que en cada paso de computación todos los objetos que *puedan* evolucionar *tienen* que evolucionar. Los objetos pueden pasar de una región a otra a través de las membranas y éstas pueden ser disueltas, creadas o divididas. Así se produce una *transición* de una configuración a otra.

Una *computación* en un sistema celular es una sucesión (finita o no) de configuraciones tal que la primera de ellas es una configuración inicial y toda configuración de la sucesión que no sea la inicial se obtiene de la anterior mediante un paso de transición. Se dice que una configuración es de *parada* si no existe ninguna regla de evolución que sea aplicable a dicha configuración. Se dice que una *computación* es de *parada* si contiene una configuración de parada, y suele tener asociado un *resultado* codificado en una membrana elemental distinguida (denominada de *salida*) de la correspondiente configuración de parada.

Así pues, una computación en un sistema celular parte de una configuración y mediante sucesivas transiciones, o bien se llega a una situación de parada (en tanto que el sistema no puede seguir evolucionando), o bien, el sistema nunca se detiene (en tanto que sigue evolucionando de manera indefinida).

Admitiremos que toda computación ejecuta un proceso sincronizado; es decir, se supone que hay una especie de *reloj universal* que marca las actuaciones de todos los elementos que integran el sistema celular. Hay que destacar la existencia de dos niveles de paralelismo en la ejecución: por una parte, las reglas asociadas a una membrana son aplicadas de manera simultánea; y, por otra, estas operaciones son realizadas en el mismo instante en todas las membranas que vertebran el sistema. Es decir, en un paso de reloj se ejecutarán todas las reglas que pueden ser aplicadas (de manera maximal), y en todas las membranas.

Los sistemas celulares con membranas constituyen un modelo de computación que conjuga la elegancia y sencillez del modelo, con su proximidad a la realidad de la célula, desde un punto de vista biológico, abstrayendo de la estructura y funcionamiento de las células una serie de ideas computacionalmente relevantes. Son modelos matemáticos con propiedades atractivas desde el punto de vista computacional: algunos sistemas celulares son *completos* (tienen la misma potencia computacional que las máquinas de Turing) y *eficientes* (capaces de proporcionar soluciones polinomiales de problemas **NP**–completos).

Aunque los sistemas celulares tienen una inspiración biológica, debe resaltarse el hecho de que constituyen igualmente un buen modelo teórico de computación distribuida, en donde distintas unidades de cálculo trabajan independientemente pero estructuradas en una cierta jerarquía vertical. Por ejemplo, la jerarquización usada en las conexiones establecidas en redes de ordenadores como Internet puede ser representada como una estructura de membranas, en la que los nodos de la red se interpretan como las membranas que forman el

sistema, y el flujo de información entre nodos conectados se interpreta como las componentes químicas que dichas membranas generan e intercambian.

Asimismo, sistemas dinámicos complejos, como aquellos que surgen en el estudio de la dinámica de poblaciones, pueden ser también interpretados como sistemas celulares en los que los individuos de las distintas especies interaccionan entre sí dentro de hábitats que permiten el traspaso de los mismos.

Como primera nota diferenciadora con respecto a otros modelos de computación natural (por ejemplo, los modelos de computación molecular basados en ADN) conviene hacer notar que no está descrito a través de un lenguaje de programación; es decir, no proporciona una serie de operaciones básicas susceptibles de ser secuenciadas sobre un dato de entrada para obtener un resultado final (solución del problema que se pretende resolver), sino que se generan *dispositivos* (al modo de las máquinas de Turing) cuya ejecución modifica el contenido de las distintas componentes que los integran hasta llegar, en su caso, a un estado de parada (en el que la máquina deja de funcionar).

### 3.1. Estructuras de membranas

Recordemos que un *alfabeto*, $A$, es un conjunto no vacío, y una *cadena* sobre $A$ es una sucesión finita de elementos de $A$. Notaremos por $A^*$ el conjunto de todas las cadenas sobre $A$. Un *lenguaje* sobre $A$ es un subconjunto de $A^*$.

Para introducir las *estructuras de membranas*, en primer lugar se define recursivamente el lenguaje $MS$ sobre el alfabeto $\{[,]\}$, como sigue: es el menor lenguaje $L$ sobre $\{[,]\}$ que verifica las condiciones siguientes:

1. $[\,] \in L$.
2. Si $\mu_1, \ldots, \mu_n \in L$, entonces $[\mu_1 \ldots \mu_n] \in L$.

Intuitivamente las estructuras de membranas se describen a través de pares de corchetes *coincidentes*, imponiendo que dentro de las membranas, el orden de aparición no sea relevante. Por ello, se define una relación, $\sim$, en $MS$ de forma que estén relacionadas aquellas cadenas que reflejen la misma estructura de corchetes (salvo el orden): $x \sim y \Leftrightarrow x = \mu_1\mu_2\mu_3\mu_4 \wedge y = \mu_1\mu_3\mu_2\mu_4 \wedge \mu_1\mu_4, \mu_2, \mu_3 \in MS$. Si notamos por $\sim^*$ la clausura transitiva y reflexiva de la relación $\sim$, entonces una *estructura de membranas* es una clase de equivalencia de $MS$ con respecto a la relación $\sim^*$. Dada una estructura de membranas, a cada par de *corchetes coincidentes* se le denomina *membrana*. El número de membranas de una estructura de membranas, $\mu$, se denomina *grado* de la estructura. A la estructura *más externa* se le llama *piel*, y aquellas membranas que no contienen otras membranas en su interior se denominan *elementales*.

Se puede representar gráficamente la estructura de membranas como un conjunto de curvas simples cerradas (a modo de *diagrama de Venn*) manteniendo la jerarquía impuesta por la cadena que la representa en el conjunto cociente $\overline{MS} = MS/\sim^*$ (figura 2).

A partir de esta representación surge el concepto intuitivo de *región* delimitada por una membrana, como el espacio comprendido entre dicha

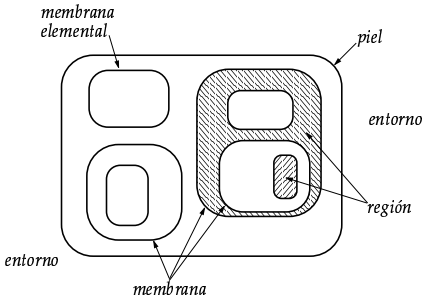


Figura 2: Representación de una estructura de membranas.

membrana y las inmediatamente interiores. Existe una biyección natural entre las membranas de una estructura y las regiones que delimitan, siendo habitual identificar unas con otras.

### 3.2. Sistemas celulares de transición

A continuación se introducen los sistemas P de transición siguiendo las ideas del trabajo fundacional de Gh. Păun[15].

**Definición 1** *Un sistema P de transición de grado $p \geq 1$ es una tupla*

$$\Pi = \left(\Gamma, \mu_\Pi, \mathcal{M}_1, \ldots, \mathcal{M}_p, (R_1, \rho_1), \ldots, (R_p, \rho_p), i_0\right)$$

*donde:*

- $\Gamma$ *es un alfabeto finito (alfabeto de trabajo).*
- $\mu_\Pi$ *es una estructura de membranas (que consta de $p$ membranas). Las membranas de $\mu_\Pi$ están etiquetadas de forma unívoca usando los números naturales desde 1 hasta $p$.*
- $\mathcal{M}_i$ $(1 \leq i \leq p)$ *es un multiconjunto finito sobre $\Gamma$ asociado a la membrana $i$ del sistema.*
- $R_i$ $(1 \leq i \leq p)$ *es un conjunto finito de reglas de evolución de transición asociado a la membrana $i$ del sistema. Una regla de evolución de transición es un par $(u, v)$, habitualmente representado $u \to v$, donde $u$ es una cadena sobre $\Gamma$ y $v = v'$ o $v = v'\delta$, siendo $v'$ una cadena sobre $\Gamma \times \left(\{here, out\} \cup \{in_i : i = 1, \ldots, p\}\right)$.*
- $\rho_i$ $(1 \leq i \leq p)$ *es un orden parcial estricto sobre $R_i$, que establece una relación de prioridad entre las reglas de $R_i$.*
- $i_0$ *es un número natural entre 1 y $p$ (membrana de salida del sistema).*

Para describir de manera informal la semántica del modelo se introduce el concepto de configuración, de donde seguirá, una vez establecida la forma en que se aplican las reglas de transición, la noción de computación del sistema. Una *configuración* de $\Pi$ es una tupla $(\mu, M_{i_1}, \ldots, M_{i_q})$ tal que $\mu$ es la estructura de membranas que se obtiene de $\mu_\Pi$ al eliminar las membranas distintas de $i_1$ a $i_q$ (la membrana piel no se puede eliminar, por lo que coincide en ambas estructuras); y $M_{i_j}$ es un multiconjunto finito sobre $\Gamma$, para cada $j = 1, \ldots, q$.

La *configuración inicial* de $\Pi$ es la tupla $(\mu_\Pi, \mathcal{M}_1, \ldots, \mathcal{M}_p)$.

La *ejecución de una regla* $u \to v$ asociada a una membrana $i$ presente en una configuración se realiza como sigue: los objetos en $u$ se eliminan de la membrana $i$ (ésta debe contener, por tanto, suficientes objetos para que la regla se pueda aplicar); entonces, para cada $(a, out) \in v$ un objeto $a \in \Gamma$ se incluye en la membrana padre de la membrana $i$ (o abandona el sistema si la membrana $i$ es la piel); para cada $(a, here) \in v$ un objeto $a \in \Gamma$ se añade a la membrana $i$; para cada $(a, in_j) \in v$ un objeto $a \in \Gamma$ se introduce en la membrana $j$ (teniendo en cuenta que si la membrana $j$ no es hija de la membrana $i$, entonces la regla no se puede aplicar); finalmente, si $\delta \in v$, entonces la membrana $i$ se disuelve; es decir, se elimina de la estructura de membranas, pasando sus objetos a su membrana padre y desapareciendo las reglas de evolución y relaciones de prioridad asociadas a ella (la membrana piel no se puede disolver).

