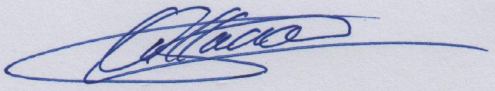
Universidad de Sevilla
Dpto. de Ciencias de la Computación
e Inteligencia Artificial

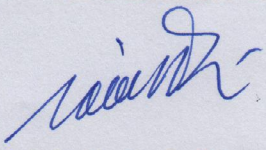
El problema P versus NP. Desarrollo de nuevas técnicas a través de modelos de computación bio-inspirados

Memoria presentada por
David Orellana Martín
para optar al grado de Doctor
por la Universidad de Sevilla

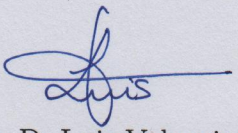


David Orellana Martín

V.º B.º Los Directores de la Tesis



Dr. D. Mario de J. Pérez Jiménez



Dr. D. Luis Valencia Cabrera

Universidad de Sevilla
Dpto. de Ciencias de la Computación
e Inteligencia Artificial

El problema P versus NP. Desarrollo de nuevas técnicas a través de modelos de computación bio-inspirados

Memoria presentada por
David Orellana Martín
para optar al grado de Doctor
por la Universidad de Sevilla

David Orellana Martín

V.º B.º Los Directores de la Tesis

Dr. D. Mario de J. Pérez Jiménez

Dr. D. Luis Valencia Cabrera

A mis padres

A Ari

Agradecimientos

De nuevo, como la última vez, vuelvo a leer los agradecimientos, tanto de mi Proyecto de Fin de Carrera como de mi Trabajo de Fin de Máster, y me doy cuenta que hay una invariante: siempre he tenido a gente importante en mi vida. Y lo mejor, es que ese número siempre va a más.

Hace poco me dijeron algo que me pareció curioso... «Me siento parte de tus éxitos, aunque sé que no debería». La pregunta subsiguiente es clara: ¿Por qué no? Todo el que ha pasado por mi vida, ha dejado algo importante en mí, ha hecho que sea como soy. Por todo eso he llegado hasta aquí. Es por ello que, obviamente, no podré nombrar a todos y a cada una de las personas a las que les tendría que agradecer algo. Pero sí que me gustaría nombrar a algunas de ellas.

Y por supuesto, estaré eternamente agradecido a Mario, cuya guía, no solo a nivel investigador, sino personal, ha sido decisiva para la realización de esta Tesis. Su trabajo desinteresado y su pasión docente, investigadora y, en general, por la Vida, ha sido una inspiración, y un gran ejemplo a seguir. Llegue donde llegue, le deberé gran parte a él.

También, tengo que agradecer a Luis por su constante apoyo, a través de unas sabias palabras o unas risas. Poder tenerlo como codirector, compañero pero sobre todo, como Amigo, es algo que no todo el mundo tiene la suerte de poder decir.

Está claro que tengo que agradecer a todo el Grupo de Investigación en Computación Natural por apoyarme y, sobre todo, aguantarme. Como ya dije en su momento, no te das cuenta del trabajo que tiene un Profesor hasta que lo conoces personalmente. Más precisamente, gracias a Agustín por su apoyo constante y su paciencia. A Miguel Ángel por poder compartir aficiones y pensamientos. A Álvaro por haberme ayudado a centrarme en algunos puntos importantes en el desarrollo de mi investigación, y a Carmen, por haberme ayudado a integrarme tan bien en el grupo (vale, y por haber aguantado MUCHO mis chistes malos). A Dani Cámpora, que aunque desde la distancia, se nota su presencia. A Ignacio, Fernando, Andrés y al resto del módulo H, por haber sonreído cada mañana. Al resto del Departamento de Ciencias de la Computación e Inteligencia Artificial por haber, de una manera u otra, influido en el desarrollo de esta Tesis.

A mis amigos, en general, por poder contar con ellos cuando lo he necesitado. A todas las personas que he conocido gracias a mis viajes.

A mi Madre, por ser mi mejor Maestra en la Vida. A mi Padre, por enseñarme el significado de la palabra Constancia. A mis hermanas y mis cuñados, por ser como son y estar para lo que haga falta sin pedir nada a cambio. Por supuesto, a Víctor, María e Inma, son tres grandes razones para seguir avanzando, para dejar el mundo mejor de como lo encontramos. A mi familia, en general, por apoyarme en cada uno de mis pasos. A mi segunda familia por hacerme sentir como en casa.

Y, por supuesto, a Ari. Gracias por estar ahí, por ser quien eres. Porque seguro que nada habría sido igual si no te hubiera conocido. Porque contigo, todo es especial. Por muchos más años, y que esta Aventura (juntos) no acabe nunca.

Índice general

Índice de figuras

Introducción

La resolución mecánica de problemas abstractos se estructura en una serie de *tareas básicas* que, para cada dato de entrada del problema, la ejecución de dichas tareas, en un determinado orden, proporcionará la solución correcta del problema asociada a esa entrada. Para materializar, implementar o llevar a cabo una solución mecánica de un problema abstracto se requiere el uso de una serie de recursos computacionales (expresados en términos de tiempo, espacio y otra medida de complejidad) que son necesarios para poder obtener la solución correspondiente a cualquier dato de entrada. En ese contexto, fijada una medida de complejidad, cada solución mecánica de un problema abstracto tiene asociada una función que determina la cantidad de recursos de esa medida que se necesita para implementar la solución. Obviamente, la función tendrá como variable independiente el tamaño del dato de entrada del problema.

Informalmente, se dirá que un problema abstracto es *tratable*, respecto de una cierta medida de complejidad, si existe una solución mecánica del mismo tal que la cantidad de recursos computacionales (expresados en esa medida) necesarios para hallar la solución correspondiente a un dato de entrada, se puede expresar a través de un polinomio que depende del tamaño de la entrada. Con la formalización del concepto de *procedimiento mecánico* que da lugar a los *modelos de computación*, la tratabilidad de problemas abstractos suele tomar como marco de referencia para expresar las soluciones mecánicas, un *modelo de computación universal*; es decir, un modelo de computación con la capacidad de poder resolver cualquier problema que, informalmente, tenga alguna solución mecánica. En particular, las máquinas de Turing deterministas es el ejemplo por antonomasia de modelo de computación universal.

El problema **P** *versus* **NP** consiste, simplemente, en determinar si son iguales dos clases de complejidad computacional: la clase **P** de los problemas *tratables* por máquinas de Turing deterministas, y la clase **NP** de los problemas *tratables* por máquinas de Turing no deterministas. Este problema es, sin lugar

a dudas, el más importante de la disciplina de *Ciencias de la Computación* y uno de los más relevantes que tiene planteada la Ciencia, en general. Precisamente el *Clay Mathematics Institute* (CMI) de Cambridge, Massachusetts (USA), lo designó como uno de los problemas del nuevo milenio y ofrece una dotación de un millón de dólares para quien proporcione una demostración de que las clases **P** y **NP** son distintas. Curiosamente, si alguien probara que esas clases son iguales entonces se produciría una auténtica revolución en la sociedad, a casi todos los niveles, y de tal magnitud que la «recompensa» de un millón de dólares (que la citada institución no estaría obligada a abonar) sería simple «calderilla», debido a que el autor dispondría de mecanismos para demostrar los restantes problemas del milenio.

Este trabajo gira en torno al problema **P** *versus* **NP** y trata de aportar una nueva visión del mismo desde una perspectiva absolutamente novedosa, basada en el paradigma de la *Computación Celular con Membranas* (*Membrane Computing*). Concretamente, los objetivos principales de esta memoria consiste, por una parte, en desarrollar una nueva metodología para abordar el citado problema y, por otra, en implementar esa metodología a través de la obtención de fronteras de la tratabilidad de problemas abstractos, expresadas en términos de ingredientes sintácticos y/o semánticos del citado paradigma. Es importante resaltar que cada una de estas fronteras proporciona una herramienta para poder resolver el problema **P** *versus* **NP**.

Contenido de la memoria

La presente memoria se ha estructurado en tres partes principales que consta, además de este capítulo a modo de prólogo, de otro capítulo a modo de epílogo, en el que se presentan las conclusiones más relevantes del trabajo desarrollado, así como la descripción de algunas líneas interesantes de investigación de cara al futuro. Las partes que vertebran este trabajo constan, a su vez, de nueve capítulos cuyos contenidos se describen a continuación de manera sucinta.

Parte I: Estado del arte

La primera parte de la memoria se organiza a través de una serie de capítulos que tratan de sentar las bases sobre las que se sustenta el cuerpo de la tesis, así como los resultados más relevantes que se han obtenido hasta la fecha.

La intención de que este trabajo sea, en cierto sentido, autocontenido ha propiciado la elaboración de un primer capítulo dedicado exclusivamente a la descripción de conceptos y resultados básicos que son necesarios para una mejor comprensión del texto. En el **Capítulo 1** se detallan los primeros elementos y notaciones relativos a lenguajes formales y multiconjuntos, se analizan algunas nociones de teoría de grafos y de lógica proposicional para, finalmente, dedicar unas secciones a la formalización de los problemas abstractos y la introducción de tres problemas de decisión que van a ser objeto de estudio en esta memoria: el problema de la satisfacibilidad (`SAT`), el problema de la accesibilidad (`Reachability`) y el problema del ciclo Hamiltoniano (`HAM-CYCLE`). También se ha considerado pertinente dedicar una sección a la función de Cantor, usada para la descripción de familias de sistemas de membranas que resuelven problemas de decisión, así como otra sección dedicada al análisis de los elementos de una matriz a partir de un determinado «recorrido».

El **Capítulo 2** está dedicado a presentar los conceptos de la *teoría de la computabilidad* y la *teoría de la complejidad computacional* que son necesarios para vislumbrar el verdadero significado y la trascendencia del problema **P** *versus* **NP**. Para ello, se describen brevemente algunos hechos importantes relacionados con la génesis y posterior desarrollo de la teoría de la computabilidad cuyo nacimiento surge, propiamente, con la introducción de los primeros modelos de computación en la década de los treinta del pasado siglo. El objetivo principal de dicha teoría consiste en clasificar los problemas abstractos de acuerdo con su resolubilidad a través de procedimientos mecánicos de un modelo de computación universal (*decidibilidad versus indecibilidad*). La aparición de las primeras máquinas de propósito general, entre 1943 y 1946, propició el nacimiento de la teoría de la complejidad computacional, cuyo objetivo principal consiste en clasificar los problemas abstractos de acuerdo con su resolubilidad, mediante máquinas reales de propósito general, para instancias del problema suficientemente grandes (*tratabilidad versus intratabilidad*). En cierto sentido, la teoría de la complejidad computacional puede ser considerada como una una especia de teoría de la computabilidad «práctica».

El **Capítulo 3** centra la atención en el problema **P** *versus* **NP**, contextualizando su formalización y describiendo la idea informal que en él subyace. Para justificar su relevancia, se analizan las posibles repercusiones que una solución del mismo tendría sobre la vida cotidiana, especialmente desde el punto de vista económico. Así mismo, se presenta una nueva metodología para abordar el citado problema que está basada en la búsqueda de fronteras entre la *no eficiencia* de modelos de computación y la *presumible eficiencia* de los mismos, de

tal manera que esas fronteras puedan ser expresadas en términos de ingredientes sintácticos y/o semánticos de éstos, dentro de un paradigma computacional.

En el **Capítulo 4** se introducen los conceptos generales relativos al paradigma de computación celular con membranas, incluidos los pilares fundamentales de una teoría de la complejidad computacional en el paradigma citado. Además, se presentan los modelos que trabajan «a modo de células» (*cell-like*) y otros que trabajan «a modo de tejidos» (*tissue-like*), que van a ser objeto de estudio en esta memoria. Entre los del primer tipo se introducirán: (a) los sistemas P básicos de transición; (b) los sistemas P con membranas activas (unos con cargas eléctricas y otros que carecen de polarizaciones); y (c) los sistemas P con reglas symport/antiport. Entre los del segundo tipo se introducirán: (a) los sistemas P básicos de tejidos; (b) los sistemas P de tejidos con división celular; (c) los sistemas P de tejidos con separación celular; y (d) los sistemas P de tejidos con reglas de comunicación evolutiva.

La primera parte de la memoria finaliza con el **Capítulo 5** cuyo objetivo es la presentación de los resultados que, con anterioridad a este trabajo, han sido establecidos en relación con las fronteras de la eficiencia en el paradigma de la computación celular con membranas. Específicamente, resultados referidos a los modelos que trabajan, bien a modo de células o bien a modo de tejidos y que han sido introducidos en el capítulo anterior.

Parte II: Cooperación minimal

La segunda parte de esta memoria tiene como objetivo presentar con todo detalle las contribuciones originales que se aportan para abordar la resolución del problema **P** *versus* **NP**. Las aportaciones están estructuradas, a su vez, en dos bloques claramente diferenciados cuyo objetivo común es la obtención de fronteras de la eficiencia en distintos tipos de sistemas de membranas.

El primero de esos bloques, constituido por los Capítulos 6 y 7, está dedicado al análisis, desde el punto de vista de las fronteras antes citadas, del uso de reglas cooperativas en el marco de los sistemas P con membranas activas y sin polarizaciones que, además, no permiten reglas de disolución. Es bien conocido que en la definición clásica, esos sistemas no utilizan cooperación en sus reglas; es decir, las correspondientes partes izquierdas de las reglas poseen un único objeto. Además, si se prohíben las reglas de disolución entonces dichos sistemas son no eficientes, en el sentido de que solo pueden resolver eficientemente

problemas de la clase de complejidad **P**. En los Capítulos 6 y 7 se estudia la eficiencia computacional de los sistemas de membranas citados cuando se incorpora un tipo particular de *cooperación*, denominada *minimal*, en las reglas de evolución de objetos, por una parte, y en las reglas de comunicación, por otra. Ese tipo de cooperación tiene la peculiaridad de que la parte izquierda de las reglas consta, exactamente, de dos objetos. En ese contexto se analizan los casos en que la parte derecha consta de un único objeto (*producción minimal*) o más objetos.

Parte III: Comunicación evolutiva

El segundo bloque, constituido por los Capítulos 8 y 9, está dedicado al análisis de las fronteras de la eficiencia en unas variantes de los sistemas P con reglas symport/antiport en donde se admite la posibilidad de que evolucionen los objetos en esas reglas de comunicación. El estudio se realiza tanto para sistemas de membranas que trabajan a modo de células como para aquellos que trabajan a modo de tejidos, obteniéndose una serie de resultados que permite deducir que la estructura utilizada (*árbol enraizado*, en el caso de sistemas que trabajan a modo de células, *versus grafo dirigido*, en el caso de sistemas que trabajan a modo de tejidos) es irrelevante respecto a las fronteras de la eficiencia de estos sistemas de membranas.

Las **contribuciones originales** que se han realizado a lo largo de esta tesis son las siguientes:

1. Desarrollo de una nueva metodología para atacar el problema **P** *versus* **NP**, en particular usada en la memoria dentro del marco de la computación con membranas.

2. Diseño de soluciones para el problema `SAT` en distintas variantes de sistemas P con cooperación minimal en reglas de evolución, justificando su presumible eficiencia.

3. Diseño de soluciones para el problema `SAT` en sistemas P con cooperación minimal en reglas de comunicación y reglas de división.

4. Establecimiento de la *no eficiencia* de los sistemas P con cooperación minimal en reglas de comunicación y reglas de separación.

5. Mejora de los resultados (y por tanto, refinamiento de las fronteras) en los sistemas P a modo de tejidos con reglas de comunicación evolutiva.

6. Establecimiento de la presumible eficiencia mediante el desarrollo de soluciones eficientes del problema `SAT` en distintas variantes de sistemas P a modo de células con reglas de comunicación evolutiva.

7. Demostración de la *no eficiencia* de ciertas variantes de sistemas P a modo de células con reglas de comunicación evolutiva.

Parte V: Conclusiones y trabajo futuro

La tercera parte de la memoria pone el punto (y seguido) a la labor desempeñada. El **Capítulo 10** comienza con las conclusiones extraídas del trabajo realizado y de las aportaciones comentadas en esta sección. Posteriormente se proponen las principales líneas de investigación en las que ya se ha empezado a trabajar para continuar la senda abierta por esta tesis, así como las principales ideas de líneas de trabajo futuro que se considera podrían aportar avances significativos en la búsqueda de nuevos métodos para abordar la resolución del problema **P** *versus* **NP**.

Publicaciones relacionadas con la tesis

Cabe destacar las siguientes aportaciones originales del trabajo descrito en este documento:

Artículos en revistas indexadas en JCR-ISI:

1. L. Valencia-Cabrera, D. Orellana-Martín, M.Á. Martínez-del-Amor, M.J. Pérez-Jiménez. Reaching efficiency through collaboration in membrane systems: Dissolution, polarization and cooperation. *Fundamenta Informaticae*, **153**, 1-2 (2017), 147-172 (doi: 10.1016/j.tcs.2017.04.015). [JCR IF 2017: **0.772** (**Q3** - 77/103 - Computer Science, Theory and Methods)]

2. L. Valencia-Cabrera, D. Orellana-Martín, M.Á. Martínez-del-Amor, A. Riscos-Núñez, M.J. Pérez-Jiménez. Computational Efficiency of minimal cooperation and distribution in polarizationless P systems with active membranes. *Fundamenta Informaticae*, **153**, 1-2 (2017), 147-172. (doi:10.3233/FI-2017-1535). [JCR IF 2017: **0.725** (**Q3** - 173/252 - Mathematics, Applied) (also: 89/104 - Computer Science, Software Engineering)]

3. L. Valencia-Cabrera, D. Orellana-Martín, M.Á. Martínez-del-Amor, A. Riscos-Núñez, M.J. Pérez-Jiménez. Cooperation in Transport of Chemical Substances: A Complexity Approach within Membrane Computing. *Fundamenta Informaticae*, **154** (2017), 373-385 (doi: 10.3233/FI-2017-1572). [JCR IF 2017: **0.725** (**Q3** - 173/252 - Mathematics, Applied) (also: 89/104 - Computer Science, Software Engineering)]

4. L. Valencia-Cabrera, D. Orellana-Martín, M.Á. Martínez-del-Amor, A. Riscos-Núñez, M.J. Pérez-Jiménez. From distribution to replication in cooperative systems with active membranes: A frontier of the efficiency. *Theoretical Computer Science*, **736** (2018), 15-24 (doi: 10.1016/j.tcs.2017.12.012). [JCR IF 2017: **0.772** (**Q3** - 77/103 - Computer Science, Theory and Methods)]

5. D. Orellana-Martín, M.Á. Martínez-del-Amor, I. Pérez-Hurtado, A. Riscos-Núñez, L. Valencia-Cabrera, M.J. Pérez-Jiménez. When object production tunes the efficiency of membrane systems. *Theoretical Computer Science*, in press (2018) (doi: 10.1016/j.tcs.2018.04.013). [JCR IF 2017: **0.772** (**Q3** - 77/103 - Computer Science, Theory and Methods)]

6. D. Orellana-Martín, M.Á. Martínez-del-Amor, L. Valencia-Cabrera, B. Song, L. Pan, M.J. Pérez-Jiménez. P systems with symport/antiport rules: When do the surroundings matter? *Theoretical Computer Science*, in press (2018) (doi: 10.1016/j.tcs.2018.04.052). [JCR IF 2017: **0.772** (**Q3** - 77/103 - Computer Science, Theory and Methods)]

Artículos en revistas no indexadas en JCR-ISI:

1. D. Orellana-Martín, L. Valencia-Cabrera, A. Riscos-Núñez, M.J. Pérez-Jiménez. Minimal cooperation as a way to achieve the efficiency in cell-like membrane systems. Journal of Membrane Computing, in press (2018).

2. L. Valencia-Cabrera, M.Á. Martínez-del-Amor, D. Orellana-Martín, I. Pérez-Hurtado, M.J. Pérez-Jiménez. Connections between literals in the Boolean satisfiability problem and cooperation in membrane systems. Journal of Membrane Computing, in press (2018).

3. L. Valencia-Cabrera, M.Á. Martínez-del-Amor, D. Orellana-Martín, I. Pérez-Hurtado, M.J. Pérez-Jiménez. Cooperative P Systems and the P versus NP Problem. Bulletin of the International Membrane Computing Society, **4**, December 2017, 39-64.

4. L. Valencia-Cabrera, D. Orellana-Martín, A. Riscos-Núñez, M.J. Pérez-Jiménez. Complexity Perspectives on Minimal Cooperation in Cell-like Membrane Systems. Bulletin of the International Membrane Computing Society, **2**, December 2016, 69-78.

Artículos presentados en congresos internacionales:

1. L. Valencia-Cabrera, D. Orellana-Martín, M.Á. Martínez-del-Amor, A. Riscos-Núñez, M.J. Pérez-Jiménez. Restricted Polarizationless P Systems with Active Membranes: Minimal Cooperation Only Outwards. Proceedings of the Fifteenth Brainstorming Week on Membrane Computing, January 31 - February 3. 2017, Sevilla, Spain, 253-290.

2. L. Valencia-Cabrera, D. Orellana-Martín, M.Á. Martínez-del-Amor, A. Riscos-Núñez, M.J. Pérez-Jiménez. Restricted Polarizationless P Systems with Active Membranes: Minimal Cooperation Only Inwards. Proceedings of the Fifteenth Brainstorming Week on Membrane Computing, January 31 - February 3. 2017, Sevilla, Spain, 215-252.

3. D. Orellana-Martín, L. Valencia-Cabrera, B. Song, L. Pan, M.J. Pérez-Jiménez. Narrowing Frontiers of Efficiency with Evolutional Communication Rules and Cell Separation. Proceedings of the Sixteenth Brainstorming Week on Membrane Computing, January 30 - February 2. 2018, Sevilla, Spain, 139-162.

4. L. Valencia-Cabrera, D. Orellana-Martín, A. Riscos-Núñez, M.J. Pérez-Jiménez. Minimal cooperation in polarizationless P systems with active membranes. Proceedings of the Fourteenth Brainstorming Week on Membrane Computing, February 1 - 5, 2016, Sevilla, Spain, 327-356.

Otras publicaciones

A lo largo de la trayectoria investigadora del autor, se ha realizado aportaciones en otras áreas cercanas a las desarrolladas en esta memoria y que se enumeran a continuación.

Artículos en revistas indexadas en JCR-ISI:

1. M.J. Pérez-Jiménez, C. Graciani, D. Orellana-Martín, A. Riscos-Núñez, Á. Romero-Jiménez, L. Valencia-Cabrera. Fuzzy reasoning spiking neural

P systems revisited: A formalization. Theoretical Computer Science, **701** (2017), 216-225 (doi: 10.1016/j.tcs.2017.04.014). [JCR IF 2017: **0.772** (**Q3** - 77/103 - Computer Science, Theory and Methods)]

2. D. Orellana-Martín, L. Valencia-Cabrera, A. Riscos-Núñez, M.J. Pérez-Jiménez. The unique satisfiability problem from a membrane computing perspective. Romanian Journal of Information Science and Technology, **21**, 3 (2018), 288-297. [JCR IF 2017: **0.288** (**Q4** - 103/103 - Computer Science, Theory and Methods) (also: **Q4** - 60/61 - Instruments & Instrumentation; **Q4** - 145/146 - Physics, Applied)]

3. D. Orellana-Martín, L. Valencia-Cabrera, A. Riscos-Núñez, M.J. Pérez-Jiménez. A path to computational efficiency through membrane computing. Theoretical Computer Science, in press (2018). [JCR IF 2017: **0.772** (**Q3** - 77/103 - Computer Science, Theory and Methods)]

4. I. Pérez-Hurtado, M.J. Pérez-Jiménez, G. Zhang, D. Orellana-Martín. Simulation of Rapidly-Exploring Random Trees in Membrane Computing with CUDA. International Journal of Computers Communications & Control, in press (2018). [JCR IF 2017: **1.290** (**Q3** - 98/148 - Computer Science, Information Systems) (also: **Q3** - 44/61 - Automation & Control Systems)]

Artículos en revistas no indexadas en JCR-ISI:

1. D. Orellana-Martín, M.Á. Martínez-del-Amor, L. Valencia-Cabrera, A. Riscos-Núñez, M.J. Pérez-Jiménez. The role of integral membrane proteins in computational complexity theory. International Journal of Advances in Engineering Sciences and Applied Mathematics, **10**, 3 (2018), 193-202 (doi: 10.1007/s12572-018-0220-2).

2. D. Orellana-Martín, L. Valencia-Cabrera, A. Riscos-Núñez, M.J. Pérez-Jiménez. P systems with proteins: a new frontier when membrane division disappears. Journal of Membrane Computing, in press (2018).

3. L. Valencia-Cabrera, D. Orellana-Martín, M.Á. Martínez-del-Amor, M.J. Pérez-Jiménez. An interactive timeline of simulators in membrane computing. Journal of Membrane Computing, in press (2018).

4. D. Orellana-Martín, L. Valencia-Cabrera, J.L. Guisado, F. Jiménez-Morales, M.J. Pérez-Jiménez. Laser Dynamics from a Membrane Computing Perspective. Bulletin of the International Membrane Computing Society, **5**, June 2018, 97-108.

5. L. Valencia-Cabrera, D. Orellana-Martín, M.Á. Martínez-del-Amor, A. Riscos-Núñez, M.J. Pérez-Jiménez. From Super-cells to Robotic Swarms: Two Decades of Evolution in the Simulation of P Systems. Bulletin of the International Membrane Computing Society, **4**, December 2017, 65-87.

Artículos presentados en congresos internacionales:

1. M.ª Martínez-del-Amor, D. Orellana-Martín, F.G. Cabarle, M.J. Pérez-Jiménez, H.N. Adorna. Sparse Matrix Representation of Spiking Neural P Systems for GPU. Proceedings of the Fifteenth Brainstorming Week on Membrane Computing, January 31 - February 3, 2017, Sevilla, Spain, 161-170.

2. D. Orellana-Martín. P systems with Active Cells. Proceedings of the Fifteenth Brainstorming Week on Membrane Computing, January 31 - February 3, 2017, Sevilla, Spain, 175-189.

3. L. Valencia-Cabrera, D. Orellana-Martín, A. Riscos-Núñez, M.J. Pérez-Jiménez. Counting Membrane Systems. Proceedings of the Eighteenth International Conference on Membrane Computing (CMC18), July 24 - 28, 2017, Bradford, United Kingdom, 359-372.

4. I. Pérez-Hurtado, M.J. Pérez-Jiménez, G. Zhang, D. Orellana-Martín. Robot Path Planning using Rapidly-exploring Random Trees: A Membrane Computing Approach. *IEEE Proceedings of 2018 7th International Conference on Computers Communications and Control*, May 8 - 12, 2018, Oradea, Romania, 37-46. (Best Paper Award)

5. D. Orellana-Martín, L. Valencia-Cabrera, A. Riscos-Núñez, M.J. Pérez-Jiménez. Limits on P systems with Proteins and Without Division. Proceedings of the Sixteenth Brainstorming Week on Membrane Computing, January 30 - February 2. 2018, Sevilla, Spain, 123-138.

6. D. Orellana-Martín, L. Valencia-Cabrera, M.J. Pérez-Jiménez. The factorization problem: A new approach through membrane systems. *Workshop on Membrane Computing, satellite workshop of the 17th International Conference on Unconventional Computation and Natural Computation, UCNC 2018*, June 25 - 29, 2018, Fontainebleau, France, 39-56.

7. M.Á. Martínez-del-Amor, D. Orellana-Martín, A. Riscos-Núñez, M.J. Pérez-Jiménez. On GPU-Oriented P systems. Proceedings of the 2018 International Conference on High Performance Computing & Simulation,

July 16 - 20, 2018, Orléans, France, 780-781. (doi: 10.1109/HPCS.2018.00125).

Posters presentados en congresos internacionales:

1. A. Riscos-Núñez, M.J. Pérez-Jiménez, M.C. Lemos, F. Jiménez, A. Millán, M.Á. Martínez-del-Amor, J.L. Guisado, Á. Romero-Jiménez, C. Graciani, D. Cagigas, A. Cordoba, L. Valencia-Cabrera, D. Orellana-Martín, D.H. Campora-Pérez, F. Díaz, A. Ríos, J.P. Domínguez, R. Tapiador. MABICAP: Bio-inspired Machines over High Performance Computing. GPU Technology Conference Europe (GTC Europe 2018), Munich (Germany), October 2018.

Desarrollo de software

Durante el desarrollo de la tesis en particular y de la investigación en general se han utilizado diversas herramientas software como apoyo para el diseño de las distintas soluciones de problemas computacionalmente duros o la modelización de diversos sistemas físicos. No obstante, dado que el software existente carecía de ciertas funcionalidades cruciales para el correcto funcionamiento o, simplemente, que serían útiles para observar ciertos resultados, el autor ha mejorado algunas de las herramientas existentes, e incluso ha generado nuevas herramientas para posibles investigaciones futuras. Dichas contribuciones se describen a continuación.

- Contribuciones al proyecto de código abierto P-Lingua. Las más importantes son las siguientes: inclusión de algunas funciones que pueden ser llamadas en los ficheros `.pli`; desarrollo de nuevos modelos de sistemas P; compilador `.pli`-a-binario para sistemas PDP [72].
- Desarrollo del plugin de Sevilla Carpets, totalmente integrado en el entorno MeCoSim [71].
- Mejoras en ABCD-GPU, permitiendo que la herramienta simule sistemas PDP que no eran compatibles en versiones anteriores [73].

Parte I

Estado del Arte

Capítulo 1

Preliminares

El trabajo que se presenta en esta memoria ha sido elaborado con la intención de ser, en cierto sentido, autocontenido. Por ello, hemos creído conveniente dedicar un capítulo a la descripción, de manera más o menos sucinta, de todos aquellos conceptos y resultados básicos que son necesarios para una mejor comprensión del texto y, lo que también es importante, para fijar las notaciones que van a ser utilizadas a lo largo del mismo.