Dada una regla $u \to v$, la longitud de la cadena $u$ se denomina *radio* de la regla. Se dice que un sistema celular es *cooperativo* si posee, al menos, una regla de evolución de radio mayor que uno.

Las *relaciones de prioridad* dadas sobre las reglas se interpretarán como sigue: si la regla $r_1$ tiene mayor prioridad que la regla $r_2$ y se puede aplicar $r_1$, entonces no se aplicará $r_2$, aunque fuera factible. Una posible interpretación de esta versión se encuentra en el consumo de energía: en cada paso de transición tenemos una cantidad fija de energía para poder aplicar las reglas, de tal manera que las reglas de prioridad superior consumen la suficiente energía como para que no quede energía para reglas de prioridad inferior.

Dadas dos configuraciones, $C$ y $C'$, de $\Pi$, diremos que $C'$ se obtiene de $C$ en un *paso de transición*, y notaremos $C \Rightarrow_\Pi C'$, si $C'$ es el resultado de aplicar a $C$, en *paralelo* y para todas las membranas al mismo tiempo, las reglas de evolución contenidas en un determinado (multi)conjunto de reglas asociadas a las membranas que aparecen en la estructura de membranas de $C$. En dicho (multi)conjunto se indica el número de veces que se aplica cada una de las reglas y debe ser *maximal*, en el sentido de que, tras la aplicación de las reglas, en ninguna membrana permanezcan objetos sin evolucionar que puedan activar alguna de las reglas asociada a la membrana. Puesto que para una configuración usualmente existe más de un (multi)conjunto de reglas aplicable, los pasos de transición se realizan de manera no determinista.

Una *computación* $\mathcal{C}$ de $\Pi$ es una sucesión (finita o no) de configuraciones, $\{C^i\}_{i<r}$, tal que $C^0$ es la configuración inicial de $\Pi$; $C^i \Rightarrow_\Pi C^{i+1}$, para todo $i < r-1$; y, o bien $r$ es un número natural no nulo y no existe una regla que se pueda aplicar en ninguna de las membranas de $C^{r-1}$ (en cuyo caso se dice que $\mathcal{C}$ es de parada, ha realizado $r-1$ pasos y $C^{r-1}$ es su *configuración de parada*),

o bien $r = \infty$ (en cuyo caso se dice que $\mathcal{C}$ no es de parada).

Diremos que una *computación* $\mathcal{C}$ es *exitosa* si es una computación de parada y, además, la membrana de salida $i_0$ aparece en $C^{r-1}$ como membrana elemental. A cada computación exitosa de $\Pi$, se le puede asociar una salida (por ejemplo, el tamaño del multiconjunto asociado a la membrana de salida del sistema en su configuración de parada). Entonces, un sistema P de transición es una máquina que genera el conjunto de números naturales formado por las salidas de todas las computaciones exitosas de dicho sistema.

A partir del modelo de computación celular básico se han diseñado múltiples variantes. A continuación hacemos un breve análisis de algunas de las variantes que se han considerado hasta la actualidad.

### 3.3. Variantes en los objetos

En los sistemas celulares de transición vistos hasta ahora se puede interpretar que no se hace uso de estructuras de datos para codificar la información: los números son codificados a través de la cardinalidad de multiconjuntos; por consiguiente, están representados en base 1. Esto puede ser adecuado para una implementación bioquímica, pero parece claramente insuficiente desde un punto de vista clásico. Más aún, por esta vía no se podrá trabajar (directamente, sin una codificación numérica) con computaciones simbólicas.

Por ello, se trata de usar sistemas que representen la información a través de una estructura de datos de tipo estándar: las cadenas. Así, en lugar de considerar objetos de tipo atómico se pueden usar objetos que pueden ser descritos por cadenas sobre un alfabeto finito prefijado. En tal situación, la evolución de un objeto correspondería propiamente a una transformación de la cadena.

#### 3.3.1. Sistemas celulares con reescritura.

En estos sistemas celulares se procesan las cadenas de objetos a través de reglas del tipo $r \equiv X \to (v; tar)$, donde $X$ es un símbolo, $v$ es una cadena de símbolos y $tar$ es la membrana destino de dicha cadena (eventualmente, la cadena $v$ puede ir seguida de un símbolo $\delta$, de disolución). La forma de considerar la ejecución de este tipo de reglas (que actúan en el *interior* de las cadenas) es la habitual en sistemas P (la aplicación de $r$ sobre una cadena $w$ reemplaza una ocurrencia de $X$ por la cadena $v$, disolviendo la membrana si así se indica en $r$), con la siguiente puntualización: todas las cadenas son procesadas en paralelo, pero sobre cada cadena presente en la membrana sólo actúa una regla [9].

#### 3.3.2. Sistemas celulares con replicación de cadenas.

Este tipo de sistemas es una extensión del procedimiento de reescritura anterior: las reglas combinan la posibilidad de actuar en el interior de las cadenas con la capacidad de replicación de las mismas [10]. De esta forma, una regla adquiere la expresión general $r \equiv X \to (u_1, tar_1)||\ldots||(u_n, tar_n)$, y su aplicación sobre una cadena $w$ produce la aparición de $n$ cadenas, $w_1, \ldots, w_n$, de modo que $w_i$ se obtiene de $w$ reemplazando una aparición de $X$ por $u_i$, y su destino es $tar_i$.

Esta variante de los sistemas P de transición ha sido usada para dar soluciones deterministas, y con coste en tiempo polinomial, de problemas **NP**-completos clásicos, como son el problema **SAT** y el problema **HPP** del camino hamiltoniano [25]. No obstante, la parte exponencial del proceso queda encubierta en la aplicación de reglas que permiten la replicación de las cadenas.

### 3.4. Variantes en las membranas

Otra posibilidad de mejorar las características de los sistemas celulares es añadir funcionalidad y dinamismo a la estructura de membranas, permitiendo la creación y división de membranas por medio de reglas.

#### 3.4.1. División de Membranas.

Este tipo de variante es conocida por el nombre de *sistemas celulares con membranas activas*. Se trata de un dispositivo útil para conseguir una cantidad exponencial de espacio, en tiempo polinomial, partiendo de una estructura de membranas descrita en forma polinomial (en función del dato de entrada).

En esta variante se permite el uso de reglas que implementan la división (análoga a la producida en la naturaleza) de membranas elementales (siempre que no sea la piel) en otras dos que contienen un duplicado del contenido de la membrana original (salvo el objeto que dispara la regla, que evoluciona a otro objeto en las nuevas membranas), y tienen asociadas el mismo conjunto de reglas. Se admite disolución, pero no cooperación ni prioridad entre reglas.

Si en esta variante se permite la división de membranas no elementales, el sistema celular se denomina *extendido*. Se prueba [13] que el problema **HPP** puede ser resuelto en tiempo cuadrático y el problema **SAT** en tiempo lineal, por medio de sistemas P extendidos con membranas activas.

Este resultado ha sido mejorado por C. Zandron, C. Ferretti, G. Mauri [26], y por M.J. Pérez Jiménez, A. Romero Jiménez y F. Sancho Caparrini [23] que han obtenido soluciones lineales a través de sistemas celulares que sólo usan reglas de división para las membranas elementales.

#### 3.4.2. Creación de membranas.

La creación de membranas es un proceso habitual en la biología. En relación con los sistemas celulares, se considera la posibilidad de crear nuevas membranas por la acción de ciertos objetos. En este sentido, se pueden introducir reglas en las que un determinado multiconjunto de objetos produce, en el interior de la membrana a la que pertenece, una nueva membrana con un cierto contenido. Este tipo de creación de membranas puede ser aplicado tanto a sistemas que usan objetos atómicos como a aquellos que usan objetos de tipo cadena.

Esta variante de sistemas fue considerada por M. Ito, C. Martín–Vide y Gh. Păun [8] para conseguir resultados de universalidad en los que se permitía un uso bastante moderado de esta nueva operación. También puede ser usada para la obtención de soluciones polinomiales de problemas **NP**-completos.

### 3.5. Variantes en las comunicaciones

Hay un tercer tipo de variantes que estudian el papel que juega la comunicación entre las membranas (por medio de los elementos que son transferidos de unas a otras). Para ello, se intenta debilitar alguna capacidad inherente a los sistemas básicos de transición con el fin de estudiar cuáles son las características mínimas necesarias para seguir disponiendo de un modelo universal (lo que podría ser útil si se consigue una implementación práctica de estos modelos).

Una primera variante consiste en cambiar el destino $in_j$ en las reglas de los sistemas celulares por un destino más general, $in$, que no especifique a qué membrana han de ir los objetos producidos, de modo que dicha elección es no determinista entre las posibles membranas existentes que sean hijas de la membrana a la que está asociada la regla. Se debe hacer notar que muchos de los resultados de universalidad obtenidos para variantes mejoradas se siguen manteniendo para esta variante débil de sistemas P, debido a que en dichas pruebas se hace uso de estructuras de membranas de grado 2, en las que no es necesario especificar qué membrana hija es la que recibe los objetos.

Otra variante introducida en esta dirección consiste en dotar de una carga eléctrica a los objetos y membranas del sistema P (con tres cargas posibles: $+, -, 0$), de tal modo que se elige *a priori* una carga en las membranas (en la definición del sistema celular) y los objetos pueden modificar sus cargas de acuerdo con las reglas de evolución que se aplican sobre ellos. Así, la relación entre las cargas de los objetos y las membranas determina si un objeto puede atravesar una membrana. Para más detalles acerca de estas dos variantes, ver [18], capítulo 3, sección 3.6.1.