1.1. Lenguajes y multiconjuntos

Un *alfabeto* Γ es un conjunto no vacío. Los elementos de un alfabeto suelen denominarse *símbolos*. Una *cadena* (o palabra) sobre Γ es una *sucesión finita ordenada* de elementos de Γ; es decir, una aplicación de un número natural n en el conjunto Γ. El número n se denomina *longitud* de la cadena. La *cadena vacía* es la cadena de longitud 0 y se denotará por λ. Notaremos por Γ^* el conjunto de todas las cadenas sobre el alfabeto Γ. El *conjunto* Γ^+ está formado por todas las cadenas sobre Γ, a excepción, de la cadena vacía. Un *lenguaje* sobre Γ es un subconjunto de Γ^*.

Dados dos conjuntos A y B, el *complemento relativo* $A \setminus B$ de B en A se define como sigue: $A \setminus B = \{x \in A \mid x \notin B\}$. Por tanto, $\Gamma^+ = \Gamma^* \setminus \{\lambda\}$. Para cada conjunto A, notaremos por $|A|$ el *cardinal* (número de elementos) del conjunto A. Dado un alfabeto finito y ordenado $\Sigma = \{a_1, \ldots, a_r\} \subseteq \Gamma$, la *función de Parikh* Ψ_Σ es una aplicación de Γ^* en $\mathbb{N}^r$ definida como sigue: $\Psi_\Sigma(u) = (|u|_{a_1}, \ldots, |u|_{a_r})$, donde $|u|_{a_i}, 1 \leq i \leq r$, denota el número de apariciones del símbolo a_i en la cadena u.

Un *multiconjunto* $\mathcal{M}$ puede ser descrito explícitamente como sigue:

$$\{(a_1, \mathcal{M}(a_1)), \ldots, (a_n, \mathcal{M}(a_n))\}$$

y se utilizará la notación $\mathcal{M} = a_1^{\mathcal{M}(a_1)} \ldots a_n^{\mathcal{M}(a_n)}$. El *cardinal* de un multiconjunto finito sobre $\Gamma = \{a_1, \ldots, a_n\}$ se define como sigue: $|\mathcal{M}| = \mathcal{M}(a_1) + \cdots + \mathcal{M}(a_n)$. Notaremos por $\mathcal{M}_f(\Gamma)$ el conjunto de todos los multiconjuntos finitos sobre Γ.

Dados dos multiconjuntos $\mathcal{M}_1, \mathcal{M}_2$ sobre un mismo alfabeto Γ, se define como la *unión* de los multiconjuntos $\mathcal{M}_1$ y $\mathcal{M}_2$, que se notará $\mathcal{M}_1 \cup \mathcal{M}_2$ o $\mathcal{M}_1 + \mathcal{M}_2$, como sigue: es la aplicación de Γ en $\mathbb{N}$ tal que $(\mathcal{M}_1 + \mathcal{M}_2)(a) = \mathcal{M}_1(a) + \mathcal{M}_2(a)$, para cada $a \in \Gamma$. La *intersección* de $\mathcal{M}_1$ y $\mathcal{M}_2$, que se notará como $\mathcal{M}_1 \cap \mathcal{M}_2$, es la aplicación de Γ en $\mathbb{N}$ definida como sigue: $(\mathcal{M}_1 \cap \mathcal{M}_2)(a) = min\{\mathcal{M}_1(a), \mathcal{M}_2(a)\}$, para cada $a \in \Gamma$. Se dirá que el multiconjunto $\mathcal{M}_1$ está incluido en $\mathcal{M}_2$ y se notará $\mathcal{M}_1 \subseteq \mathcal{M}_2$, si $\mathcal{M}_1(a) \leq \mathcal{M}_2(a)$, para cada $a \in \Gamma$.

1.2. Grafos, árboles y ciclos Hamiltonianos

En esta sección, se introducen algunos conceptos básicos de teoría de grafos que van a ser usados a lo largo de este trabajo. Para más detalles, véase [12].

Un *grafo dirigido* G es un par ordenado (V, E), en donde V es un conjunto finito cuyos elementos se denominan *vértices* o *nodos* del grafo, y $E = \{(x, y) \mid x \in V, y \in V\}$ es un conjunto finito cuyos elementos se denominan *arcos* del grafo. Si (x, y) es un arco de G, se dice que los nodos x e y son los extremos de dicho arco (también se dice que el nodo y es *adyacente* al nodo x). Un *grafo no dirigido* G es un par ordenado (V, E), en donde V es un conjunto finito cuyos elementos se denominan *vértices* o *nodos* del grafo, y $E = \{\{x, y\} \mid x \in V, y \in V, x \neq y\}$ es un conjunto finito cuyos elementos se denominan *aristas* del grafo. Si $\{x, y\}$ es una arista de G, se dice que los nodos x e y son los extremos de dicha arista (también se dice que los nodos x e y son *adyacentes*). Obsérvese que mientras en un grafo dirigido pueden existir arcos con los dos extremos iguales, en un grafo no dirigido los dos extremos de una arista deben ser nodos distintos. Es interesante tener presente que, en cierto sentido, un grafo no dirigido $G = (V, E)$ puede considerarse como un caso particular de un grafo dirigido $G' = (V, E')$ en el cuál, para cada par de nodos distintos x e y se tiene que $\{x, y\} \in E$ si y sólo si $(x, y) \in E'$ e $(y, x) \in E'$.

Un *recubrimiento de vértices* de un grafo no dirigido $G = (V, E)$ es un conjunto de vértices $V' \in V$ que satisface la siguiente condición: para cada

arista del grafo, al menos uno de los extremos de la misma debe ser un nodo de V'.

Dado un grafo $G = (V, E)$ (dirigido o no dirigido), un *camino de longitud* $k \geq 1$ que va desde un nodo x a un nodo y es una $(k+1)$-tupla ordenada $(u_1, \ldots, u_{k+1})$ que satisface las condiciones siguientes: (a) $u_1 = x, u_{k+1} = y$; y (b) para cada $i, 1 \leq i \leq k$, se tiene que $(u_i, u_{i+1}) \in E$, en el caso dirigido, o $\{u_i, u_{i+1}\} \in E$, en el caso no dirigido. Dados dos nodos x, y de un grafo G diremos que y es *alcanzable* desde el nodo x si existe, al menos, un camino en G que va desde x hasta y. Se dice que un *camino* $(u_1, \ldots, u_{k+1})$ de un grafo G es *simple* si $|\{u_1, \ldots, u_{k+1}\}| = k+1$; es decir, si todos los nodos del camino son distintos entre sí. Un camino simple de un grafo G se dice que es *Hamiltoniano* si contiene a todos los nodos del grafo. Un *ciclo* de un grafo G es un camino $(u_1, \ldots, u_{k+1})$ de G tal que $u_1 = u_{k+1}$. Se dice que un *ciclo* $(u_1, \ldots, u_{k+1})$ de G es simple si $|\{u_1, \ldots, u_{k+1}\}| = k + 1$. En el caso de grafos dirigidos, convendremos en que si (x, x) es un arco entonces determina un ciclo simple. Se dice que un grafo G (dirigido o no dirigido) es *acíclico* si carece de ciclos. Un *ciclo Hamiltoniano* de un grafo $G = (V, E)$ es un ciclo simple $(u_1, \ldots, u_{k+1})$ tal que $k \geq 1$ y $\{u_1, \ldots, u_{k+1}\} = V$.

Se dice que un grafo G no dirigido es *conexo* si para cada par de vértices distintos existe, al menos, un camino que va desde el uno hasta el otro. Un *árbol enraizado* es un grafo no dirigido, conexo y acíclico que tiene un vértice distinguido (llamado *raíz del árbol*). Dado un nodo x de un árbol enraizado, distinto de la raíz r, cualquier nodo $y \neq x$ del único camino que va desde la raíz al nodo x se denomina *antecesor* de x. Si y es un antecesor de x se dice que x es un *descendiente* de y. Si la última arista del único camino que va desde r al nodo x es $\{x, y\}$ (por tanto, $x \neq y$), entonces y se denomina el *padre* del nodo x y se dice, además, que x es un *hijo* del nodo y. En tal situación, notaremos $y = p(x)$ y $x \in ch(x)$. La raíz r de un árbol enraizado es el único nodo del árbol que no tiene padre. Con frecuencia, convendremos en notar $p(r) = 0$, en donde $0 \notin V$. Un nodo del árbol que no tenga hijos se denomina *hoja* del árbol.

1.3. Conceptos de lógica proposicional

El lenguaje de la lógica proposicional consta de: (a) un conjunto numerable, VP, de variables proposicionales; (b) unas conectivas lógicas ($\neg$, negación y $\vee$, disyunción); y (c) unos símbolos auxiliares, (y).

El conjunto $PForm$ de las *fórmulas proposicionales* es el menor conjun-

to Γ que contiene a VP y verifica las condiciones siguientes: (a) si $P \in \Gamma, entonces \neg P \in \Gamma$; y (b) si $P, Q \in \Gamma$, entonces $(P \vee Q) \in \Gamma$. A partir de $\neg$ y $\vee$ se definen las conectivas lógicas $\wedge$, $\rightarrow$ y $\leftrightarrow$, de acuerdo con las tablas de verdad usuales. Notaremos las fórmulas $\neg P$, $P \vee Q$ y $P \wedge Q$ por $\overline{P}$, $P + Q$, y $P \cdot Q$.

Un *literal* es una variable proposicional o la negación de una variable proposicional. Una *cláusula* es la disyunción de un número finito de literales. Una fórmula proposicional está en *forma normal conjuntiva* (FNC) si es la conjunción de un número finito de cláusulas; es decir, toda fórmula proposicional en FNC es la conjunción de una disyunción de literales. Además, se puede suponer, sin que ello implique restricción alguna, que toda fórmula proposicional en FNC está en *forma simplificada*; es decir, en cada cláusula de dicha fórmula no puede existir un literal y su negación ni, tampoco, dos literales repetidos.

Una *valoración de verdad* o *asignación de verdad* es una aplicación de VP en $\{0, 1\}$. Toda valoración de verdad se extiende de manera natural a una aplicación de $PForm$ en $\{0, 1\}$ (a través de las *tablas de verdad*). Una valoración de verdad *relevante* para una fórmula φ es una aplicación del conjunto de variables de dicha fórmula en el conjunto $\{0, 1\}$. Por tanto, si una fórmula proposicional φ posee n variables, entonces el número total de valoraciones relevantes para φ es 2^n.

Diremos que una fórmula proposicional, φ, es *satisfactible* si y sólo si existe, al menos, una valoración de verdad, σ, tal que $\sigma(\varphi) = 1$. Diremos que dos fórmulas proposicionales son *semánticamente equivalentes* si cualquier valoración le asigna a ambas el mismo valor. Es fácil probar que toda fórmula proposicional tiene una fórmula semánticamente equivalente en FNC y simplificada.

1.4. Problemas abstractos

Informalmente, un *problema abstracto* es una «cuestión general a responder que, usualmente, posee varios parámetros cuyos valores no son especificados» [16]. *Resolver* un problema abstracto consiste en responder a la cuestión planteada. Por tanto, un problema abstracto consta de un conjunto (finito o infinito) de *problemas concretos*, denominados *instancias*, que se obtienen mediante la especificación de los valores de los parámetros asociados al problema. Cada instancia tiene asociada un conjunto (eventualmente vacío) de posibles *soluciones* y la respuesta a la cuestión general formulada está relacionada con este conjunto.

Informalmente, un *problema de búsqueda* es un problema abstracto tal que la cuestión consiste en identificar o encontrar *una* solución del conjunto de soluciones asociadas a cada instancia y, en caso de no existir ninguna solución asociada a una cierta instancia, responder «*no*» a esa instancia. Así por ejemplo, dada una fórmula Booleana φ en forma normal conjuntiva, hallar una asignación de verdad a las variables de φ que la hace verdadera, y si no existe una tal asignación entonces responder «*no*» es un problema de búsqueda. De esta manera, dicho problema se puede identificar con una cierta *función* que para cada entrada puede tener muchas soluciones diferentes o ninguna.

Un *problema de optimización* es un caso particular de un problema de búsqueda cuya finalidad es encontrar una *mejor solución* asociada a cada instancia del problema, de entre un conjunto de posibles soluciones candidatas, de acuerdo con un concepto de optimalidad especificado por una *función objetivo*. Así por ejemplo, el problema de optimización `MAXSAT` consiste en lo siguiente: dada una fórmula Booleana φ en forma normal conjuntiva y un número natural $k \geq 1$, determinar una asignación de verdad a las variables de φ que hace verdaderas, al menos, k cláusulas de la fórmula.

Un *problema de decisión* es otro caso particular de un problema de búsqueda y de un problema de optimización. Específicamente, de manera informal se puede afirmar que un problema de decisión es un problema abstracto cuyas únicas respuestas pueden ser «*sí*» o «*no*». Este tipo de problemas puede ser formulado especificando una instancia genérica del mismo y formulando una cuestión *sí/no*relativa a dicha instancia. Un ejemplo de problema de decisión es el problema `SAT` de la satisfactibilidad de la lógica proposicional, que puede ser formulado como sigue: dada una fórmula Booleana φ en forma normal conjuntiva determinar si existe una asignación de verdad a las variables de φ que la hace verdadera.

A continuación vamos a formalizar los conceptos «problema de búsqueda», «problema de optimización» y «problema de decisión».

Definición 1.1. *Un problema de búsqueda X es una tupla (Σ_X, I_X, S_X) en donde: (a) Σ_X es un alfabeto finito; (b) I_X es un lenguaje sobre Σ_X cuyos elementos se denominan instancias de X; y (c) S_X es una función cuyo dominio es I_X. Para cada instancia $u \in I_X$, $S_X(u)$ es un conjunto cuyos elementos se denominan soluciones asociadas a la instancia u.*

Resolver un problema de búsqueda, X, significa lo siguiente: para cada instancia $u \in I_X$ hallar un elemento del conjunto $S_X(u)$, en el caso en que éste sea no vacío y, en caso contrario, devolver «*no*». Es decir, resolver un problema de búsqueda consiste en *buscar*, *encontrar*, *hallar* algún elemento

del conjunto $S_X(u)$ de soluciones asociadas a u, en el caso en que $S_X(u) \neq \emptyset$; en caso contrario, devuelve «*no*».

Todo problema de búsqueda $X = (\Sigma_X, I_X, S_X)$ tiene asociado, de manera natural, una relación binaria Q_X definida como sigue: $Q_X = \{(u, z) \mid u \in I_X \wedge z \in S_X(u)\}$. En este contexto, resolver el problema de búsqueda X puede ser interpretado así: dada una instancia $u \in X$, hallar/encontrar un elemento z tal que $(u, z) \in Q_X$.

Definición 1.2. *Un problema de optimización X es una tupla $(\Sigma_X, I_X, S_X, O_X)$ tal que:*

- *(Σ_X, I_X, S_X) es un problema de búsqueda tal que para cada $u \in I_X$ se tiene que $S_X(u) \neq \emptyset$.*
- *O_X es una función (denominada objetivo) cuyo dominio es I_X y para cada instancia $u \in I_X$ y para cada posible solución $a \in S_X(u)$ asociada a u, $O_x(u, a)$ es un número racional positivo.*
- *Para cada instancia $u \in I_X$ existe una solución $a \in S_X(u)$ tal que:*
  - *O bien $\forall b(b \in S_X(u) \Rightarrow O_X(u, b) \leq O_X(u, a))$ (en este caso diremos que a es una solución maximal para la instancia u).*
  - *O bien $\forall b(b \in S_X(u) \Rightarrow O_X(u, b) \geq O_X(u, a))$ (en este caso diremos que a es una solución minimal para la instancia u).*

Resolver un problema de optimización X consiste en lo siguiente: para cada instancia $u \in I_X$, devolver una *solución maximal* o una *solución minimal*. Diremos que una máquina de Turing determinista M resuelve un problema de optimización X si dada cualquier instancia $u \in I_X$, la máquina M con dato de entrada u devuelve una *solución óptima* (maximal o minimal) asociada a dicha instancia.

Definición 1.3. *Un problema de decisión X es un problema de búsqueda (Σ_X, I_X, S_X) tal que para cada instancia $u \in I_X$, $S_X(u) = \{0\}$ o $S_X(u) = \{1\}$. En el caso $S_X(u) = \{0\}$ diremos que la respuesta del problema de decisión es negativa («*no*») para la instancia u. En el caso $S_X(u) = \{1\}$ diremos que la respuesta del problema de decisión es afirmativa («*sí*») para dicha instancia u.*

Resolver un problema de decisión X consiste en lo siguiente: para cada instancia $u \in I_X$, devolver *sí* en el caso $S_X(u) = \{1\}$, y devolver *no* en caso contrario. Obsérvese que todo problema de decisión $X = (\Sigma_X, I_X, S_X)$ puede

ser considerado como un problema de optimización $(\Sigma_X, I_X, S_X, O_X)$, en donde $O_X(u, a)$ es constante e igual a 1 (recuérdese que para cada instancia $u \in I_X$ el conjunto de posibles soluciones $S_X(u)$ contiene un único elemento y, por tanto, o bien es igual a $\{0\}$, o bien es igual a $\{1\}$).

Todo problema de decisión $X = (\Sigma_X, I_X, S_X)$ tiene asociado, de manera natural, un lenguaje L_X definido como sigue: $L_X = \{u \in \Sigma_X^* \mid S_X(u) = \{1\}\}$. Recíprocamente, cada lenguaje L sobre un alfabeto Γ tiene asociado un problema de decisión $X_L = (\Sigma_{X_L}, I_{X_L}, S_{X_L})$ definido como sigue: $\Sigma_{X_L} = \Gamma$, $I_{X_L} = \Gamma^*$ en donde $S_{X_L}(u) = \{1\}$, para cada $u \in L$, y $S_{X_L}(u) = \{0\}$, para cada $u \notin L$. De acuerdo con estas definiciones, para cada problema de decisión X tenemos que $X_{L_X} = X$ y para cada lenguaje L tenemos que $L_{X_L} = L$.

Diremos que una máquina de Turing determinista M resuelve un problema de decisión X si la máquina M *reconoce* o *decide* el lenguaje L_X asociado al problema X; es decir, para cada cadena u sobre Σ_X, si $u \in L_X$, entonces la respuesta de la máquina M con dato de entrada u es *sí*(M *acepta* u) y, en caso contrario, la respuesta es *no*(M *rechaza* u). El concepto de MTD que trabaja en tiempo polinomial se define de manera natural a través del número de pasos que proporciona la (única) computación de parada asociada a cada instancia del problema. Diremos que una máquina de Turing *no determinista* M resuelve un problema de decisión X si la máquina M *reconoce* o *decide* L_X; es decir, si para cada cadena u sobre Σ_X se tiene que $u \in L_X$ si y sólo si existe al menos una computación de la máquina M con dato de entrada u tal que la respuesta es *sí*. Diremos que una máquina de Turing no determinista M que resuelve un problema de decisión X, trabaja en tiempo polinomial si existe un polinomio $p(n)$ que verifica la siguiente condición: para cada instancia u de X de tamaño n, la máquina M acepta u si y sólo si existe, al menos, una computación de M con entrada u tal que la respuesta es *sí*y realiza a lo sumo $p(n)$ pasos de transición. Así pues, dada una máquina de Turing no determinista M que resuelve un problema de decisión X y trabaja en tiempo acotado por un polinomio $p(n)$, se verifica lo siguiente: para cada instancia u de tamaño n, si ninguna computación de $M(u)$ devuelve *sí*en, a lo sumo, $p(n)$ pasos, entonces la máquina M rechaza u.

Normalmente, la teoría de la complejidad computacional trabaja, básicamente, con problemas de decisión, a pesar que en la vida real gran parte de los problemas relevantes suelen ser problemas de optimización. No obstante, esta restricción no es especialmente significativa. En efecto, se puede probar que todo problema de optimización X tiene asociado un problema de decisión D_X, de tal manera que D_X se puede «construir» a partir de X con *poco coste* adicional (es decir, usando adecuadamente una máquina de Turing determi-

nista que trabaja en tiempo polinomial)y, además, dada una solución S_{D_X}del problema de decisión D_X se puede «construir» una solución del problema X con *poco coste* adicional. Así por ejemplo, el problema *Minimum Vertex Cover* (`MVC`) es el siguiente: dado un grafo no dirigido, hallar un recubrimiento de vértices de tamaño mínimo. Pues bien, dicho problema de optimización tiene un problema de decisión (`DVC`) asociado de manera natural como sigue: dado un grafo no dirigido G y un número natural k, determinar si G posee un recubrimiento de vértices de tamaño exactamente k. Además, si $S_{\texttt{DVC}}$ es una solución de la versión de decisión, entonces se puede construir una solución del problema de optimización `MVC` como en el Algoritmo 1.4.1.

Algoritmo 1.4.1 Algoritmo de resolución del problema de optimización `MVC`

Entrada: un grafo no dirigido $G = (V, E)$
  para $k = 1$ hasta $|V|$ **hacer**
    ejecutar la solución $S_{\texttt{DVC}}$ para la entrada (G, k)
    si la respuesta es *sí* **entonces**
      devolver *sí*
    fin si
    devolver *no*
  fin para

En este ejemplo de ilustración resulta que si la solución de $S_{\texttt{DVC}}$ de la versión de decisión tiene un coste en tiempo del orden $O(f(m))$, siendo $m = max\{|V|, |E|\}$, entonces el coste de la construcción de la solución $S_{\texttt{MVC}}$ de la versión de optimización a partir de la solución $S_{\texttt{DVC}}$, será del orden $O(|V|\cdot f(m))$.

Por tanto, una forma indirecta para hallar una *buena solución* de un problema de optimización X sería la siguiente: (a) generar un problema de decisión D_X asociado a X; (b) hallar una *buena solución* S_{D_X} del problema de decisión D_X; y (c) a partir de la solución S_{D_X} hallar una *buena solución* del problema de optimización X. Por supuesto, esta metodología no sería exitosa si somos incapaces de hallar una *buena solución* del problema de decisión D_X.

Finalmente, observemos que al tratar con problemas de decisión, la resolubilidad de los mismos puede ser expresada a través de reconocimiento de lenguajes. En consecuencia, a la hora de desarrollar una teoría de la complejidad computacional en un determinado paradigma, será importante considerar *dispositivos computacionales reconocedores* de lenguajes.

1.5. El problema SAT y el problema de la accesibilidad

El problema de la satisfactibilidad de la lógica proposicional (SAT), es el siguiente problema de decisión:

Dada una fórmula proposicional en forma normal conjuntiva y simplificada, determinar si es satisfactible.

Existen algoritmos de fuerza bruta que resuelven este problema en tiempo exponencial. Se desconoce la existencia de algún algoritmo de coste en tiempo polinomial que resuelva el problema SAT.

El problema de la accesibilidad (Reachability) es el siguiente problema de decisión:

Dado un grafo no dirigido G y dos nodos distinguidos s, t, determinar si existe, al menos, un camino en G que va desde el nodo s al nodo t.

Existen algoritmos que resuelven este problema basados en las estrategias de búsqueda en profundidad (*depth-first search*) o en anchura (*breadth-first search*). El coste en tiempo de estos algoritmos es del orden $O(max(|V|, |E|))$, para un grafo de entrada (V, E). Más aún, en estas aproximaciones es necesario almacenar, a lo sumo, $|V|$ elementos y, por ello, estos algoritmos una cantidad de espacio del orden $O(|V|)$. Pues bien, esta cantidad de espacio puede ser reducida al orden $O(\log^2 |V|)$ usando un algoritmo que podría ser denominado *middle-first search* (para más detalles, véase [40], pp. 149-150).

1.6. El problema del ciclo Hamiltoniano

El problema del ciclo Hamiltoniano (HAM-CYCLE) para grafos dirigidos es el siguiente problema de decisión:

Dado un grafo dirigido G, determinar si posee un ciclo Hamiltoniano.

Sea $(x_{i_1}, x_{i_2}, \ldots, x_{i_r}, x_{i_{r+1}})$ un camino simple de un grafo dirigido $G = (V, E)$, con $V = \{1, \ldots, n\}$. El camino γ antes descrito lo notaremos, así mismo, como el siguiente conjunto: $\{(x_{i_1}, x_{i_2})_1, \ldots, (x_{i_r}, x_{i_{r+1}})_r\}$. Así pues, en esta representación, el «arco etiquetado» $(x_{i_k}, x_{i_{k+1}})_k$ se interpreta como sigue: es el k-ésimo arco del camino γ, para cada $k, 1 \leq k \leq r$. En este trabajo, también usaremos las siguientes notaciones:

$$A_G = \{(i,j)_k \mid 1 \leq i,j,k \leq n, (i,j) \in E\};$$
$$A'_G = \{(i,j)'_k \mid 1 \leq i,j,k \leq n, (i,j) \in E\};$$
$$A''_G = \{(i,j)'_k \mid 1 \leq i,j,k \leq n, (i,j) \in E\}.$$

Es fácil probar el siguiente resultado:

Proposición 1.1. *Sea $G = (V,E)$ un grafo dirigido con $V = \{1,\ldots,n\}$. Sea $A_G = \{(i,j)_k \mid 1 \leq i,j,k \leq n, (i,j) \in E\}$. Si $B \subseteq A_G$ entonces los siguientes asertos son equivalentes:*

1. *B es un ciclo Hamiltoniano.*
2. *$|B| = n$ y, además, para cada $i,i',j,j',k,k' \in \{1,\ldots n\}$ se tiene lo siguiente:*
   - *(a) si $(i,j)_k \in B$, $(i',j')_{k'} \in B$ e $(i,j)_k \neq (i',j')_{k'}$, entonces $k \neq k'$;*
   - *(b) si $(i,j)_k \in B$, $(i',j')_{k'} \in B$ e $(i,j)_k \neq (i',j')_{k'}$, entonces $i \neq i'$;*
   - *(c) si $(i,j)_k \in B$, $(i',j')_{k'} \in B$ e $(i,j)_k \neq (i',j')_{k'}$, entonces $j \neq j'$;*
   - *(d) si $(i,j)_k \in B$, $(i',j')_{k+1} \in B$, entonces $j = i'$;*

Comentario 1.1. *Sea $B \subseteq A_G$ un ciclo Hamiltoniano de G. Para cada $i,i',j,j',k,k' \in \{1,\ldots,n\}$ se tiene lo siguiente:*

- *si $(i,j)_k \in B$ e $(i,j) \neq (i',j')$, entonces $(i',j')_{k'} \notin B$;*
- *si $(i,j)_k \in B$ e $i \neq i'$, entonces $(i',j')_{k'} \notin B$;*
- *si $(i,j)_k \in B$ y $j \neq j'$, entonces $(i,j')_{k'} \notin B$;*
- *si $(i,j)_k \in B$ y $j = i'$, entonces $(i',j')_{k+1} \in B$.*

Comentario 1.2. *Si $(x_{i_1}, x_{i_2}, \ldots, x_{i_n}, x_{i_1})$ es un ciclo Hamiltoniano de G, entonces puede ser descrito por el conjunto $B_1 = \{(x_{i_1},x_{i_2})_1, (x_{i_2},x_{i_3})_2, \ldots, (x_{i_n},x_{i_1})_n\} \subseteq A_G$. Ahora bien, $(x_{i_2}, x_{i_3}, \ldots, x_{i_n}, x_{i_1}, x_{i_2})$ también representa el mismo ciclo Hamiltoniano y puede ser descrito por $B_2 = \{(x_{i_2},x_{i_3})_1, (x_{i_3},x_{i_4})_2, \ldots, (x_{i_1},x_{i_2})_n\}$. Por tanto, dado un ciclo Hamiltoniano G, existen exactamente n subconjuntos diferentes de A_G codificando dicho ciclo.*

Comentario 1.3. *Supongamos que el número total de ciclos Hamiltonianos de G es q. Entonces, el número de diferentes subconjuntos B de A_G verificando las condiciones (a), (b), (c) y (d) de la Proposición 1.1 es exactamente $n \cdot q$.*

1.7. La función par de Cantor

La *función par* de cantor codifica pares de números naturales a través de números naturales individuales y está definida como sigue: para cada $m, n \in \mathbb{N}$,

$$\langle m, n\rangle = \frac{(m+n)(m+n+1)}{2} + m$$

La función par de Cantor es una función primitiva recursiva biyectiva de $\mathbb{N} \times \mathbb{N}$ en $\mathbb{N}$. Por tanto, para cada número natural $t \in \mathbb{N}$ existen unos (únicos) números naturales $m, n \in N$ tales que $t = \langle m, n\rangle$.

1.8. Recorrido de una matriz

A lo largo de este trabajo, vamos a utilizar matrices para representar fórmulas proposicionales en FNC y simplificada. De tal manera que cada una de las filas de esa matriz va a representar una cláusula de la fórmula y las distintas columnas están asociadas a las variables proposicionales de la fórmula. Así por ejemplo, la fórmula proposicional $\varphi = (x_1 + x_2 + \overline{x}_3) \cdot (\overline{x}_2 + x_4) \cdot (\overline{x}_2 + x_3 + \overline{x}_4)$ que consta de 3 cláusulas y 4 variables, se puede codificar por la siguiente matriz:

$$cod(\varphi) = \begin{pmatrix} x_{1,1,0} & x_{2,1,0} & \overline{x}_{3,1,0} & x^*_{4,1,0} \\ x^*_{1,2,0} & \overline{x}_{2,2,0} & x^*_{3,2,0} & x_{4,2,0} \\ x^*_{1,3,0} & \overline{x}_{2,3,0} & x_{3,3,0} & \overline{x}_{4,3,0} \end{pmatrix}$$

De tal manera que si el literal x_i (respectivamente, $\neg x_i$) aparece en la cláusula j-ésima entonces notaremos en la posición (j, i) de la matriz el objeto $x_{i,j,0}$ (respectivamente, $\overline{x}_{i,j,0}$). Si en la cláusula j-ésima no aparece en x_i ni $\neg x_i$, entonces notaremos en la posición (j, i) de la matriz el objeto $x^*_{i,j,0}$. Así pues, cada fila de esa matriz representa una cláusula de la fórmula y cada columna representa los literales de una cierta variable que aparece en φ. Es decir, dada una fórmula $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_p, C_j = l_{j_1} \vee \cdots \vee l_{j_{r_j}}, 1 \leq j \leq p, l_{j_k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$, tendremos el siguiente conjunto de elementos en dicha matriz:

$$\{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$$

Conviene hacer notar que en esa representación, el tercer subíndice (en este caso, 0) no tiene relación con variables ni cláusulas, sino con el procesamiento que, en determinados diseños, se haga con estos objetos a lo largo del tiempo.