Por otra parte, es interesante observar que las membranas biológicas tienen una permeabilidad que es variable y, por tanto, se puede considerar un modelo en el que se controle el paso de objetos de una membrana a otra mediante un parámetro de *permeabilidad de una membrana*, como valor numérico que indica su proximidad a un estado que permite su disolución. En dichos sistemas [17], se puede modificar esa permeabilidad por medio de reglas, de tal manera que la ejecución de una regla puede hacer que la membrana engrose o adelgace. Si una membrana tiene un grosor muy elevado, se vuelve *impermeable*, impidiendo la transferencia de objetos a través de ella. Así por ejemplo, el control de la permeabilidad de las membranas se puede implementar introduciendo una acción, que notaremos $\tau$, y de tal manera que es opuesta a $\delta$, en el sentido de que *engrosa* la membrana, la aleja de la disolución.

En los *sistemas celulares con transportadores* (*carriers*) [25], los objetos no son transformados, sino que únicamente son transmitidos entre las membranas de acuerdo con ciertas reglas de transferencia. Usualmente, en esta variante se permite la transmisión con objetos del *entorno* del sistema, con el fin de poder disponer de una cantidad ilimitada de los mismos.

Una variante interesante de los sistemas con transportadores son los *sistemas celulares con porteros* (*porters*) [14], en los que se restringe la capacidad en la transmisión de objetos, reduciendo el *radio* de las reglas.

## 4. Una formalización de los sistemas P de transición

El problema de describir una formalización completa de los sistemas celulares, así como de sus computaciones y los resultados de las mismas, fue formulado explícitamente por Gh. Păun en [16]. En este trabajo se presenta una formalización con una orientación distinta de las propuestas parciales dadas por A. V. Baranda y otros en [2] y por A. Obtulowicz en [12]. Asimismo es ligeramente diferente de la presentada por M. J. Pérez Jiménez y F. Sancho en [20].

### 4.1. Una sintaxis para los sistemas celulares de transición

En primer lugar, vamos a introducir una serie de conceptos que serán de utilidad para la formalización que se va a presentar.

Un *multiconjunto* sobre un conjunto, $A$, es una aplicación $m : A \to \mathbb{N}$, siendo $\mathbb{N}$ el conjunto de los números naturales. Un subconjunto $B \subseteq A$ se puede identificar con el multiconjunto sobre $A$ dado por la función característica de $B$ (en $A$). Notaremos por $\mathbf{M}(A)$ el conjunto de todos los multiconjuntos sobre $A$. El *soporte* de un multiconjunto $m$ es el conjunto $\{a \in A : m(a) > 0\}$. Un *multiconjunto* se dice que es *finito* si su soporte es un conjunto finito, y se dice que es vacío si lo es su soporte. El *tamaño* de un multiconjunto finito, $m$, se define por $|m| = \sum_{a \in A} m(a)$ (obsérvese que dicha suma es finita).

**Definición 2** *Sean $m_1$ y $m_2$ multiconjuntos sobre $A$.*

- *Inclusión:* $m_1 \leq m_2 \Leftrightarrow \forall\, j\ (j \in A \to m_1(j) \leq m_2(j))$.
- *Inclusión estricta:* $m_1 < m_2 \Leftrightarrow m_1 \leq m_2 \wedge m_1 \neq m_2$.
- *Unión: La aplicación* $+ : \mathbf{M}(A) \times \mathbf{M}(A) \to \mathbf{M}(A)$, *se define como sigue:* $(m_1 + m_2)(j) = m_1(j) + m_2(j)$, *para cada* $j \in A$.
- *Diferencia: La aplicación* $- : \mathbf{M}(A) \times \mathbf{M}(A) \to \mathbf{M}(A)$, *se define como sigue:* $(m_1 - m_2)(j) = \text{máx}\{m_1(j) - m_2(j), 0\}$, *para cada* $j \in A$.
- *Amplificación: La aplicación* $\otimes : \mathbb{N} \times \mathbf{M}(A) \to \mathbf{M}(A)$, *se define como sigue:* $(n \otimes m_1)(j) = n \cdot m_1(j)$, *para cada* $j \in A$.

Todo multiconjunto finito, $m$, con soporte $\{a_1, \ldots, a_n\}$, se puede representar mediante una cadena $w$ sobre $A$ en donde $a_i$ aparece exactamente $m(a_i)$ veces. Las cadenas obtenidas a partir de $w$ permutando sus símbolos también representan a $m$. El multiconjunto vacío se representa por la cadena vacía, $\lambda$.

Dado un grafo no dirigido, $G$, diremos que un nodo $v$ es *alcanzable* desde otro nodo $u$, y notaremos por $u \rightsquigarrow_G v$, si existe un camino desde $u$ hasta $v$.

Sea $G = (V, E)$ un árbol enraizado de raíz $r$. Para cada nodo, $u$, que no sea la raíz, notaremos por $f(u)$ (el *padre* de $u$ en $G$) el único antecesor propio de $u$ que verifica $\{f(u), u\} \in E$. Asimismo, notaremos por $Ch(u)$ (los *hijos* de $u$ en $G$) el conjunto de descendientes propios de $u$ tales que $u$ es su padre. Además, se puede definir de manera natural una relación binaria $E^*(G)$ en $V$ como sigue: $(x, y) \in E^*(G) \Leftrightarrow y \in Ch(x)$. Esta relación establece, de alguna forma, un direccionamiento sobre las aristas del árbol enraizado.

**Definición 3** *Una* estructura de membranas *es un árbol enraizado en el que los nodos se denominan* membranas*, la raíz se denomina* piel*, y las hojas se denominan* membranas elementales.

En esta formalización las etiquetas de las membranas serán los propios vértices del grafo. Si esta formalización se extiende a otras versiones de sistemas celulares, puede ser útil un etiquetado independiente de los nodos; no obstante, para la versión que estamos formalizando esto no es necesario.

**Definición 4** *Una* célula *sobre un alfabeto, $A$, es un par $(\mu, M)$, en donde $\mu$ es una estructura de membranas y $M$ es una aplicación de $V(\mu)$ en $\mathbf{M}(A)$, por la que a cada nodo del árbol se le asocia un multiconjunto sobre el alfabeto base.*

Si $C = (\mu, M)$ es una célula sobre $A$, denotaremos $\mu = (V(\mu), E(\mu))$; es decir, $V(\mu)$ y $E(\mu)$ son los conjuntos de vértices y aristas, respectivamente, del árbol enraizado etiquetado, $\mu$, que determina la célula. El número total de membranas (o nodos) de la célula, se denomina *grado* de dicha célula.

**Definición 5** *Sea $(\mu, M)$ una célula sobre un alfabeto $A$. Una* regla (de evolución) *asociada a dicha célula es una tupla $r = (d_r, v_r, \delta_r, x_r)$, en donde: $d_r$ es un multiconjunto sobre $A$; $v_r$ es una función de dominio $V(\mu) \cup \{here, out\}$ y rango contenido en $\mathbf{M}(A)$, en donde $here, out \notin V(\mu)$ y $here \neq out$; $\delta_r \in \{\neg\delta, \delta\}$, con $\neg\delta, \delta \notin A$ y $\neg\delta \neq \delta$; y $x_r$ es un nodo de $\mu$.*

**Definición 6** *Sea $C = (\mu, M)$ una célula sobre un alfabeto y $x \in V(\mu)$. Sea $r = (d_r, v_r, \delta_r, x_r)$ una regla asociada a $C$. Diremos que $r$ está asociada a $x$ si se verifica que $x_r = x$.*

Finalmente, para dar la sintaxis completa del sistema P se necesita asignar a cada membrana de la estructura un conjunto de reglas que serán las que se puedan aplicar a los elementos presentes en esa membrana.

**Definición 7** *Sea $C = (\mu, M)$ una célula sobre un alfabeto $A$. Una* colección $R$ *de* reglas *asociada a $C$ es una función con dominio $V(\mu)$ tal que para cada membrana $x \in V(\mu)$, $R(x)$ (que notaremos $R_x$) es un conjunto finito (posiblemente vacío) de reglas asociadas a $x$.*

**Definición 8** *Sea $C = (\mu, M)$ una célula sobre un alfabeto, $A$, y sea $R$ una colección de reglas asociada a $C$. Una* relación de prioridad sobre $R$ *es una función, $\rho$, con dominio en $V(\mu)$ tal que para cada membrana $x \in V(\mu)$, $\rho(x)$ (que notaremos $\rho_x$) es un orden parcial estricto (posiblemente vacío) sobre $R_x$.*

El orden parcial estricto, $\rho_x$, sobre $R_x$, se interpretará como sigue: $(r', r) \in \rho_x$ significa que la regla $r'$ tiene una prioridad estrictamente mayor que la regla $r$.

**Definición 9** *Un sistema $P$ de transición de grado $p \geq 1$ es una tupla, $\Pi = (A, C_0, \mathcal{R}, i_0)$, en donde:*

- *$A$ es un conjunto finito no vacío (alfabeto de trabajo).*

- $C_0 = (\mu_\Pi, M_0)$ *es una célula sobre* $A$*, de grado* $p$*.*
- $\mathcal{R}$ *es un par ordenado* $(R, \rho)$ *donde* $R$ *es una colección de reglas (de evolución) asociadas a* $C_0$*, y* $\rho$ *es una relación de prioridad sobre* $R$*.*
- $i_0$ *es un nodo del árbol enraizado* $\mu_\Pi$*, que especifica la membrana de salida del sistema* $P$ *de transición* $\Pi$*.*

### 4.2. Una semántica para los sistemas celulares de transición

La semántica de un sistema P de transición trata de describir la forma en que funciona como dispositivo computacional. Para definir formalmente la semántica se necesita el concepto de *configuración*, que intenta capturar cada uno de los estados puntuales, instantáneos, que puede alcanzar un sistema a lo largo de su ejecución/evolución. Un sistema P consta de una estructura de membranas con objetos en su interior, con reglas de evolución propias para cada membrana y con especificaciones de entrada–salida. Un sistema celular tiene como únicos elementos variables, la propia estructura de membranas (debido a la operación de disolución) y los objetos contenidos en cada una de las membranas. Por tanto, los elementos básicos para describir una configuración de un sistema P en un instante de su evolución serán las estructuras de membranas y los objetos. Intuitivamente, las configuraciones contienen una descripción completa del estado actual de la ejecución; por ello, van a jugar un papel fundamental en el análisis y discusión de las computaciones de los sistemas celulares.