Para implementar ciertos procesos de diseño, es interesante describir un mecanismo que permita «recorrer» dicha matriz de izquierda a derecha y de arriba hacia abajo. Veamos cómo se puede formalizar esta idea, partiendo de una cierta matriz con p filas y n columnas:

$$\begin{pmatrix} a_{1,1,0} & a_{2,1,0} & \dots & a_{n,1,0} \\ a_{1,2,0} & a_{2,2,0} & \dots & a_{n,2,0} \\ \dots\dots & \dots\dots & \dots & \dots\dots \\ \dots\dots & \dots\dots & \dots & \dots\dots \\ a_{1,p,0} & a_{2,p,0} & \dots & a_{n,p,0} \end{pmatrix}$$

Se trata de recorrer los objetos de dicha matriz, de izquierda a derecha y de arriba hacia abajo; es decir, en el orden descrito a continuación:

$$a_{1,1,0}\ a_{2,1,0} \dots\ a_{n,1,0}\ a_{n,1,0}\ a_{2,2,0} \dots\ a_{n,2,0} \dots\ a_{1,p,0} \dots\ a_{n,p,0}$$

En este recorrido, es importante observar que el elemento $a_{i,j,0}$ de la matriz ocupa la posición $[(j-1)\cdot n + i]$-ésima de la lista anterior.

Capítulo 2

Computabilidad versus complejidad computacional

El objetivo de este capítulo es introducir los conceptos básicos de la *teoría de la computabilidad* y la *teoría de la complejidad computacional* que permitan vislumbrar el verdadero significado y la trascendencia del problema **P** *versus* **NP** que vertebra este trabajo.

En la primera sección se describe, de manera sucinta, algunos hechos de especial relevancia relacionados con la génesis y posterior desarrollo de la teoría de la computabilidad. El objetivo de esta teoría consiste en proporcionar una definición matemática de la idea intuitiva de resolución mecánica de problemas abstractos, lo cual se materializa a través de una especificación formal de la sintaxis y de la semántica. De esta manera surgieron los primeros modelos de computación, en la década de los treinta del pasado siglo.

La construcción de las primeras máquinas de propósito general (de soporte electromecánico y, posteriormente, de soporte electrónico), entre 1943 y 1946, propició el nacimiento de la teoría de la complejidad computacional. Informalmente, se puede decir que el objetivo de esta teoría consiste en determinar qué problemas abstractos pueden ser resueltos por máquinas reales (en tiempo «razonable») para instancias de esos problemas cuyo tamaño sea *suficientemente grande*. Así pues, la teoría de la complejidad computacional puede ser entendida, en cierto sentido, como una contrapartida «práctica» al desarrollo teórico de los modelos de computación (*computabilidad teórica versus computabilidad práctica*). La segunda sección está dedicada a contextualizar el nacimiento de la teoría citada y a describir algunos hechos y resultados de especial relevancia, relacionados con la misma.

2.1. Teoría de la computabilidad

En la legítima aspiración por mejorar su calidad de vida, el ser humano ha tenido que enfrentarse, constantemente, a una serie de problemas concretos que surgían en el devenir cotidiano, y cuya resolución parecía importante para materializar dicho anhelo. A la vista de la dificultad creciente de las cuestiones que se iban planteando, se hizo imprescindible explorar nuevos métodos de resolución que propiciaran el uso de *asistentes* para ayudar al hombre en la realización de tareas tediosas que solían estar asociadas a la resolución de muchos problemas. Para ello, esos métodos deberían estar organizados en una serie de *tareas básicas* de tal manera que su ejecución, debidamente secuenciada, proporcionara la solución correcta del problema. Así, cuando se encontraba una solución de un problema conforme a la estructura citada, se decía que el problema había sido resuelto mediante un *procedimiento mecánico*. La ventaja de esta aproximación consistía en que cualquier *entidad* que tuviese la capacidad de realizar ciertas tareas, sería idónea para abordar la resolución de aquellos problemas para los que se pudieran diseñar procedimientos mecánicos descritos a través de tareas que la propia entidad fuera capaz de materializar. Por tanto, en este contexto, existe la posibilidad de que una entidad pueda resolver algunos problemas sin necesidad de que conozca detalles acerca de ellos. Obviamente, esa entidad debería disponer de ciertos *canales de comunicación* a fin de poder recibir unos procedimientos mecánicos y *actuar* en consecuencia.

En los tiempos de Euclides, entre el año 400 y el año 300 a. C., comenzaron a utilizarse métodos de resolución de problemas basados en la idea antes descrita. El primer ejemplo importante fue el famoso *algoritmo de Euclides* que resuelve el problema de hallar el máximo común divisor de dos números enteros arbitrarios. De esa forma surgieron las primeras soluciones de problemas abstractos descritas a través de procedimientos mecánicos. La aparición de los primeros métodos formales de razonamiento (posiblemente en Mesopotamia, y mejorados sustancialmente en la civilización griega) proporcionó una herramienta interesante para poder expresar, de forma algo más precisa, el concepto intuitivo de *procedimiento mecánico* o *algoritmo*.

Es usual encontrar en un diccionario el término *algoritmo* como sinónimo de *cualquier método especial de resolución de un cierto tipo de problemas*. La palabra *algoritmo* debe su nombre al autor persa Abú Jáfar Mohammed ibn al-Khowarizmi, que escribió un text en el año 825 d. C. en el que recogía una serie de procedimientos mecánicos para el álgebra; en particular una serie de reglas que, secuenciadas en un determinado orden, permitían realizar las operaciones con números decimales. No obstante, debería pasar mucho tiempo

hasta que G.W. Leibniz, hacia finales del siglo XVII, planteara 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*).

Todo parece indicar que era una creencia generalizada entre la comunidad científica el hecho de que cualquier problema podía ser resuelto de manera mecánica. No obstante, hacia finales del siglo XIX, se habían contabilizado un gran número de problemas, algunos de ellos de mucha envergadura, de los cuales no se conocían ninguna solución del tipo citado. Ello provocó que algunos científicos se plantearan la posibilidad de que existieran problemas que carecieran de soluciones mecánicas. Llegado este punto, se produjo una reflexión interesante acerca de la asimetría existente entre proporcionar una respuesta afirmativa o una respuesta negativa a la siguiente cuestión:

«Dado un problema abstracto, determinar si es resoluble mecánicamente»

Para responder afirmativamente, es decir, para afirmar que el *problema* es *decidible*, se suele ser *poco exigente* ya que basta presentar un procedimiento mecánico (que satisfaga los requisitos exigidos de manera intuitiva) para aceptar la respuesta. Por contra, para afirmar que el *problema* es *indecidible*, es decir, para asegurar que *ningún* procedimiento mecánico resuelve un cierto problema, es imprescindible disponer de una definición rigurosa que permita determinar, con total exactitud, la clase de todos los procedimientos mecánicos. Y esto ya es un asunto de bastante más calado. En definitiva, hacia finales del siglo XIX se planteó la necesidad de definir en un contexto matemático, la idea intuitiva de procedimiento mecánico. Más aún, cualquier definición formal de dicho concepto debía satisfacer un requisito fundamental: todos los procedimientos que, hasta la presente, habían sido considerados como mecánicos, desde un punto de vista intuitivo, deberían ser casos particulares de los procedimientos mecánicos acordes con dicha definición matemática.

En este contexto histórico surgen las tendencias *logicista* y *formalista* dentro de las Matemáticas, con B. Russell y D. Hilbert al frente. Esas escuelas usaban la metodología de la *lógica matemática*, en la que juega un papel preponderante los *lenguajes formales* que, en cierto sentido, se pueden considerar como una materialización de los lenguajes universales por los que suspiraba Leibniz. En la escuela formalista comienza a gestarse el concepto de *computabilidad efectiva* y uno de sus principales objetivos consistía en reducir todas las Matemáticas a la manipulación formal de símbolos lo cual, en definitiva, no es más que una forma de *computación*.

El *método axiomático* permitía describir el comportamiento de una *teoría* o *sistema* y constaba de tres ingredientes básicos: un *lenguaje* formal, unos *axiomas* y unas *reglas de inferencia*. El lenguaje permite describir los objetos que se van a estudiar, así como las propiedades relativas a dichos objetos. Los axiomas son asertos elementales, expresados en el lenguaje, que describen ciertas propiedades básicas de la teoría. Las reglas de inferencia son una especie de reglas de juego que permiten obtener nuevos enunciados en la teoría, denominados *teoremas*, a partir de los axiomas. De esta manera, en una teoría descrita a través de un sistema axiomático, se pueden generar nuevos resultados (es decir, teoremas de la teoría) tomando como punto de partida los axiomas y realizando un número finito de operaciones elementales (que resultan directamente de las reglas de inferencia). Con otras palabras, los teoremas de una teoría axiomática se pueden obtener, propiamente, a partir de una serie de procesos que pueden ser calificados de *mecánicos*.

A principios del siglo XX, los matemáticos estaban a punto de realizar una serie de descubrimientos de especial relevancia. D. Hilbert dio un paso al frente y se erigió, en cierto sentido, como portavoz de la comunidad científico-matemática, al concretar algunas de las tareas pendientes que él consideraba trascendentes para el devenir del nuevo siglo. En el Congreso Internacional de Matemáticos celebrado en París en 1900, Hilbert hizo pública una famosa lista de 23 problemas (que fueron enumerados) cuya resolución, en su opinión, marcarían el futuro de la ciencia, en general, y de las Matemáticas, en particular. Entre esos problemas, sólo uno estaba relacionado con la resolubilidad mecánica, el *problema décimo* (determinar si existe un procedimiento mecánico tal que, dada una ecuación diofántica arbitraria con coeficientes enteros, *decida* si existe alguna solución entera). Posteriormente, en el Congreso Internacional de Matemáticos celebrado en Bolonia, en 1928, Hilbert planteó a la comunidad científica tres cuestiones cruciales:

1. *Completitud de las matemáticas*: determinar si, dado cualquier aserto matemático, es posible hallar una demostración de éste o una demostración de su negación.

2. *Consistencia de las matemáticas*: determinar si, dado cualquier aserto matemático, es imposible hallar una demostración de éste y otra demostración de su negación.

3. *Decidibilidad de las matemáticas* (*Entscheidungsproblem*): determinar si existe algún procedimiento mecánico que nos permita, dado cualquier aserto matemático, *decidir* si existe o no una demostración del mismo.

Desde el punto de vista matemático, el ambiente del primer tercio del siglo XX estuvo bastante condicionado por el *problema décimo* y el *Entscheidungsproblem*: dos problemas importantes relacionados con la decidibilidad (es decir, con procedimientos mecánicos). Hilbert estaba convencido de que no existía límite para la capacidad de la inteligencia humana y de que las Matemáticas podrían ser descritas a través de un sistema axiomático, de tal manera que bastaba encontrar el lenguaje, los axiomas y las reglas de inferencia adecuadas para deducir en él los enunciados; es decir, él era optimista en relación con las dos primeras cuestiones formuladas en Bolonia. Es interesante hacer notar que las ideas de Hilbert que subyacen en esas cuestiones, representan la culminación de dos mil años de tradición matemática en donde las referencias obligadas eran el *método axiomático* de Euclides, el proyecto de Leibniz y Boole de crear una *lógica simbólica* y, más recientemente, los *Principia Matematica* de Russell y Whitehead. Hilbert pretendía clarificar de una vez por todas los métodos del razonamiento matemático y propugnó un *programa* que consistía en probar, mediante *métodos finitistas*, la existencia de un sistema axiomático formal que abarcara todas las Matemáticas, desde la aritmética elemental hasta el álgebra, pasando por la geometría, el cálculo, etc. Dicho sistema debería cumplir una serie de requisitos indispensables para los matemáticos, empezando por la *consistencia* y la *completitud* del mismo, y que viene a significar que el sistema sea capaz de probar la *la verdad, toda la verdad y nada más que la verdad*.

Hacia finales de 1930, en un congreso celebrado en la ciudad de Königsberg (actualmente, Kaliningrado), K. Gödel expuso su primer resultado de incompletitud (*si un conjunto finito de axiomas es capaz de deducir la aritmética de los números naturales y, además, es consistente, entonces no puede ser completo*). En ese evento, participaba J. von Neumann como matemático de reconocido prestigio en representación de la escuela formalista y «portavoz» de las ideas de Hilbert. Al parecer, von Neumann quedó impresionado de la posible trascendencia del resultado presentado por Gödel, hasta el punto de que, al mes siguiente, le envió una carta con una prueba de lo que sería el segundo teorema de incompletitud, resultado que el propio Gödel había obtenido previamente y de manera independiente. De los resultados de incompletitud se deducía que las dos primeras cuestiones planteadas por Hilbert acerca de la completitud y la consistencia de las Matemáticas, tenían una respuesta negativa. Más aún, von Neumann había deducido, a partir del resultado expuesto por Gödel, que el programa de Hilbert era irrealizable. Por cierto, no deja de ser un tanto sorprendente que, con los resultados de incompletitud encima de la mesa, ni Gödel ni von Neumann resolveieran el problema de la decidibilidad de las Matemáticas (el *Entscheidungsproblem*). No obstante, todo parece in-

dicar que ni siquiera llegaron a abordarlo en profundidad debido a que, tanto uno como otro, tenían su mente puesta en otras cuestiones que consideraban más interesantes.

Los trabajos de K. Gödel, A. Church, S. Kleene y A. Turing entre 1931 y 1936, proporcionaron, de manera independiente, las primeras definiciones matemáticas de procedimiento mecánico, dando lugar a los primeros modelos formales de computación. Concretamente, en 1931, K. Gödel definió el concepto de relación recursiva e introdujo la clase de funciones que denominó *recursivas* (y que hoy se conocen por el nombre de *funciones primitivas recursivas*). Posteriormente, en 1934 el propio Gödel extendió la clase anterior a las funciones *general recursivas* (y que hoy se conocen por el nombre de *funciones recursivas*). A su vez, entre 1931 y 1936, A. Church y su discípulo S. Kleene desarrollaron el concepto de λ-cálculo relacionándolo directamente con el de función computable. Por su parte, en 1036, A. Turing utilizó por primera vez el concepto abstracto de *máquina* para formalizar la noción de algoritmo y, por tanto, el concepto de función computable. Al introducir el concepto de *computabilidad* a través de *máquinas abstractas*, Turing trató de desarrollar un sistema teórico capaz de modelizar cualquier procedimiento que pudiera ser interpretado intuitivamente como *proceso de cálculo*. Específicamente, trató de diseñar un *modelo de cómputo* en su sentido más literal: imaginó a un *calculador-humano*, a una persona con papel y lápiz, resolviendo un determinado problema e intentó capturar la esencia del proceso, centrándose en los pasos más simples en los que se podía descomponer la tarea que estaba realizando. En este contexto, se puede asegurar que una idea central de Turing consistió en considerar a las máquinas como *manipuladoras de símbolos* (*symbol manipulator*) en lugar de dispositivos *devoradores de números* (*number cruncher*) [19, 46]

2.1.1. Decidibilidad e indecibilidad de problemas

En unas conferencias impartidas en 1934, en el *Institute for Advanced Study* de Princeton, Gödel afirmó que las funciones primitivas recursivas verificaban una propiedad importante: para cada conjunto de valores de sus variables, el valor de la misma podría ser calculado mediante un proceso finito, añadiendo que *«el recíproco parece ser cierto si se considera el tipo más general de recursión»*. Además, añadió que ese resultado no podía ser probado debido a que la noción de «computación finita» no había sido definida matemáticamente y, por tanto, el aserto debería entenderse como un *principio heurístico*. Todo parece indicar que A. Church conocía estas ideas de Gödel cuando, el 19 de

abril de 1935, hizo pública su famosas tesis en una breve intervención que tuvo en la reunión de la *American Mathematical Society* [9], en la que se expresaba que toda función *efectivamente computable* tenía que ser recursiva. Hay que hacer hincapié en el hecho de que esta tesis relaciona dos conceptos de naturaleza completamente distinta: por una parte, la existencia de un procedimiento mecánico que resuelve un problema (concepto informal) y, por otra, la existencia de una función recursiva que permite resolver dicho problema (concepto formal). En consecuencia, no tiene sentido alguno tratar de buscar una prueba matemática de dicha tesis. Por ello, en esta ocasión, el término *tesis* parece ser poco afortunado.