**Definición 10** *Sea* $\Pi = (A, C_0, \mathcal{R}, i_0)$ *un sistema P de transición, con* $C_0 = (\mu_\Pi, M_0)$*. Una* configuración *de* $\Pi$ *es una célula* $C = (\mu, M)$ *sobre* $A$*, donde* $V(\mu) \subseteq V(\mu_\Pi)$*, y de tal manera que los árboles* $\mu$ *y* $\mu_\Pi$ *tienen la misma raíz.*

De la definición anterior resulta que $C_0$ es, en particular, una configuración de $\Pi$, denominada *configuración inicial* del sistema. Para cada $i \in V(\mu)$, diremos que $\mathcal{M}_i = M_0(i)$ es el *multiconjunto inicial* asociado a la membrana $i$.

A continuación estudiamos cómo decidir si una regla puede ser aplicada o no en un instante determinado. Informalmente, una regla de evolución, $r = (d_r, v_r, \delta_r, x_r)$ de una membrana $x \in V(\mu)$ se puede expresar como sigue:

$$a_1 \dots a_m \to (b_1, in_{j_1}) \dots (b_n, in_{j_n})(c_1, out) \dots (c_k, out)(h_1, here) \dots (h_l, here)s$$

con $j_1, \dots, j_n \in V(\mu)$; $a_i, b_i, c_i, h_i \in A$; y $s = \delta$ ó $s = \lambda$.

En esta formalización $d_r = a_1 \dots a_m$; para cada $y \in V(\mu)$, $v_r(y)$ es el multiconjunto formado por los elementos que aparecen en la regla bajo la forma $(b, in_y)$; $v_r(out) = c_1 \dots c_k$, y $v_r(here) = h_1 \dots h_l$. La componente $\delta_r$ ($\delta$ o $\neg\delta$) representa si $s = \delta$ o $s = \lambda$. Por último, la componente $x_r$ representa la membrana a la que está asociada la regla.
Entonces

- Esta regla puede ser aplicada sólo cuando hay suficientes copias de los objetos $a_1, \dots, a_m$ en la membrana $x$ y, además, ninguna regla de $R_x$ con prioridad superior es aplicable.

- El resultado de usar esta regla viene determinado por su lado derecho:
  - Los objetos $h_1, \ldots, h_l$ van a parar a la misma región $x$.
  - Los objetos $c_1, \ldots, c_k$ van a parar a la región inmediatamente exterior a $x$ (es decir, al entorno, si que $x$ sea la piel, o bien al padre de $x$, en caso contrario).
  - Cada objeto $b_i$ pasará a formar parte del multiconjunto asociado a la membrana $j_i$, en el caso en que $j_i$ fuese un hijo de la membrana $x$. Caso contrario, la regla no sería aplicable.
  - Si el símbolo $\delta$ aparece en la regla (es decir, si $s = \delta$), entonces la membrana $x$ se elimina (se *disuelve*) y el conjunto de reglas $R_x$ (junto con su relación de prioridad asociada) no volverá a ser usado. El multiconjunto asociado a $x$ se añade a la región inmediatamente exterior a ella (su padre). Una regla que contenga el símbolo $\delta$ no puede ser aplicada en la membrana piel.

Notaremos por $\mathbf{R} = \bigcup_{x \in V(\mu_0)} R_x$, al conjunto de todas las reglas de evolución asociadas a la configuración inicial (es decir, al sistema P de transición).

**Definición 11** *Sea $C = (\mu, M)$ una configuración de un sistema P de transición $\Pi = (A, C_0, \mathcal{R}, i_0)$, con $C_0 = (\mu_\Pi, M_0)$. Diremos que la regla (de evolución) $r = (d_r, v_r, \delta_r, x_r) \in \mathbf{R}$ es* aplicable *a $C$ si:*

- *La membrana $x_r$ a la que está asociada existe en la célula $C$: $x_r \in V(\mu)$.*
- *No intenta disolver el nodo raíz: si $x_r$ es el nodo raíz de $\mu$, $\delta_r = \neg\delta$.*
- *La membrana $x_r$ tiene los objetos necesarios: $d_r \leq M(x_r)$.*
- *Los nodos a los que la regla intenta enviar objetos son hijos de la membrana a la que pertenece la regla: $\forall j \in V(\mu)(v_r(j) \neq \emptyset \rightarrow j \in Ch(x_r))$.*
- *No hay otras reglas aplicables a $C$ con prioridad mayor: no existe una regla $r'$ tal que $\rho_{x_r}(r', r) \wedge r'$ aplicable a $C$.*

A continuación, se define el *índice de aplicabilidad de una regla $r$ en una configuración $C$*, que notaremos $N_{Ap}(r, C)$, y que nos va a indicar el máximo número de veces que la regla $r$ puede ser aplicada a $C$. Es decir,

$$N_{Ap}(r, C) = \begin{cases} 0 & \text{, si } r \text{ no es aplicable a } C \\ \max\{n \ : n \otimes d_r \leq M(x_r)\} & \text{, en otro caso.} \end{cases}$$

Seguidamente, se introduce un concepto que representa un multiconjunto sobre $\mathbf{R}$ de reglas que indica, con la multiplicidad asociada a cada regla, cuántas veces ha de aplicarse dicha regla a fin de realizar un paso de transición en el sistema.

**Definición 12** *Sea $C = (\mu, M)$ una configuración de un sistema P de transición $\Pi = (A, C_0, \mathcal{R}, i_0)$. Diremos que un multiconjunto $m$ sobre $\mathbf{R}$ es de aplicabilidad sobre $C$, y lo notaremos por $m \in \mathbf{M_{Ap}}(C)$, si :*

- $\forall\, r \in \mathbf{R}\ (m(r) \leq N_{Ap}(r, C))$.
- $\forall\, x \in V(\mu_0)\ (\sum_{r \in R_x} m(r) \otimes d_r \leq M(x))$.
- *Es maximal, en el sentido siguiente: no existe un multiconjunto $m'$ sobre* $\mathbf{R}$ *tal que* $m < m' \wedge m' \in \mathbf{M_{Ap}}(C)$.

Para determinar la configuración siguiente de una configuración, $C$, por un multiconjunto, $m$, de aplicabilidad sobre $C$, se procede en dos etapas. En la primera, las reglas se ejecutan sin atender a la disolución. En la segunda, se disuelve cada membrana que se tenga que disolver, se distribuye su contenido a su primer antecesor no disuelto, y se actualiza el árbol enraizado que resulta.

La primera etapa (aplicación de las reglas sin atender a la disolución) produce la configuración $C'' = (\mu, M'')$, en donde:

$$M''(x) = M(x) - \sum_{r \in R_x} m(r) \otimes d_r + \sum_{r \in R_x} m(r) \otimes v_r(here) + \sum_{r \in R_{f(x)}} m(r) \otimes v_r(x) + \sum_{y \in C\, h(x)} \sum_{r \in R_x} m(r) \otimes v_r(out)$$

Para llevar a cabo la segunda etapa (disolución y distribución de los contenidos) necesitamos caracterizar, previamente, los nodos que deben disolverse por la ejecución de un multiconjunto de aplicabilidad.

**Definición 13** *Sea $C = (\mu, M)$ una configuración de un sistema $P$ de transición $\Pi$, y $m \in \mathbf{M_{Ap}}(C)$ un multiconjunto de aplicabilidad sobre $C$. Se define el conjunto:* $\Delta(m, C) = \{x_r :\ r \in \mathbf{R} \wedge m(r) > 0 \wedge \delta_r = \delta\}$.

Es decir, $\Delta(m, C)$ representa el conjunto de nodos de la estructura de membranas de $C$ que deben ser disueltos tras la aplicación del multiconjunto $m$.

Hay que tener presente que en un paso del sistema celular de transición, debido al paralelismo, se puede disolver más de una membrana de la estructura. Por tanto, será conveniente determinar para cada membrana, $x$, que permanezca en la próxima configuración, cuál es el conjunto de membranas que se disuelven por la aplicación de un cierto multiconjunto de aplicabilidad y añaden su contenido a $x$ (tales membranas se llamarán *donantes* de $x$).

**Definición 14** *Sea $C = (\mu, M)$ una configuración de un sistema $P$ de transición $\Pi$. Sea $m \in \mathbf{M_{Ap}}(C)$ un multiconjunto de aplicabilidad sobre $C$. Para cada nodo $x \in V(\mu)$, se define el conjunto $Don(x, m, C)$ de los* donantes *de $x$ para $C$ por la aplicación de $m$, como sigue:*

$$\begin{cases} \emptyset & , si\ x \in \Delta(m, C) \\ \{y \in V(\mu) : y \in \Delta(m, C) \wedge x \rightsquigarrow_\mu y \wedge \\ \wedge\ \forall\, z \in V(\mu)(x \rightsquigarrow_\mu z \rightsquigarrow_\mu y \rightarrow z \in \Delta(m, C))\} & , si\ x \notin \Delta(m, C) \end{cases}$$

Es decir, si un nodo $x$ no se disuelve, entonces el nodo $y$ es donante de $x$ si $y$ se disuelve por la aplicación de $m$, así como todo nodo que sea descendiente propio de $x$ y antecesor propio de $y$.

A continuación se define la ejecución de un multiconjunto de aplicabilidad sobre una configuración dada.