En abril de 1936, A. Church estableció el primer ejemplo de un problema (relacionado con la lógica de primer orden) que es irresoluble mecánicamente (a través de funciones λ-calculables), respondiendo negativamente a la tercera cuestión planteada por Hilbert (el *Entscheidungsproblem*). En mayo de 1936, A. Turing envió un artículo a la revista *Proceedings of the London Mathematical Society* para ser sometido al proceso de revisión con vista a su posterior publicación. En ese trabajo, Turing proporcionaba, en primer lugar, una prueba de la irresolubilidad del *problema de la parada* (*dada una máquina de Turing, M, y un dato de entrada de la máquina, u, determinar si M para sobre u*) a través de las máquinas que él había introducido. En segundo lugar, a partir de ese resultado y usando un novedoso proceso de *reducibilidad*, estableció la indecibilidad del problema de la lógica de primer orden que había sido analizado previamente por Church. En tercer lugar, Turing estableció la equivalencia de su modelo y el de las funciones λ-calculables para, finalmente, anunciar la equivalencia entre la clase de funciones computables por sus máquinas y la clase de funciones recursivas (resultado que publicaría en 1937 [62]). De esta manera, Turing demostró que los tres modelos de computación introducidos formalmente y de manera independiente, eran equivalentes, en el sentido de que todo aquello que es calculable o resoluble en uno de esos modelos lo es, también, en cualquiera de los otros dos.

A finales de 1936, Turing formuló una tesis similar a la de Church en términos de que toda función computable en sentido intuitivo tenía que ser computable por una de sus máquinas. Teniendo presente que la clase de funciones recursivas y la clase de funciones computables por máquinas de Turing son iguales, resulta que las tesis de Church y de Turing son «equivalentes». Debido, quizás, a esta tesis, los tres modelos a los que nos hemos referido, reciben el calificativo de *modelos universales* o *modelos computacionalmente completos*, capturando la idea intuitiva de tener la «mayor potencia computacional posible».

2.2. Teoría de la complejidad computacional

A lo largo de la historia, el hombre ha tratado de diseñar y construir artilugios que le asistieran en la resolución de problemas; específicamente, en procesos que requerían de la realización de tareas más o menos tediosas. De esta forma aparecieron el ábaco (en torno al año 1000 a. C.), la anticítera (año 87 a. C), las tablas de Néper (1617), la máquina calculadora de Pascal (1642-1653), la máquina de Leibniz (1670-1694), el telar de Jacquard (1801-1805), y la máquina de diferencias de Babbage (1821), entre otras muchas. Todas ellas son *máquinas de propósito específico*, es decir, dispositivos que habían sido diseñados para asistir al hombre en la resolución de un tipo concreto de problemas y que, por tanto, funcionaban siempre, esencialmente, «de la misma manera».

A diferencia de las máquinas de propósito específico, las *máquinas de propósito general* son dispositivos capaces de abordar la resolución de cualquier problema abstracto que tenga alguna solución mecánica, de acuerdo con la idea intuitiva de este término. Para ello, estas máquinas deberían tener la capacidad de adaptar, en cierto sentido, su funcionamiento interno acorde con una serie de «instrucciones» recibidas desde el «exterior» (con otras palabras, la máquina debería ser *programable*). Alrededor de 1837, Charles Babbage empezó a trabajar en el diseño de una *máquina analítica* cuya idea fue perfilando hasta el fin de sus días. Se trataba de una máquina que podía ser programada mediante tarjetas perforadas, a modo y manera de las usadas en los telares de Jacquard, de tal manera que el programa pudiera ser almacenado en una memoria. Así pues, la propia máquina podía alterar su funcionamiento acorde con los «dictados» de un tal programa, a fin de ejecutar una serie de tareas o instrucciones básicas en un orden, determinao a priori. Era la primera idea del concepto de *máquina de propósito general*. Curiosamente, por aquella época (en 1847), G. Boole publicó un artículo en el que presentaba su ya famosa «álgebra» [8] que constituye uno de los pilares básicos de las actuales máquinas de propósito general. La máquina analítica no pudo ser construida en esa época debido a que no se disponía de la tecnología y los medios adecuados. Parece ser que Lady Ada Lovelace diseñó los primeros programas para ser implementados en la máquina analítica y, por ello, se le conoce como la primera *programadora* de la historia.

Las primeras máquinas de propósito general fueron construidas entre 1943 y 1944 (Colossus Mark I y Mark II), eran de tipo electromecánico y totalmente automáticas. Fue la culminación de un proyecto del gobierno británico, desarrollado durante la segunda guerra mundial, encaminado a descifrar los

mensajes de los alemanes cifrados por la máquina *Enigma*. En ese proyecto intervino de manera muy activa A. Turing. Por su parte, las ideas de Babbage inspiraron a J. von Neumann y a sus discípulos, J.P. Eckert y J.W. Mauchly, para diseñar la denominada *arquitectura de von Neumann*, que daría lugar a la construcción, en 1946, de la que es considerada como la primera máquina electrónica *programable* de propósito general: el ENIAC (*Electronic Numerical Integrator And Computer*).

La aparición de nuevos modelos de computación llevaba implícito la exigencia de desarrollar dispositivos que, de la manera más fidedigna posible, implementara los modelos a través de máquinas *reales*. Entonces surge la siguiente cuestión: ¿qué problemas concretos pueden ser resueltos en esas máquinas reales? Para responder a esa pregunta es necesario disponer de unas herramientas que permitan cuantificar la mínima cantidad de *recursos* computacionales que necesita cualquier solución algorítmica de un problema abstracto. De acuerdo con el *principio de invariancia* (dadas dos máquinas reales distintas, existe una constante numérica asociada a las mismas tal que el tiempo de ejecución de dos implementaciones de un mismo algoritmo en esas máquinas, difieren en esa constante multiplicativa) resulta que el cálculo teórico de los recursos es fundamental para hallar una medida de la *computabilidad práctica* del problema.

En la década de los cincuenta del pasado siglo se desarrollaron los primeros *lenguajes de programación*, *traductores de lenguajes* y *sistemas operativos*. La potencia de los ordenadores en esa época estaba muy limitada por la excesiva lentitud de los procesadores y la escasa memoria de la que disponían para almacenar la información. Por ello, empezaron a desarrollarse teorías cuyo objetivo era explorar el uso eficiente de los ordenadores, lo que conlleva de alguna manera el estudio de la *complejidad intrínseca* de problemas abstractos. En la década de los sesenta se elaboraron los primeros cimientos de la *teoría de la complejidad computacional*, con la clasificación de lenguajes y funciones (debidas a J. Hartmanis, P.M. Lewis y R.W. Stearns [25, 61]) en función del tiempo y del espacio necesario para su generación o cálculo. Así mismo, se desarrollaron métodos de análisis para estudiar la eficiencia de los algoritmos y las estructuras de datos usadas en los mismos, la expresividad de los lenguajes formales, la capacidad computacional de las arquitecturas de los ordenadores, y la clasificación de problemas según la cantidad de recursos necesarios para su resolución.

A partir de ese momento se hizo imprescindible el análisis del tiempo y el espacio que un algoritmo necesitaba para ser ejecutado en una máquina de propósito general. Dicho estudio requirió el uso de técnicas matemáticas

(inducción, ecuaciones de recurrencia, notaciones asintóticas, manipulación de sumas finitas, etc.) que, además, permitan un análisis comparativo de distintas soluciones algorítmicas de un mismo problema. De esta manera se plantea una nueva cuestión:

«Dado un problema abstracto resoluble mecánicamente, hallar el mejor algoritmo que lo resuelva»

El concepto de *mejor* solución mecánica de un problema abstracto estará referido a una cierta *medida de complejidad* que cuantifique los recursos computacionales. Pues bien, un protocolo para hallar un *algoritmo óptimo*, respecto de una medida de complejidad, que resuelve un determinado problema abstracto podría ser el siguiente:

1. Determinar una cota inferior asintótica de la cantidad de recursos (respecto de la medida considerada) que *cualquier* algoritmo que resuelva dicho problema, necesita para su ejecución.

2. Hallar un algoritmo que resuelva el problema y, además, la cantidad de recursos que utiliza es del orden exacto de la cota inferior antes obtenida.

Si de un cierto problema abstracto se conoce un *algoritmo óptimo* que lo resuelve, entonces la cantidad de recursos necesarios para su ejecución proporcionará, de manera natural, la *complejidad computacional inherente* a dicho problema. Como es fácilmente imaginable, la tarea de calcular un *algoritmo óptimo* que resuelva un problema suele ser ardua y complicada, pero lo más sorprendente es que, en algunos casos, dicha tarea puede ser *imposible*. En efecto, si las medidass de complejidad que se consideran para cuantificar los recursos computacionales satisfacen unos requisitos mínimos (por ejemplo, los *axiomas de Blum* [6]) entonces, para cualquier medida de complejidad, existe algún problema resoluble algorítmicamente que carece de algoritmo óptimo, respecto de dicha medida (*teorema de aceleración*). En consecuencia, no se puede definir de manera *individual* el concepto de complejidad computacional de un problema a partir de la complejidad *inherente* al mismo, ya que siempre existiría, al menos, un problema abstracto al que no se le podría asignar complejidad alguna de acuerdo con esa definición. La alternativa que se considera es el estudio de la complejidad de problemas abstractos de una manera *global*; es decir, a través del análisis de la complejidad computacional de conjuntos de problemas, dando lugar a las denominadas *clases de complejidad*. Los ingredientes necesarios para definir una tal clase son los siguientes:

(a) Un *modelo de computación* que proporcione los dispositivos sobre los que se van a resolver los problemas.

(b) Un *modo de computación* que precise el concepto de *aceptación de un dato de entrada* (y, por tanto, fije el significado de resolución de un problema).

(c) Una *medida de complejidad* que permita cuantificar los recursos usados por los dispositivos computacionales en la resolución de problemas.

(d) Una *función* total computable entre números naturales que sirva de *cota superior* de los recursos usados.

Así por ejemplo, podríamos considerar como *modelo* de computación las máquinas de Turing, como *modo* de computación el ordinario o determinista, como *medida* de complejidad el tiempo y como *cota* una cierta familia de funciones distinguidas entre números naturales (por ejemplo, las logarítmicas, las polinomiales o las exponenciales) que actuarían de cotas superiores de los tiempos de ejecución que necesitan los procedimientos mecánicos, en relación con el tamaño de las instancias que describen el problema.

2.2.1. Tratabilidad e intratabilidad de problemas

La *teoría de la complejidad computacional* proporciona herramientas para medir el grado de dificultad de la resolución mecánica de problemas abstractos, a la vez, en términos absolutos (complejidad inherente de un problema) y en términos comparativos con otros problemas (clases de complejidad computacional). El objetivo fundamental de dicha teoría es la clasificación de problemas mediante la *resolubilidad algorítmica práctica* de los mismos; es decir, problemas que poseen soluciones mecánicas que pueden ser ejecutadas en máquinas reales para instancias del problema de tamaño *suficientemente grande*. Para ello, se define el concepto de *eficiencia* o resolubilidad práctica a través del concepto de *computabilidad en tiempo polinomial*. Este concepto fue introducido de manera explícita en la década de los sesenta por A. Cobham[10] y J. Edmonds [14], si bien J. von Neumann [69] había establecido en 1953 una distinción entre algoritmos que se ejecutan en tiempo polinomial y los que se ejecutan en tiempo exponencial. Además, Edmonds llamó *buenos algoritmos* (respecto del tiempo como medida de complejidad) a los que se ejecutan en tiempo polinomial y los consideró como *algoritmos tratables* o *manejables* por máquinas reales. Así, un *algoritmo* se dirá que es *eficiente* respecto de una

medida de complejidad, si la cantidad de recursos (expresados en la medida prefijada) necesarios para su ejecución está acotada, en el caso peor, por un polinomio en el tamaño del dato de entrada. De esta manera, cuando se usa el tiempo como medida de complejidad, se habla de una frontera entre la resolubilidad práctica (*tratabilidad*) y la irresolubilidad algorítmica práctica (*intratabilidad*) de problemas abstractos. Llegado este punto podríamos preguntarnos por qué se consideran a las funciones polinómicas para establecer dicha frontera. El motivo fundamental radica en que, por una parte, dicha clase de funciones es estable por ciertas operaciones importantes (suma, producto y composición de funciones) y en que, por otra, el crecimiento de las funciones polinónicas suele ser relativamente *lento*. Por su parte, los problemas *comptuacionalmente intratables* serán aquellos que no pueden ser resueltos por máquinas reales para instancias de tamaño razonablemente grande.

La *teoría de la complejidad computacional* juega un papel fundamental en la *criptografía* moderna (*public-key cryptography*). Actualmente en Internet existe una gran cantidad de información de tipo confidencial, se realizan transacciones comerciales que mueven una cantidad ingente de dinero, etc. La seguridad en la red depende básicamente de la complejidad computacional inherente a problemas como el de la *factorización entera* (base del sistema de cifrado RSA) o el *descifrado de cadenas codificadas por el sistema DES* (Data Encryption Standard), sistema que cifra textos planos a través de 64 bits usando 56 símbolos claves. Un ataque convencional sobre un texto cifrado por el sistema DES realizado mediante búsqueda exhaustiva, a través de un ordenador que es capaz de realizar un millón de operaciones por segundo, tardaría unos mil años aproximadamente. En 1995, D. Boneh, Ch. Dunworth y R.J. Lipton [7] estimaron que un cifrado DES podía ser decodificado por un ordenador molecular en unos cuatro meses.

2.2.2. Computaciones no deterministas

Una de las características fundamentales de las máquinas de Turing *ordinarias* (denominadas *deterministas*) es que, para cada dato de entrada, existe una única computación de la máquina y, por tanto, si ésta es de parada entonces devuelve un resultado que estará asociado, de manera unívoca, a dicha entrada. Por tanto, en esas máquinas, para cada configuración de la misma que no sea de parada, existe una única configuración siguiente. Por su parte, en una máquina de Turing *no determinista* pueden existir configuraciones asociadas a una cierta entrada que no sean de parada y, además, posean más de una configuración siguiente. En tales circunstancias, para algunos datos de

entrada, pueden existir varias computaciones, de las cuales algunas pueden ser de parada y otras no; más aún, aquellas computaciones obtenidas a partir del mismo dato de entrada y que, además, sean de parada, no tienen por qué devolver el mismo resultado. De alguna forma, podemos decir que las máquinas de Turing no deterministas *no son fiables*, en el sentido de que el resultado que proporcione una computación (en caso de que sea de parada) puede no ser una información relevante. Así pues, se puede considerar la existencia de otro tipo singular de procedimientos mecánicos en los que, a partir de un conjunto de instrucciones que son *ejecutables* en un determinado momento, el procedimiento puede *seleccionar*cualquiera de ellas para proseguir su ejecución. En un cierto paso de transición no determinista puede suceder que una configuración posea más de una configuración siguiente y, en consecuencia, a partir de un cierto dato de entrada de un problema, un procedimiento mecánico no determinista puede proporcionar, de manera independiente, muchas computaciones distintas.

Es interesante hacer notar que: (a) toda máquina de Turing determinista se puede considerar como un caso particular de una máquina de Turing no determinista; (b) toda máquina de Turing no determinista puede ser simulada por una máquina de Turing determinista, si bien el coste de dicha simulación es de tipo exponencial. Por tanto, las máquinas de Turing no deterministas no añaden más potencia computacional a las máquinas de Turing deterministas, como no podía ser de otra manera ya que éstas son máquinas universales. No obstante, podríamos preguntarnos si añade añgo realmente nuevo el modo no determinista respecto del modo ordinario o determinista. Esta cuestión está relacionada, en cierto modo, con el famoso problema **P** *versus* **NP**.

Capítulo 3

El problema P versus NP

El problema **P** *versus* **NP** consiste en determinar si las clases de complejidad computacional **P** y **NP** (clase de los problemas de decisión que son resolubles por máquinas de Turing deterministas o por máquinas de Turing no deterministas, respectivamente) son o no iguales. El objetivo de este capítulo es introducir los conceptos básicos de la *teoría de la computabilidad* y la *teoría de la complejidad computacional* que permitan vislumbrar el verdadero significado y la trascendencia del problema antes citado.

El capítulo está estructurado como sigue: en la primera sección se describe dicho problema y se analizan algunas repercusiones interesantes que su resolución tendría en diferentes ámbitos de nuestra vida cotidiana. La segunda sección está dedicada a presentar los conceptos de *eficiencia* y *presumible eficiencia* de modelos de computación. En la última sección se presenta una nueva metodología para atacar el problema **P** *versus* **NP** descrito a través de ingredientes sintácticos o semánticos de modelos de computación desarrollados en el marco de un cierto paradigma computacional.

3.1. Descripción del problema

La clase de complejidad computacional **P** es el conjunto de todos los problemas de decisión que son resolubles por máquinas de Turing deterministas que trabajan en tiempo polinomial. Es decir, **P** es la clase de complejidad de los problemas de decisión que son *tratables en modo determinista*, respecto del tiempo. Por su parte, la clase de complejidad computacional **NP** es el conjunto de todos los problemas de decisión que son resolubles por máquinas de Turing no deterministas que trabajan en tiempo polinomial. Así pues, en cierto sentido se podría afirmar que **NP** es la clase de complejidad de los problemas de

decisión que son *tratables en modo no determinista*.

Teniendo presente que toda máquina de Turing determinista se puede considerar como un caso particular de una máquina de Turing no determinista, resulta que $\mathbf{P} \subseteq \mathbf{NP}$. El problema **P** *versus* **NP** consiste, simplemente, en lo siguiente:

«Determinar si las clases de complejidad computacional **P** *y* **NP** *son iguales»*

Este problema es, sin lugar a dudas, el más importante en las *Ciencias de la Computación* y uno de los más relevantes que tiene planteada la Ciencia, en general. Con el fin de «dar la bienvenida a las Matemáticas del nuevo milenio», el *Clay Mathematics Institute* (CMI) de Cambridge, Massachusetts (USA), seleccionó siene problemas de especial relevancia (entre los que se encuentra el problema **P** *versus* **NP**), ofreciendo una dotación de un millón de dólares para aquel que resolviera alguno de ellos. Hasta la presente, sólo uno de ellos (la conjetura de Poincaré) ha sido demostrado (en 2003 por G. Perelman que, por cierto, rechazó la dotación económica del CMI).

¿Dónde radica la importancia de dicho problema? De su propia formulación, es fácil deducir que se trata de un problema teórico, muy teórico, ya que consiste, básicamente, en determinar si dos clases de complejidad computacional, **P** y **NP**, son iguales. A pesar de lo cual, una respuesta a esa cuestión tendría unas repercusiones extraordinarias, especialmente desde el punto de vista económico.

3.1.1. Todo parece indicar...

Informalmente, el problema **P** *versus* **NP** consiste en decidir si es «más difícil» *hallar* una solución correcta de un problema abstracto que *comprobar* que una presunta solución del mismo es, realmente, una solución correcta. Desde el punto de vista intuitivo, todo parece indicar que *hallar*, buscar o encontrar una solución de un problema debería ser «más difícil» que *comprobar* si una presunta solución concreta del mismo es correcta. Si, por ejemplo, consideramos el problema de la *factorización entera* (dado un número natural que es producto de dos números primos, hallar su descomposición) entonces dado un tal número n parece más difícil hallar dos números primos p_1, p_2 cuyo producto sea n que, dados dos números primos q_1, q_2 comprobar si su producto es igual a n. De ahí que, mayoritariamente, la comunidad científica conjeture que eso, precisamente, es lo que debería verificarse; es decir, que $\mathbf{P} \neq \mathbf{NP}$, en cuyo caso, *hallar* una solución de un problema abstracto sería «estrictamente

más difícil» que *comprobar* la corrección de una presunta solución concreta del mismo.

Pues bien, para probar la citada conjetura, los candidatos idóneos para ser testigos de ese hecho (es decir, problemas de la clase **NP** que no sean tratables) serían aquellos problemas «más difíciles» de **NP**, en el sentido de que cualquier otro problema de la clase **NP** puede ser resuelto a través de él con un coste en tiempo adicional de tipo polinomial. Este tipo de problemas se denominan **NP**-*completos* y, en 1971, S.A. Cook [11] proporcionó el primer ejemplo: el problema `SAT` de la satisfactibilidad de la lógica proposicional. Un año después, teniendo presente que la reducibilidad en tiempo polinomial permite generar nuevos problemas **NP**-completos a partir de otros cuya **NP**-completitud ya ha sido establecida, R.M. Karp [23] da 24 ejemplos nuevos de problemas **NP**-completos (entre los que destacan el problema de recubrimiento de vértices, el problema del recubrimiento exacto, el problema del número cromático, el problema del ciclo Hamiltoniano y el problema del viajante de comercio). En la actualidad se conocen muchos problemas **NP**-completos de disciplinas tan diversas como lógica, teoría de números, teoría de grafos, investigación operativa, etc. En el texto de M.R. Garey y D.S. Johnson [16], lectura obligada para la iniciación en la teoría de la complejidad computacional, proporciona un catálogo exhaustivo de más de trescientos problemas **NP**-completos.

La *aproximación clásica* para abordar el problema **P** *versus* **NP** consiste en considerar un problema **NP**-completo, **uno solo**, e intentar probar que ese problema **NP**-completo no pertenece a la clase **P**, entonces, obviamente, habríamos demostrado que $\mathbf{P} \neq \mathbf{NP}$. Lo más interesante de esta aproximación es que si, por el contrario, fuéramos capaces de probar que ese problema **NP**-completo pertenece a la clase **P**, entonces se deduciría fácilmente que todos los problemas de la clase **NP** pertenecerían a la clase **P** y, en consecuencia, se habría certificado que $\mathbf{P} = \mathbf{NP}$. Por tanto, para resolver el problema **P** *versus* **NP**, acerca de la igualdad o no de las citadas clases de complejidad computacional (tanto en un sentido afirmativo como negativo), sería suficiente analizar **un solo** problema **NP**-completo. De ahí la importancia de disponer de problemas **NP**-completos pertenecientes a áreas de lo más dispares.

Lo relevante es que una respuesta a esa cuestión tendría unas repercusiones extraordinarias, especialmente desde el punto de vista económico. En efecto, si se probara que $\mathbf{P} \neq \mathbf{NP}$ entonces se confirmaría que los actuales sistemas de cifrado (RSA, DES, etc.) son seguros desde un punto de vista práctico y, por ende, todas las transacciones comerciales realizadas a diario vía electrónica, estarían libres de ataques que desvelaran su contenido. No obstante, en este supuesto podría suceder que todo problema **NP**-completo tuviese un algorit-

mo determinista de coste polinomial que trabaje *correctamente* sobre *muchas* entradas del problema. Con ello, el mundo de la criptografía no se tambalearía y, en cambio, se conseguiría algunos beneficios parciales del caso en que la respuesta fuese afirmativa. En esta línea, L. Levin [24] y R. Impagliazzo [22] desarrollaron una teoría de la completitud del caso promedio, en donde la cuestión $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$ es sustituida por esta otra: determinar si cada problema **NP**-completo puede ser resuelto, con una *razonable* distribución de probabilidad sobre sus entradas en tiempo polinomial en el caso promedio.

Es interesante hacer notar que, en 1975, R.E. Ladner demostró que si $\mathbf{P} \neq \mathbf{NP}$, entonces existirían *problemas intermedios*; es decir, problemas que ni serían tratables ni serían **NP**-completos.

3.1.2. Pero si $\mathbf{P} = \mathbf{NP}$...

Si, en cambio, se probara que $\mathbf{P} = \mathbf{NP}$, entonces resultaría que, por ejemplo, un algoritmo de orden cuadrático que resolviera una variante simple del problema `SAT` se podría usar para factorizar los números de 200 dígitos en algunos minutos. Por tanto, en este caso, no sólo se alimentaría la incertidumbre acerca de la fiabilidad de los sistemas criptográficos que son utilizados en la actualidad, sino que, además, según todos los indicios, una tal respuesta contraria a la creencia generalizada, debería venir «acompañada» de un mecanismo general que proporcionaría soluciones eficientes (y deterministas) para la mayoría de los problemas **NP**-completos. En particular, se habría puesto el caldo de cultivo apropiado para el diseño de un *decodificador universal*: un dispositivo que permitiría decodificar de manera eficiente, cualquier mensaje descrito mediante un sistema de cifrado. Teniendo presente que, actualmente, la tecnología permite acceder en tiempo real a casi todos los mensajes o comunicaciones de interés que puedan circular por el orbe, la implementación de un decodificador universal se convertiría en un «arma letal» para disponer de toda la información que pudiera ser interesante. Y ya se sabe aquello de que la *información es poder*. Por otra parte, en el hipotético caso de que se probara que $\mathbf{P} = \mathbf{NP}$, también existiría un efecto que podríamos denominar positivo: sería factible realizar demostraciones de manera automática y en «tiempo razonable» acerca de cuestiones abiertas de gran calado en Matemáticas que el hombre no ha sabido resolver hasta la fecha. En tal situación, habría una pequeña *pega*: cómo entender las demostraciones de esos problemas abiertos, suministradas por una máquina que implementara el espíritu subyacente en la prueba de que $\mathbf{P} = \mathbf{NP}$.

No obstante, en tal caso, también existirían consecuencias positivas. Por

ejemplo, sería posible diseñar programas que permitieran a un ordenador electrónico convencional encontrar demostraciones de teoremas que tengan pruebas de *longitud razonable* (ya que las pruebas formales pueden ser *reconocidas* en tiempo polinomial). Desgraciadamente, ocurriría que muchas de las pruebas no serían entendidas por los humanos. Pero bueno, entre otras consecuencias *positivas* de esa hipotética situación, muy probablemente se podrían conseguir seis millones de dólares que garantizan los premios CMI (con tal de conservar sigilosamente la prueba de que $\mathbf{P} = \mathbf{NP}$ durante el tiempo necesario para obtener las soluciones de los restantes seis problemas del milenio). Lo que no estaría nada mal.

3.2. Nueva metodología para atacar el problema P versus NP

Recordemos que un *paradigma de computación* proporciona un marco que permite definir una serie de modelos de computación que responden a unas determinadas especificaciones sintácticas y/o semánticas, fijadas de antemano.

La tratabilidad o computabilidad práctica de problemas abstractos trata de capturar la idea de la resolubilidad de problemas a través de máquinas reales (que capturen la semántica de una máquina de Turing determinista) cuya ejecución nos proporciona la solución del mismo para instancias de tamaño suficientemente grande. Las soluciones mecánicas de problemas abstractos en tiempo polinomial se denominan *soluciones eficientes*. Es interesante tener presente que en ciertos modelos de computación, problemas intratables podrían, eventualmente, ser resueltos en tiempo polinomial. Esto da lugar al concepto de *eficiencia* asociado a modelos de computación.

Definición 3.1. *Diremos que un modelo de computación es eficiente si tiene la capacidad de proporcionar soluciones de problemas intratables en tiempo polinomial.*

Es decir, los modelos de computación eficientes pueden resolver problemas intratables de manera eficiente. Por tanto, los modelos de computación no eficientes sólo pueden resolver en tiempo polinomial, problemas pertenecientes a la clase **P**. En particular, las máquinas de Turing deterministas son modelos no eficientes.

La comunidad científica está convencida de que $\mathbf{P} \neq \mathbf{NP}$ y, por ello, los problemas **NP**-completos (los más difíciles de la clase **NP**) son considerados como problemas *presuntamente intratables*. Un modelo de computación capaz

de proporcionar soluciones de problemas **NP**-completos en tiempo polinomial se denomina *presumiblemente eficiente*.

Definición 3.2. *Diremos que un modelo de computación es presumiblemente eficiente si tiene la capacidad de proporcionar soluciones de problemas* **NP***-completos en tiempo polinomial.*

Sean M_1 y M_2 dos modelos de computación de un cierto paradigma computacional. Diremos que M_1 es un *submodelo* de M_2 (o bien que M_2 es una *extensión* de M_1) si cada solución de un problema abstracto en M_1 es, también, una solución del mismo problema en M_2. Si M_2 es una extensión de M_1, entonces se puede suponer que el modelo M_2 es obtenido a partir del modelo M_1 añadiendo algunos ingredientes de tipo sintáctico o semántico. Por tanto, sería interesante estudiar qué tipo de soluciones de un problema abstracto en M_2 podrían servir para diseñar una solución del problema en M_1. De esta manera, estaríamos analizando el papel que juegan los ingredientes añadidos a M_1 para obtener M_2, en el diseño de soluciones de problemas abstractos.

Supongamos que: (a) M_1 es un modelo de computación no eficiente; (b) M_2 es un modelo de computación presumiblemente eficiente; y (c) M_2 es una extensión de M_1. Entonces se puede afirmar que pasar del modelo M_1 al modelo M_2 equivale a *pasar de lo tratable a lo presuntamente intratable*, en el sentido de que mientras en M_1 sólo podemos resolver problemas tratables, en M_2 es posible resolver eficientemente problemas **NP**-completos. Dicho con otras palabras, suponiendo que $\mathbf{P} \neq \mathbf{NP}$, el conjunto de ingredientes sintácticos o semánticos que se han de añadir al modelo M_1 para obtener el modelo M_2, se pueden considerar como una *frontera de la tratabilidad*. Este concepto se puede representar gráficamente como en la Figura 3.1.