**Definición 15** *Sea $C = (\mu, M)$ una configuración de un sistema $P$ de transición $\Pi$. Sea $m \in \mathbf{M_{Ap}}(C)$ un multiconjunto de aplicabilidad sobre $C$. Se define la* ejecución *de $m$ sobre $C$, y se notará por $m(C)$, como la configuración $C' = (\mu', M')$ de $\Pi$, en donde:*

- *$\mu' = (V(\mu'), E(\mu'))$ es el árbol enraizado obtenido de $\mu$ como sigue:*
  - *$V(\mu') = V(\mu) - \Delta(m, C)$*
  - *Si $x, y \in V(\mu')$, entonces $(x, y) \in E^*(\mu')$ si y sólo si existen $x_0, \ldots, x_n \in V(\mu)$ tales que $(x_1, \ldots, x_{n-1} \in \Delta(m, C) \wedge x_0 = x \wedge x_n = y \wedge \forall i\ (0 \leq i < n \rightarrow (x_i, x_{i+1}) \in E^*(\mu))$*
- $$M'(x) = \begin{cases} M''(x) \cup \bigcup_{y \in Don(x,m,C)} M''(y) & , \text{ si } x \notin \Delta(m, C) \\ \emptyset & , \text{ si } x \in \Delta(m, C) \end{cases}$$

Es decir, $\mu'$ es el árbol enraizado obtenido de $\mu$ eliminando los nodos que se disuelven por la aplicación de $m$, y restaurando las conexiones en la forma correcta. Si un nodo $y$ es disuelto, el nodo se elimina, junto con todas las aristas adyacentes a él, y se añaden aristas entre su primer antecesor no disuelto y los primeros descendientes de $y$ no disueltos. Si un nodo no se disuelve, debemos añadir a su contenido el de todos sus donantes por la aplicación de $m$.

A continuación, formalizamos la ejecución de *un paso* en un sistema celular; es decir, precisamos cómo se produce la transición de una configuración del sistema P a otra configuración tras ejecutar un paso. Se trata, pues, de precisar cómo calcula, cómo evoluciona un sistema P a lo largo del tiempo.

**Definición 16** *Se dice que una configuración $C_2$ de un sistema $P$ de transición, $\Pi$, deriva de una configuración $C_1$ por una* transición en un paso *de $\Pi$ (o tras la ejecución de un paso), y se nota por $C_1 \Rightarrow_\Pi C_2$, si existe un multiconjunto, $m$, de aplicabilidad sobre $C_1$ tal que $m \neq \emptyset$ y $m(C_1) = C_2$.*

Intuitivamente, $C_1 \Rightarrow_\Pi C_2$ significa que en un paso de computación del sistema $\Pi$, de la configuración $C_1$ resulta la configuración $C_2$.

Una vez definida la relación de *transición en un paso* entre configuraciones de un sistema celular, se define la relación de *transición* como su clausura transitiva.

**Definición 17** *Sean $C$ y $C'$ dos configuraciones de un sistema $\Pi$ y $k \in \mathbb{N}$. Diremos que $C'$ se obtiene de $C$ en $k$ pasos de transición (es decir, al ejecutarse $k$ pasos), si existen configuraciones $C_1, \ldots, C_{k+1}$ tales que*

$$C_1 = C \wedge C_{k+1} = C' \wedge \forall i\ (1 \leq i \leq k \rightarrow C_i \Rightarrow_\Pi C_{i+1})$$

*Se dice que $C'$ se obtiene de $C$ en una computación de $\Pi$, y se nota por $C \stackrel{\star}{\Rightarrow}_\Pi C'$, si existe $k \in \mathbb{N}$ tal que $C'$ se obtiene de $C$ en $k$ pasos de transición.*

A partir de la configuración inicial de un sistema celular se puede construir el árbol de computación asociado al mismo.

**Definición 18** *El* árbol de computación *de un sistema P de transición,* $\Pi$*, es el árbol enraizado etiquetado maximal definido como sigue: la raíz es la configuración inicial de* $\Pi$*; los hijos de un nodo son las configuraciones que se obtienen a partir de él por un paso de transición; los nodos y las aristas están etiquetadas por configuraciones y multiconjuntos de aplicabilidad, respectivamente, de tal forma que dos nodos etiquetados,* $C$ *y* $C'$*, son adyacentes por medio de una arista etiquetada por* $m$*, si y sólo si* $m \in \mathbf{M_{Ap}}(C) - \{\emptyset\} \wedge C' = m(C)$*. Las ramas maximales se denominan* computaciones *de* $\Pi$*. Diremos que una computación de* $\Pi$ *es de* parada *si es una rama finita. Las configuraciones* $C$ *que verifican* $\mathbf{M_{Ap}}(C) = \{\emptyset\}$ *se denominan* de parada*.*

Es decir, una computación, $\mathcal{C}$, de un sistema P de transición, $\Pi = (A, C_0, \mathcal{R}, i_0)$, es una sucesión (finita o no) de configuraciones del sistema $C_0 \Rightarrow_\Pi C_1 \Rightarrow_\Pi C_2 \Rightarrow_\Pi \cdots \Rightarrow_\Pi C_r$, con $r \in \mathbb{N} \cup \{\infty\}$. En tal situación, diremos que $r$ es la longitud de $\mathcal{C}$ y notaremos $|\mathcal{C}| = r$. Además, si $r \in \mathbb{N}$ entonces diremos que $C_r$ es la *configuración final* de la computación $\mathcal{C}$.

**Definición 19** *Sea* $\Pi$ *un sistema P de transición. Diremos que una computación de* $\Pi$*, es* exitosa *si* $i_0$ *es una hoja del árbol enraizado de la configuración final asociada (en este caso, diremos que dicha configuración es* exitosa*.*

Es decir, una computación es exitosa si es de parada y, además, la membrana de salida es elemental en la configuración final de dicha computación. Es claro que se puede construir sistemas celulares en los que exista una configuración que posea distintos multiconjuntos de aplicabilidad. En ese momento de la computación, el sistema tendrá varias posibilidades para continuar, apareciendo el no determinismo al que se ha hecho mención con anterioridad.

### 4.3. Sistemas celulares con entrada

**Definición 20** *Un sistema P de transición con entrada es una tupla* $(\Pi, \Sigma, i_\Pi)$*, tal que:*

- $\Pi$ *es un sistema P de transición, con alfabeto de trabajo* $\Gamma$*, con* $p$ *membranas etiquetadas por* $1, \ldots, p$*, y multiconjuntos iniciales* $\mathcal{M}_1, \ldots, \mathcal{M}_p$ *asociados a cada una de ellas.*
- $\Sigma$ *es un alfabeto (denominado de entrada) estrictamente contenido en* $\Gamma$*.*
- $\mathcal{M}_1, \ldots, \mathcal{M}_p$ *son multiconjuntos sobre* $\Gamma - \Sigma$*.*
- $i_\Pi$ *es la etiqueta de una membrana distinguida (membrana de entrada).*

Si $m$ es un multiconjunto sobre $\Sigma$, entonces la *configuración inicial de* $(\Pi, \Sigma, i_\Pi)$ *con entrada* $m$ es $(\mu, \mathcal{M}_1, ..., \mathcal{M}_{i_\Pi} + m, ..., \mathcal{M}_p)$.

Las computaciones en un sistema celular con entrada asociada a un cierto multiconjunto de entrada, se definen de manera análoga a como se hizo en el caso general, con la única diferencia de que la configuración inicial está asociada a ese multiconjunto de entrada.

### 4.4. Sistemas celulares con salida externa

Una primera clasificación de los sistemas celulares se puede realizar en función de dónde se recoge la salida de las computaciones. Los *sistemas celulares de transición con salida externa* se caracterizan porque la salida de las computaciones se recoge en el entorno de la estructura de membranas.

La membrana piel de una estructura de membranas (nos referiremos a ella usando la meta–etiqueta *skin*) aísla a la estructura de su entorno (al que nos referiremos con la meta–etiqueta *env*). A continuación vamos a asociar a cada estructura de membrana una estructura *con entorno*.

**Definición 21** *Sea* $\mu = (V(\mu), E(\mu))$ *una estructura de membranas. La* estructura de membranas con entorno *asociada a* $\mu$ *es el árbol enraizado* $Ext(\mu)$ *donde:* $V(Ext(\mu)) = V(\mu) \cup \{env\}$; $E(Ext(\mu)) = E(\mu) \cup \{\{env, skin\}\}$; *y la raíz del árbol es el nodo env, que se denomina* entorno *de la estructura* $\mu$.

Obsérvese que lo único que se hace es añadir un nuevo nodo que representa al entorno y que, por tanto, sólo es adyacente a la piel, mientras que la estructura de membranas original permanece inalterada.

**Definición 22** *Un* sistema P de transición con salida externa *es un par* $(\Pi, \mathcal{M}_e)$, *en donde* $\Pi$ *es un sistema P de transición y* $\mathcal{M}_e = \emptyset$ *es el multiconjunto vacío de objetos (que representa el contenido del entorno).*

A continuación vamos a detallar de qué forma evoluciona un sistema P de transición con salida externa atendiendo a los multiconjuntos de objetos contenidos en cada una de las membranas de su estructura de membranas, así como a las reglas de evolución asociadas a ellas.

**Definición 23** *Sea* $\Pi$ *un sistema P de transición con salida externa. Una configuración de* $\Pi$ *es un par* $C = \big(Ext(\mu), M\big)$ *tal que:*

- $\mu = (V(\mu), E(\mu))$ *es una estructura de membranas.*
- $Ext(\mu)$ *es la estructura de membranas con entorno asociada a* $\mu$.
- *El conjunto* $V(\mu)$ *de nodos de* $\mu$ *está incluido en el conjunto* $V(\mu_\Pi)$ *de nodos de* $\mu_\Pi$, *y contiene la raíz de* $\mu_\Pi$.
- *Las raíces de ambas estructuras de membranas coinciden.*
- $M$ *es una aplicación de dominio* $V(Ext(\mu))$ *y rango contenido en* $\mathbf{M}(\Gamma)$.

Notaremos por $C = (\mu, M_{env}, M_{i_1}, \ldots, M_{i_q})$ una configuración de $\Pi$, donde $V(\mu) = \{i_1, \ldots, i_q\}$, $M_{env} = M(env)$ es el multiconjunto asociado al entorno de $\mu$, y $M_{i_j} = M(i_j)$ es el multiconjunto asociado a la membrana $i_j$ de $\mu$, para cada $j = 1, \ldots, q$.

A la hora de definir las configuraciones que especifican el estado inicial de un sistema P (es decir, sus configuraciones iniciales) debemos tener en cuenta si dicho sistema posee o no membrana de entrada.

**Definición 24** *Sea* $(\Pi, \mathcal{M}_e)$ *un sistema P de transición con salida externa, cuyos multiconjuntos iniciales son* $\mathcal{M}_1, \ldots, \mathcal{M}_p$. *Si* $\Pi$ *no posee membrana de entrada, entonces existe una única configuración inicial del sistema, a saber* $C^0 = (\mu_\Pi, \emptyset, \mathcal{M}_1, \ldots, \mathcal{M}_p)$ *Si* $\Pi$ *posee membrana de entrada, entonces existe una configuración inicial para cada multiconjunto* $m \in \mathbf{M}(\Sigma)$, *a saber* $C^0(m) = (\mu_\Pi, \emptyset, \mathcal{M}_1, \ldots, \mathcal{M}_{im} + m, \ldots, \mathcal{M}_p)$

A partir de aquí, se define de manera usual el concepto de paso de transición y computación del sistema celular. Además, una de las características de estos sistemas radica en que el resultado de cualquier computación de parada se *recoge* en el entorno, de acuerdo con una cierta función de salida.

### 4.5. Sistemas celulares reconocedores

**Definición 25** *Un* sistema P de decisión *es un par* $(\Pi, F_\Pi)$ *tal que* $\Pi$ *es un sistema P y* $F_\Pi$ *es una función total booleana sobre el conjunto de las computaciones de parada de* $\Pi$. *Si* $\mathcal{C}$ *es una computación de parada y* $F_\Pi(\mathcal{C}) = 1$ *(respectivamente,* $F_\Pi(\mathcal{C}) = 0$*), entonces diremos que* $\mathcal{C}$ *es de aceptación (respectivamente, de rechazo).*

A continuación se introduce un caso particular de sistemas de decisión: aquellos que pueden ser considerados como dispositivos reconocedores de lenguajes.

**Definición 26** *Un* sistema P reconocedor *es un sistema P de transición con entrada y con salida externa tal que:*

- *El alfabeto de trabajo tiene dos elementos distinguidos: yes, no.*
- *Todas las computaciones del sistema son de parada.*
- *Si* $\mathcal{C}$ *es una computación del sistema, entonces o bien algún objeto yes o bien algún objeto no (pero no ambos) son enviados al entorno (y, únicamente, en el último paso de la computación).*

Sea $\Pi$ un sistema P reconocedor. Definimos la función *Output* sobre el conjunto de las computaciones de $\Pi$. Si $\mathcal{C}$ es una computación de $\Pi$ y $M_{env}$ es el multiconjunto asociado al entorno en la configuración de parada, entonces

$$Output(\mathcal{C}) = \begin{cases} yes & \text{, si } yes \in M_{env} \\ no & \text{, si } yes \notin M_{env} \end{cases}$$

Diremos que una computación $\mathcal{C}$ es de *aceptación* (respectivamente, de *rechazo*) si el objeto *yes* (respectivamente, *no*) aparece en el entorno asociado a la correspondiente configuración de parada de $\mathcal{C}$.

Notaremos por $\mathcal{LR}$ a la clase de los sistemas P reconocedores de lenguajes.

## 5. Una Teoría de la Complejidad en sistemas P

Siempre que se introduce un nuevo modelo de computación es necesario desarrollar una teoría formal de la Complejidad Computacional, a fin de cuantificar la cantidad de recursos necesarios para resolver problemas dentro del modelo, y para poder comparar esa cantidad con la que se necesita en otros modelos de computación. La primera teoría de la complejidad en sistemas celulares con membranas ha sido desarrollada en [22] y [23]

En lo que sigue, supondremos que $\mathcal{D}$ es una clase genérica de sistemas P de decisión, $g : \mathbb{N}^+ \to \mathbb{N}^+$ una función total computable y $X = (I_X, \theta_X)$ un problema de decisión (en donde $\mathbb{N}^+$ es el conjunto $\mathbb{N} - \{0\}$, $I_X$ es un lenguaje sobre un alfabeto finito, el conjunto de *instancias* de $X$, y $\theta_X$ es un predicado sobre dicho conjunto).

### 5.1. Clases de complejidad en sistemas P

Los primeros resultados sobre resolución eficiente de problemas **NP**-completos por medio de sistemas de computación celular con membranas se han obtenido usando variantes de sistemas celulares que carecían de membrana de entrada. Por ello, en las pruebas de dichos resultados para *cada instancia* del problema, se construye un sistema que la decide en tiempo polinomial.

**Definición 27** *Diremos que un problema de decisión, $X$, pertenece a la clase $MC^f_{\mathcal{D}}(g)$ si existe una familia de sistemas celulares, $\mathbf{\Pi} = \left(\Pi(w)\right)_{w \in I_X}$, verificando las siguientes propiedades:*

1. *La familia $\mathbf{\Pi}$ es consistente, respecto de la clase $\mathcal{D}$; es decir, todo sistema de la familia pertenece a la clase $\mathcal{D}$.*

2. *Para cada $w \in I_X$, el sistema $\Pi(w)$ carece de membrana de entrada.*

3. *La familia $\mathbf{\Pi}$ es polinomialmente uniforme, por máquinas de Turing; es decir, existe una tal máquina que, dado $w \in I_X$, construye el sistema $\Pi(w)$ en tiempo polinomial en el tamaño de $w$.*

4. *La familia $\mathbf{\Pi}$ está acotada, respecto del problema $X$ y la función $g$; es decir, para cada $w \in I_X$ se tiene que toda computación del sistema $\Pi(w)$ es de parada y, además, realiza a lo sumo $g(|w|)$ pasos.*

5. *La familia $\mathbf{\Pi}$ es adecuada, respecto del problema $X$; es decir, para cada $w \in I_X$ se tiene que si existe una computación del sistema $\Pi(w)$ que es de aceptación, entonces $\theta_X(w) = 1$.*

6. *La familia $\mathbf{\Pi}$ es completa, respecto del problema $X$; es decir, para cada $w \in I_X$ se tiene que si $\theta_X(w) = 1$, entonces toda computación del sistema $\Pi(w)$ es de aceptación.*

Si $X \in MC_{\mathcal{D}}^{f}(g)$, entonces diremos que el problema $X$ es resoluble por una familia de sistemas celulares sin membrana de entrada, pertenecientes a $\mathcal{D}$, en tiempo acotado por $g$.

Si en la definición anterior consideramos como cotas los polinomios, se obtiene la clase de complejidad polinomial para familias de sistemas P sin entrada.

**Definición 28** *La clase de problemas de decisión resolubles en tiempo polinomial por una familia de sistemas celulares sin membrana de entrada, pertenecientes a la clase $\mathcal{D}$, es la clase*

$$\mathbf{PMC}_{\mathcal{D}}^{f} = \bigcup_{k>0} MC_{\mathcal{D}}^{f}(n^k)$$

Esta clase es cerrada bajo reducibilidad en tiempo polinomial [24].

**Proposición 1** *Sean $X$ e $Y$ problemas de decisión tales que $X$ es reducible a $Y$ en tiempo polinomial. Si $Y \in \mathbf{PMC}_{\mathcal{D}}^{f}$, entonces $X \in \mathbf{PMC}_{\mathcal{D}}^{f}$.*

Además, se tiene que la clase $\mathbf{PMC}_{\mathcal{D}}^{f}$ es cerrada bajo complementario.

Por otra parte, si trabajamos con sistemas con membrana de entrada podemos considerar familias tales que cada uno de los sistemas que pertenezcan a ellas decida instancias de *tamaño equivalente.*

**Definición 29** *Diremos que un problema de decisión $X$ pertenece a la clase $MC_{\mathcal{D}}^{fi}(g)$ si existe una familia de sistemas celulares, $\mathbf{\Pi} = (\Pi(n))_{n\in\mathbb{N}^+}$, verificando las siguientes propiedades:*

1. *La familia $\mathbf{\Pi}$ es consistente, respecto de la clase $\mathcal{D}$.*
2. *Para cada $n \in \mathbb{N}^+$, el sistema $\Pi(n)$ tiene membrana de entrada.*
3. *La familia $\mathbf{\Pi}$ es polinomialmente uniforme, por máquinas de Turing.*
4. *Existen dos funciones, $cod : I_X \to \bigcup_{n\in\mathbb{N}^+} I_{\Pi(n)}$ y $s : I_X \to \mathbb{N}^+$, computables en tiempo polinomial, tales que:*
   - *Para todo $w \in I_X$, $cod(w) \in I_{\Pi(s(w))}$.*
   - *La familia $\mathbf{\Pi}$ está acotada, respecto del problema $X$, la función cod, la función $s$ y la función $g$; es decir, para cada $w \in I_X$ se tiene que toda computación del sistema $\Pi(s(w))$ con entrada $cod(w)$ es de parada y, además, realiza a lo sumo $g(|w|)$ pasos.*
   - *La familia $\mathbf{\Pi}$ es adecuada, respecto del problema $X$, la función cod y la función $s$; es decir, para cada $w \in I_X$ se tiene que si existe una computación del sistema $\Pi(s(w))$ con entrada $cod(w)$ que es de aceptación, entonces $\theta_X(w) = 1$.*
   - *La familia $\mathbf{\Pi}$ es completa, respecto del problema $X$, la función cod y la función $s$; es decir, para cada $w \in I_X$ se tiene que si $\theta_X(w) = 1$, entonces toda computación del sistema $\Pi(s(w))$ con entrada $cod(w)$ es de aceptación.*

Si $X \in MC_{\mathcal{D}}^{fi}(g)$, entonces diremos que el problema $X$ es resoluble por una familia de sistemas celulares con membrana de entrada, pertenecientes a la clase $\mathcal{D}$, en tiempo acotado por $g$.

Por otra parte, en las condiciones de la definición anterior diremos que el par $(cod, s)$ es una *codificación polinomial* de $X$ en $\mathbf{\Pi}$.

**Definición 30** *La clase de problemas de decisión resolubles en tiempo polinomial por una familia de sistemas de computación celular con membrana de entrada, pertenecientes a la clase $\mathcal{D}$, es la clase*

$$\mathbf{PMC}_{\mathcal{D}}^{fi} = \bigcup_{k>0} MC_{\mathcal{D}}^{fi}(n^k)$$

Esta clase es cerrada bajo reducibilidad en tiempo polinomial[24].

**Proposición 2** *Sean $X$ e $Y$ problemas de decisión tales que $X$ es reducible a $Y$ en tiempo polinomial. Si $Y \in \mathbf{PMC}_{\mathcal{D}}^{fi}$, entonces $X \in \mathbf{PMC}_{\mathcal{D}}^{fi}$.*

Además, se tiene que la clase $\mathbf{PMC}_{\mathcal{D}}^{fi}$ es cerrada bajo complementario.

Obsérvese que en las definiciones anteriores suponemos una *confluencia* de los sistemas de la familia; es decir, los sistemas proporcionan la misma respuesta para todas las computaciones que partan de la misma configuración inicial.

A continuación vamos a caracterizar la relación $\mathbf{P} \neq \mathbf{NP}$ mediante la irresolubilidad en tiempo polinomial de problemas **NP**–completos a través de sistemas P de transición.

### 5.2. Simulación de máquinas de Turing por sistemas celulares

Comenzamos observando que es posible asociar un problema de decisión a cada máquina de Turing, lo que nos permitirá establecer con precisión qué significa que una tal máquina sea simulada por sistemas celulares.

**Definición 31** *Sea $TM$ una máquina de Turing, como dispositivo de decisión de lenguajes, y con alfabeto de entrada $\Sigma_{TM}$. El* problema de decisión asociado a $TM$ *es el problema $X_{TM} = (I_{TM}, \theta_{TM})$, siendo $I_{TM} = \Sigma_{TM}^*$; y para cada cadena $w \in \Sigma_{TM}^*$, $\theta_{TM}(w) = 1$ si y solo si $TM$ acepta $w$.*

Diremos que la máquina $TM$ es simulada en tiempo polinomial por una familia de sistemas en la clase $\mathcal{D}$ si $X_{TM} \in \mathbf{PMC}_{\mathcal{D}}^{fi}$ o $X_{TM} \in \mathbf{PMC}_{\mathcal{D}}^{f}$, según si los sistemas poseen o no membrana de entrada.

Consideramos las máquinas de Turing como dispositivos de decisión de lenguajes; es decir, dichas máquinas paran sobre cualquier dato de entrada, siendo el estado de parada igual al de aceptación, en el caso en que la cadena pertenezca al lenguaje decidido, o bien igual al estado de rechazo, en el caso en que la cadena no pertenezca a ese lenguaje.

**Proposición 3** *Sea $TM$ una máquina de Turing determinista que trabaja en tiempo polinomial. Entonces $X_{TM} \in \mathbf{PMC}_{\mathcal{LR}}^{fi}$.*

La prueba del resultado consiste en asociar a la máquina de Turing $TM$ una familia de sistemas celulares de tal forma que cada uno de ellos decida las cadenas de entrada de un tamaño determinado. En la figura 3 se proporciona un diagrama de flujo que muestra el funcionamiento de estos sistemas una vez recibida la cadena de entrada, codificada como un multiconjunto de objetos del sistema (para detalles de la prueba, véase [24]).

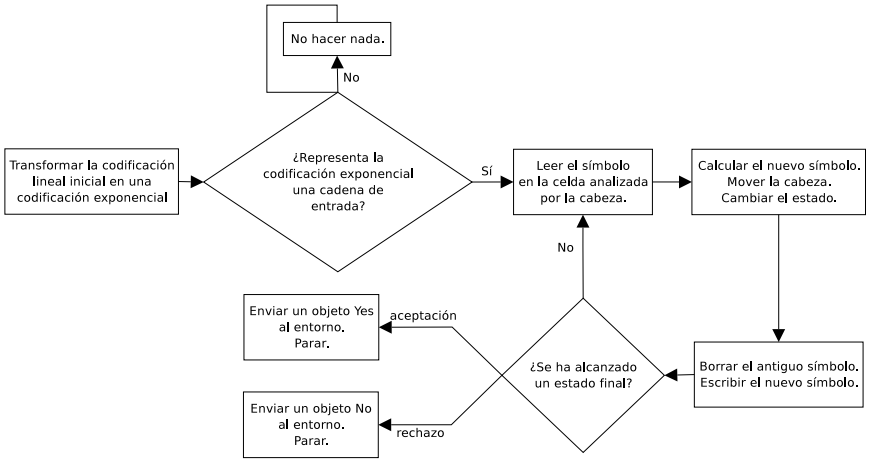


Figura 3: Simulación de máquinas de Turing por sistemas celulares

Seguidamente, se establece una condición *suficiente* para que se verifique la relación $\mathbf{P} \neq \mathbf{NP}$, en términos de irresolubilidad en tiempo polinomial de problemas **NP**–completos por sistemas P de transición.

**Proposición 4** *Si existe un problema* **NP***–completo irresoluble en tiempo polinomial por una familia de sistemas celulares reconocedores, entonces* $\mathbf{P} \neq \mathbf{NP}$.

*Demostración*. Sea $X$ un problema **NP**–completo tal que $X \notin \mathbf{PMC}^{fi}_{\mathcal{LR}}$. Supongamos que $\mathbf{P} = \mathbf{NP}$. Entonces $X \in \mathbf{P}$. Luego existe una máquina de Turing determinista, $TM$, que resuelve el problema $X$ en tiempo polinomial.

Por la proposición 3, $X_{TM} \in \mathbf{PMC}^{fi}_{\mathcal{LR}}$. Por tanto, existe una familia $\mathbf{\Pi}_{TM} = \left(\Pi_{TM}(k)\right)_{k\in\mathbb{N}^+}$ de sistemas reconocedores de lenguajes que simula $TM$ en tiempo polinomial.

Consideremos la función $cod_X : I_X \to \bigcup_{k\in\mathbb{N}^+} I_{\Pi_{TM}(k)}$, dada por $cod_X(w) = cod_{TM}(w)$, y la función $s_X : I_X \to \mathbb{N}^+$, dada por $s_X(w) = |w|$. Entonces se prueba que la familia $\mathbf{\Pi}_{TM}$ es adecuada y completa, respecto de $X$, la función $cod_X$ y la función $s_X$.

En consecuencia, se tendría que $X \in \mathbf{PMC}^{fi}_{\mathcal{LR}}$. Lo que es una contradicción. □

### 5.3. Simulación de sistemas celulares por máquinas de Turing

A continuación vamos a probar que si un problema de decisión se puede resolver en tiempo polinomial por una familia de sistemas reconocedores, también se puede resolver en tiempo polinomial por una máquina de Turing determinista.

Para el diseño de la máquina de Turing nos hemos inspirado en un trabajo de C. Zandron, C. Ferretti y G. Mauri [26], aunque dicho artículo se enmarca dentro del estudio de los sistemas celulares que usan membranas activas.

**Proposición 5** *Si* $X \in \mathbf{PMC}_{\mathcal{LR}}^{fi}$, *entonces existe una máquina de Turing determinista que resuelve el problema* $X$ *en tiempo polinomial.*

*Demostración*. Sea $\mathbf{\Pi} = \left(\Pi(n)\right)_{n\in\mathbb{N}^+}$ una familia de sistemas de la clase $\mathcal{LR}$ que es polinomialmente uniforme por máquinas de Turing; y para la que existe una codificación polinomial $(cod, s)$ de $X$ en $\mathbf{\Pi}$ tal que dicha familia está polinomialmente acotada, es adecuada y completa respecto de $X$, la función $cod$ y la función $s$.

A continuación asociamos al sistema $\Pi(n)$ una máquina de Turing determinista, $TM(n)$, *con múltiples cintas*, tal que, dado un multiconjunto de entrada, $m \in I_{\Pi(n)}$, la máquina reproduce *una* determinada computación, fijada por el proceso de funcionamiento de la máquina, de $\Pi(n)$ sobre $m$.

El alfabeto de entrada de la máquina $TM(n)$ coincide con el del sistema $\Pi(n)$. El alfabeto de trabajo contiene, además, un símbolo por cada etiqueta asignada a las membranas de dicho sistema; los símbolos 0 y 1, que permitirán operar con números representados en base 2; tres símbolos que indicarán si una membrana no se ha disuelto, ha de disolverse o se ha disuelto; y tres símbolos que indicarán si una regla está en espera, es aplicable o no es aplicable.

El proceso de simulación, como veremos, se puede estructurar en varias fases realizan distintos bucles en los que se recorren las membranas de la estructura de membranas original del sistema y las reglas asociadas a ellas. Así, el conjunto de estados de la máquina de Turing contendrá estados relacionados con dichas membranas y reglas para cada una de las fases. Estos estados determinarán en qué momento de la simulación nos encontramos en cada paso del funcionamiento de la máquina.

Seguidamente, detallamos las cintas de las que consta esta máquina.

- Una *cinta de entrada*, que guarda una cadena representando el multiconjunto de entrada recibido.
- Por cada membrana del sistema tenemos:
  - Una *cinta de estructura*, que guarda en la segunda celda la etiqueta del padre de la membrana, y en la tercera uno de los tres símbolos que indican si la membrana no se ha disuelto, si la membrana ha de disolverse, o si la membrana se ha disuelto.
  - Para cada objeto del alfabeto de trabajo del sistema:
    - Una *cinta principal*, que guarda la multiplicidad del objeto, en base 2, en el multiconjunto contenido en la membrana.

  - Una *cinta auxiliar*, que guarda resultados temporales, también en base 2, de aplicar las reglas asociadas a la membrana.

  Tras la simulación de cada paso del sistema P, el contenido de estas dos cintas coincidirá.
- Una *cinta de reglas*, en la que cada celda a partir de la segunda corresponde a una regla asociada a la membrana (suponemos que el conjunto de dichas reglas está ordenado), y guarda uno de los tres símbolos que indican si dicha regla está en espera, si es aplicable, o si no es aplicable.

- Para cada objeto del alfabeto de salida tenemos:
  - Una *cinta de entorno*, que guarda la multiplicidad del objeto, en base 2, en el multiconjunto asociado al entorno externo.

A continuación describimos sucintamente los pasos realizados por la máquina $\Pi(n)$ para simular un paso de transición del sistema celular. Los algoritmos que implementan cada uno de los pasos pueden encontrarse en [24].

- *Inicialización del sistema.* En la primera fase del proceso de simulación seguido por la máquina de Turing se colocan en las cintas correspondientes los símbolos necesarios para que quede reflejada la configuración inicial de la computación con entrada el multiconjunto $m$ que se va a simular.

- *Determinar las reglas aplicables.* Para simular un paso del sistema celular, lo primero que debe hacer la máquina de Turing es determinar el conjunto de reglas que son aplicables (cada una de ellas de forma independiente) a la configuración considerada en las membranas a las que están asociadas.

- *Aplicación de las reglas.* Una vez determinadas las reglas aplicables, éstas se aplican de forma maximal a las membranas a las que están asociadas. El hecho de que las reglas se consideren en un cierto orden (usando maximalidad local para cada regla, según dicho orden) determina un multiconjunto de reglas aplicable particular, fijando así *la* computación del sistema que simula la máquina de Turing. No obstante, de la definición de clases de complejidad que hemos realizado se deducirá que la computación elegida no es relevante para la prueba del resultado que pretendemos demostrar, debido a la *confluencia* del sistema.

- *Actualización de los multiconjuntos.* Una vez aplicadas las reglas a las membranas, las cintas auxiliares de éstas guardan los resultados obtenidos, por lo que dichos resultados deben ser trasladados a las cintas principales correspondientes.

- *Disolución de las membranas.* Para terminar la simulación de un paso de la computación del sistema P es necesario disolver las membranas según indiquen las reglas que se han aplicado en la fase anterior y reestructurar el árbol de membranas, redefiniendo para ello la función padre de dicho árbol.

- *Comprobar si la simulación ha terminado.* Finalmente, tras terminar la simulación de un paso de la computación del sistema $\Pi(n)$, la máquina de Turing debe comprobar si dicha computación ha llegado a una configuración de parada y, en su caso, si la computación es de aceptación o de rechazo.

Finalmente, consideremos la siguiente máquina de Turing determinista, $TM_{\mathbf{\Pi}}$:

**Entrada:** $w \in I_X$

– Calcular $s(w)$

– Construir $TM(s(w))$

– Calcular $cod(w)$

– Reproducir el funcionamiento de $TM(s(w))$ con dato de entrada $cod(w)$

Entonces, la máquina $TM_{\mathbf{\Pi}}$ resuelve el problema $X$ en tiempo polinomial. □

Seguidamente vamos a establecer una condición *necesaria* para que se verifique la relación $\mathbf{P} \neq \mathbf{NP}$ a través de sistemas reconocedores.

**Proposición 6** *Si* $\mathbf{P} \neq \mathbf{NP}$*, entonces ningún problema* **NP***–completo puede ser resuelto en tiempo polinomial por una familia de sistemas celulares reconocedores.*

*Demostración*. Supongamos que existe un problema **NP**–completo, $X$, tal que $X \in \mathbf{PMC}^{fi}_{\mathcal{LR}}$. Entonces, por la proposición 5 existe una máquina de Turing que resuelve el problema $X$ en tiempo polinomial. Por tanto, $X \in \mathbf{P}$. Luego $\mathbf{P} = \mathbf{NP}$, lo que lleva a una contradicción. □

De las proposiciones 4 y 6, se deduce la siguiente caracterización de la relación $\mathbf{P} \neq \mathbf{NP}$ a través de la irresolubilidad de problemas **NP**–completos por familias de sistemas P reconocedores.

**Teorema 7** *Las proposiciones siguientes son equivalentes:*

1. $\mathbf{P} \neq \mathbf{NP}$.
2. $\exists X \big(X$ *es un problema de decisión* **NP***–completo* $\wedge\ X \notin \mathbf{PMC}^{fi}_{\mathcal{LR}}\big)$.
3. $\forall X \big(X$ *problema de decisión* **NP***–completo* $\rightarrow X \notin \mathbf{PMC}^{fi}_{\mathcal{LR}}\big)$.

Finalmente, vamos a establecer una descripción de la clase **P** de los problemas *tratables* en modo determinista, a través de la clase de complejidad polinomial relativa a sistemas celulares reconocedores de lenguajes.

**Teorema 8** *Se verifica que* $\mathbf{P} = \mathbf{PMC}^{fi}_{\mathcal{LR}}$.

*Demostración*. Si $X \in \mathbf{P}$, entonces de la proposición 3, se deduce que $X \in \mathbf{PMC}^{fi}_{\mathcal{LR}}$.

Por otra parte, si $X$ es un problema de decisión resoluble en tiempo polinomial por una familia de sistemas P reconocedores de lenguajes, entonces de la proposición 5, resulta que $X \in \mathbf{P}$. □

## Referencias

[1] L.M. Adleman. Molecular Computations of Solutions to Combinatorial Problems. *Science*, 226, (1994), 1021–1024.

[2] A.V. Baranda, J. Castellanos, F. Arroyo, R. Gonzalo. Towards an electronic implementation of membrane computing: A formal description of nondeterministic evolution in transition P systems. En *Proceedings of the 7th International Meeting on DNA Based Computers* (N. Jonoska, N.C. Seeman, eds.), Tampa, Florida, USA, (2001), 273–282.

[3] G. Berry, G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96, (1992), 217–248.

[4] A.W. Burks, H.H. Goldstine, J. von Neumann. Preliminary discussion of the logical design of an electronic computing instrument. En *Collected Works of John von Neumann* (A. H. Taub, ed.), vol. 5, 34–79, The Macmillan Company, New York, 1963. Taken from report to U. S. Army Ordnance Department, 1946.

[5] S. Cook. The P versus NP Problem. *Manuscript prepared for the Clay Mathematics Institute for the Millennium Prize Problems* (revised November, 2000).

[6] T.J. Head. Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviours. *Bulletin of Mathematical Biology*, 49, (1987), 737–759.

[7] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[8] M. Ito, C. Martín-Vide, Gh. Păun. A characterization of Parikh sets of ETOL languages in terms of P systems. En *Words, Semigroups, and Transductions* (M. Ito, G. Paun, S. Yu, eds.), World Scientific, Singapore, (2001), 239-253.

[9] S.N. Krishna, R. Rama. On the power of P systems with sequential and parallel rewriting. *International Journal of Computer Mathematics*, 77, 1-2 (2000), 1–14.

[10] S.N. Krishna, R. Rama. P systems with replicated rewriting. *Journal of Automata, Languages and Combinatorics*, 6, 3 (2001), 345–350.

[11] W.S. McCulloch, W. Pitts. A logical calculus of the ideas inmanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, (1943), 115–133.

[12] A. Obtulowicz. Membrane computing and one-way functions. *International Journal of Foundations of Computer Science*, 12, 4 (2001), 551–558.

[13] A. Păun. On P systems with membrane division. En *Unconventional Models of Computation* (I. Antoniou, C.S. Calude, M.J. Dinnen, eds.), Springer-Verlag, London (2000), 187–201.

[14] A. Păun, Gh. Păun. The power of communication: P systems with Symporters-Antiporters. *New Generation Computing*, 20, 3 (2002).

[15] Gh. Păun. Computing with membranes. *Journal of Computer and System Sciences*, 61, 1 (2000), 108–143, and *Turku Center for Computer Science-TUCS Report* No 208 (1998) (www.tucs.fi).

[16] Gh. Păun. Further research topics about P systems. *Pre-Proceedings of Workshop on Membrane Computing*, Curtea de Arges, Rumania, Agosto 2001, Technical Report 17/01 of Research Group on Mathematical Mathematical Linguistics, Rovira i Virgili University, Tarragona, España, (2001), 243–250.

[17] Gh. Păun. Computing with membranes. A variant. *International Journal of Foundations of Computer Sciences*, 11, 1 (2000), 167–182.

[18] Gh. Păun. *Membrane Computing. An introduction*, Springer–Verlag, Berlin, 2002.

[19] Gh. Păun, G. Rozenberg. A guide to membrane computing. *Theoretical Computer Science*, 287 (2002), 73–100.

[20] M.J. Pérez Jiménez, F. Sancho Caparrini. A formalization of transition P systems. *Fundamenta Informaticae*, 49 (2002), 261–272.

[21] M.J. Pérez Jiménez, F. Sancho Caparrini. Verifying a P system generating squares. *Romanian Journal of Information Science and Technology*, 5, 2–3 (2002), 181–191.

[22] M.J. Pérez Jiménez, A. Romero Jiménez, F. Sancho Caparrini. *Teoría de la Complejidad en modelos de computación celular con membranas*. Ed. Kronos, 2003.

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

[24] A. Romero Jiménez. *Complejidad y universalidad en modelos de computación celular*. Tesis doctoral. Universidad de Sevilla, 2003.

[25] C. Martín-Vide, V. Mitrana, Gh. Păun. On the power of P systems with valuations. *Computación y sistemas*, 5, 2, 120-127.

[26] C. Zandron, G. Mauri, C. Ferreti. Solving NP-complete problems using P systems with active membranes. En *Unconventional Models of Computation* (I. Antoniou, C.S. Calude, M.J. Dinnen, eds.), Springer-Verlag, London, (2000), 289–301.