De acuerdo con lo indicado anteriormente, cada frontera entre la tratabilidad y la presunta intratabilidad, expresada en términos de ingredientes sintácticos y/o semánticos de modelos de un paradigma computacional, proporciona una herramienta para atacar el problema **P** *versus* **NP**. En efecto:

- Para demostrar que $\mathbf{P} = \mathbf{NP}$, sería suficiente encontrar una solución en tiempo polinomial a un problema **NP**-completo en el modelo M_2 y «traducirla» a una solución en tiempo polinomial en el modelo M_1. Esto probaría que los ingredientes añadidos para obtener el modelo M_2 a partir del modelo M_1 no juegan un papel relevante en la descripción de esa solución.

- Para demostrar que $\mathbf{P} \neq \mathbf{NP}$, sería suficiente encontrar un problema **NP**-completo que no pueda ser resuelto de manera eficiente en el modelo M_1.

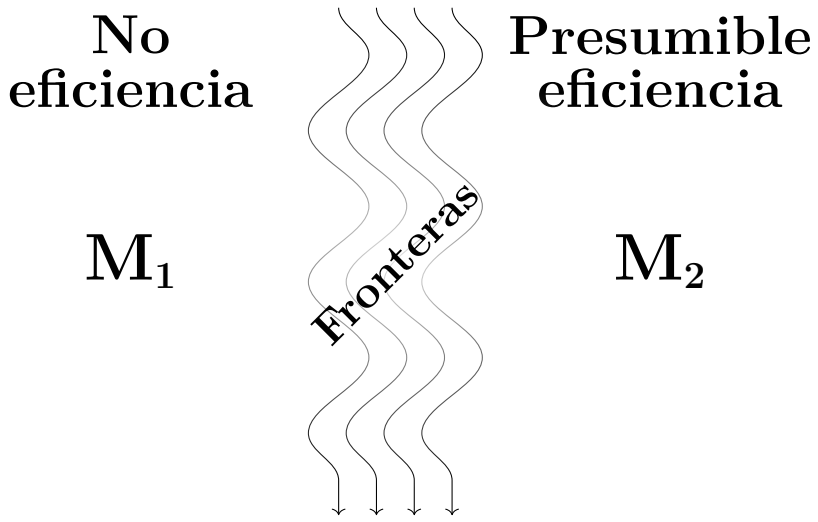


Figura 3.1: Fronteras de la tratabilidad entre los modelos M_1 y M_2

Esto probaría que los ingredientes añadidos para obtener el modelo M_2 a partir del modelo M_1 son cruciales para obtener la presumible eficiencia de M_2.

Capítulo 4

El paradigma de la computación celular con membranas

El objetivo de este capítulo consiste en introducir los conceptos básicos relativos al paradigma de la computación celular con membranas (*Membrane Computing*) que van a ser objeto de estudio en el presente trabajo. Específicamente, se van a presentar distintos modelos de dicho paradigma que trabajan a «modo de células» (*cell-like*) y otros que trabajan a «modo de tejidos» (*tissue-like*). Entre los del primer tipo se introducirán: (a) los sistemas P básicos de transición; (b) los sistemas P con membranas activas; (c) los sistemas P con membranas activas que carecen de polarizaciones; y (d) los sistemas P con reglas symport/antiport. Entre los del segundo tipo se introducirán: (a) los sistemas P básicos de tejidos; (b) los sistemas P de tejidos con división celular; (c) los sistemas P de tejidos con separación celular; y (d) los sistemas P de tejidos con reglas de comunicación evolutiva.

4.1. Introducción

El campo de investigación denominado *Computación Natural* es 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. Entre las áreas que se enmarcan dentro de la Computación Natural, destacan: (a) los *algoritmos genéticos*; (b) las *redes neuronales artificiales*; (c) la *computación molecular basada en ADN*; y (d) la *computación celular con membranas*. No obstante, conviene observar que no existe un consenso en la comunidad científica sobre la inclusión o no de la *computación cuántica* dentro del ámbito propio de la Computación Natural, si

bien parece que no se ajusta a la interpretación dada anteriormente acerca de esta disciplina.

Los principios básicos de los *algoritmos genéticos* fueron propuestos por J.H. Holland [21] en 1975, y están basados en el proceso genético de los organismos vivos a través del cual evolucionan, cuyo elemento fundamental es el principio de selección natural. El objetivo de estos procedimientos mecánicos consiste en encontrar una buena solución del problema objeto de estudio a partir de una gran cantidad de posibles soluciones candidatas. Para ello, se consideran poblaciones de genes de los individuos que codificarán posibles soluciones del problema y se analizará su evolución a lo largo del tiempo, de acuerdo con unas reglas de selección y de otros operadores tales como *recombinación* y *mutación*. Durante ese proceso, cada individuo va siendo procesado de forma dinámica recibiendo una medida de su adecuación o adaptación al entorno. Los procesos de selección centrarán su atención en los individuos mejor adaptados, pudiendo alterar los individuos inadaptados mediante procesos de recombinación o mutación. Si el algoritmo genético es diseñado correctamente, entonces la evolución de la población convergerá, teóricamente, a un individuo óptimamente adaptado.

Las *redes neuronales artificiales* están inspiradas en las interconexiones y el funcionamiento de las neuronas en el cerebro. Dichas redes surgieron originalmente como una simulación del sistema nervioso, formado por un conjunto de unidades (*neuronas*) conectadas entre sí, simulando a las *dendritas* y los *axones* en los sistemas nerviosos biológicos. El primer modelo de red neuronal artificial fue propuesto en 1943 por W.S. McCulloch y W. Pitts [31] como simulación de la actividad nerviosa y, poco después, surgieron nuevos modelos debidos a M. Minsky [32] y F. Rosenblatt [59], entre otros. El objetivo fundamental de las redes neuronales artificiales consiste en mejorar la potencia de cálculo de los dispositivos computacionales a partir de unidades sencillas conectadas entre sí, imitando el esquema de interconexiones que presentan las neuronas en el cerebro. En la evolución de una tal red, cada neurona recibe una cierta cantidad de datos numéricos (es decir, «información» que puede provenir de otras neuronas, denominadas pre-sinápticas) y produce un único valor numérico como salida (que puede ser enviado a otras neuronas, denominadas post-sinápticas). Este modelo ha resultado ser uno de los más efectivos en cuanto a problemas de interpretación de sensores del mundo real, tales como reconocimiento de lenguaje hablado o aprendizaje de estrategias para el control de robos. Las redes neuronales constituyen una interesante herra-

mienta de análisis estadístico que permite la construcción de un modelo de comportamiento a partir de una base de ejemplos de dicho comportamiento. Una red neuronal es, inicialmente, completamente *ignorante*, y tiene la capacidad de efectuar un aprendizaje partiendo de los ejemplos para transformarse o evolucionar, a través de una serie de modificaciones sucesivas, en un modelo susceptible de predecir el comportamiento futuro del sistema simulado. Esta capacidad para aprender todo aquello que tenga un cierto sentido (*aproximador universal*) ha sido establecida de manera rigurosa (*teorema de Kolmogorov*), por lo que las redes neuronales son consideradas hoy día como una herramienta de gran rigor cuyas bases teóricas han sido debidamente justificadas. Las redes neuronales destacan, pues, como técnica de aprendizaje automático por su capacidad de aprender una variedad considerable de funciones no lineales que, potencialmente, involucren un gran número de variables.

La *computación molecular basada en ADN* trata de utilizar efectivamente algunos medios físicos disponibles en la Naturaleza (específicamente, moléculas orgánicas: ADN, ARN, proteínas, etc.) para usarlos como sustrato físico sobre el que ejecutar procesos mecánicos. La idea de realizar computaciones a nivel molecular (como principio básico para producir una auténtica revolución en la miniaturización de las componentes físicas de una máquina de propósito general) fue propuesta por el premio Nobel R. Feynman a principio de la década de los sesenta del pasado siglo [15], y estudiada en 1973 por C. Bennet [4] al analizar la posibilidad de usar el ácido ribonucleico (ARN) como medio físico (reversible) para implementar computaciones, estableciendo un paralelismo entre el funcionamiento de la enzima polimerasa y la cabeza lectora/escritora de una máquina de Turing. Posteriormente, en [5] el propio Bennet dio una descripción teórica de un computador que usa moléculas tipo ARN para almacenar información y enzimas imaginarias para catalizar reacciones sobre dichas moléculas. En 1987, T. Head [20] describió un modelo teórico (el modelo *splicing*) que estaba basado en la manipulación de lenguajes de cadenas de ADN a través de la operación de recombinación. Sin embargo, no es hasta finales de 1994 cuando tiene lugar, propiamente, la primera implementación de una computación a nivel molecular. L. Adleman [1] llevó a cabo un experimento en el laboratorio que permitió resolver una instancia «modesta» (de tamaño pequeño) de un problema muy complejo desde el punto de vista computacional, el *problema del camino Hamiltoniano*, en su versión dirigida y con dos nodos distinguidos, realizando determinadas operaciones con moléculas de ADN en un laboratorio de biología molecular. El impacto del trabajo de Adleman fue de tal envergadura que promovió, indirectamente, el uso del término *Compu-*

tación Natural para una disciplina cuyos orígenes se remontan a la década de los cuarenta del pasado siglo. En la computación molecular se trató de rentabilizar la capacidad de las moléculas de ADN para implementar el paralelismo masivo y, al mismo tiempo, de usar dichas moléculas como soporte físico de la información, logrando así una extraordinaria densidad de almacenamiento de información (por ejemplo, un gramo de ADN en polvo ocupa un volumen aproximado de 1 cm^3 y es capaz de almacenar información equivalente a un billón de CDs convencionales).

La *computación celular con membranas* (*Membrane Computing*) 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. Fue introducida por Gh. Păun en octubre de 1998 [43] como un paradigma computacional que proporciona modelos de computación de tipo no determinista, distribuido y paralelo. Este es el marco donde se sitúa el presente trabajo.

Las membranas suelen jugar un papel relevante en este paradigma y son, propiamente, abstracciones de las membranas biológicas que aparecen, debidamente jerarquizadas, en el interior de la célula. Las membranas delimitan compartimentos que son susceptibles de contener objetos que, a su vez, son abstracciones de los compuestos químicos y codifican información. Los objetos pueden evolucionar a lo largo del tiempo a través de reglas de reescritura que son abstracciones de las reacciones químicas, proporcionando un flujo de información entre los diferentes compartimentos. Algunos modelos en este paradigma computacional se basan en una estructura similar a la de una célula (aproximación *cell-like*), en el sentido de tener una disposición jerárquica de membranas a modo de árbol enraizado. Otros modelos trabajan a modo de tejidos (aproximación *tissue-like*) basados en una estructura dispuesta como una red (materializada mediante un grafo dirigido) en el que distintas unidades de proceso (denominadas membranas, células o neuronas) intercambian información a través de los canales de comunicación dispuestos en la red. A lo largo de esta memoria se usará el término de *sistema de membranas* o *sistema P* para referirnos, genéricamente, a un dispositivo de este paradigma computacional. Una referencia clara puede ser la página web dedicada a la disciplina [74], además de los dos libros de referencia de la misma [57, 58].

Conviene resaltar el hecho de que, de manera un tanto sorprendente, existe un amplio consenso en la comunidad científica a la hora de precisar cuándo una entidad puede ser considerada como un organismo vivo. La propiedad ca-

racterística es su capacidad para realizar, de forma autónoma, los siguientes procesos: (a) *replicación del ADN*; (b) *fabricación* (*síntesis*) *de proteínas*; (c) *producción de energía*; y (d) *realización de procesos metabólicos*. De lo que se conoce hasta la fecha, la célula es la entidad más simple en la que se da el fenómeno de la vida. Es la unidad fundamental de todo organismo vivo y posee una estructura compleja y, a la vez, muy organizada. La actividad que ocurre en su interior puede ser considerada, sin lugar a dudas, como procesamiento de información: en las células existen unas *membranas biológicas* que delimitan *compartimentos* o *regiones* donde se localizan las interacciones entre las sustancias químicas presentes en ellas, así como el intercambio de sustancias entre compartimentos vecinos. Una membrana (denominada *plasmática* o membrana *piel*) separa el espacio interno de la célula (el *citoplasma*) protegiéndolo de su entorno y dándole entidad propia a la célula. Las restantes membranas proporcionan una estructura jerarquizada y ofrecen la adecuada protección al *núcleo*, que es el «depositario» de la información genética (una molécula de ADN que identifica, propiamente, la célula). Existen dos tipos de células:L 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 conteniendo en su interior la molécula de ADN que identifica, propiamente, la célula. Los dos tipos de células realizan de manera similar los procesos que son esenciales para la vida. A lo largo de este trabajo, los modelos de computación estarán inspirados en las células eucariotas.

En la Figura 4.1 se pueden apreciar con claridad las distintas partes que componen una célula eucariota.

Cada membrana biológica (retículo endoplasmático, mitocondria, aparato de Golgi, etc.) determina una serie de compartimentos que pueden considerarse como unidades básicas de procesos, 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*. El término «no convencional» es usado para significar que uno de los objetivos básicos de dicho paradigma computacional consiste en construir máquinas reales de propósito general que implementen la semántica del modelo y de tal manera que el soporte físico no sea el electrónico, debido a las limitaciones que derivan del uso de la tecnología electrónica a la hora de abordar la resolución de problemas de alta complejidad computacional.

Las ideas de Feynman a las que antes nos hemos referido, adquirieron especial relevancia a partir de 1983, cuando R. Churchhouse estableció las limita-

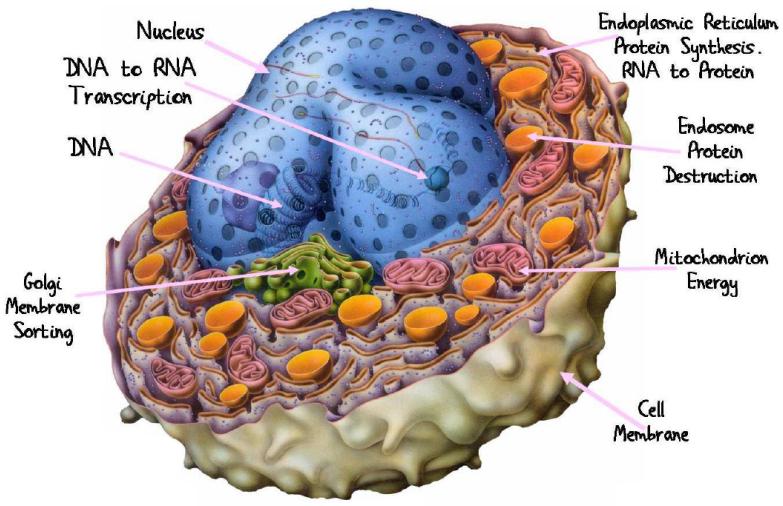


Figura 4.1: Célula eucariota

ciones físicas, tanto del tamaño como de la velocidad de cálculo, de los microprocesadores convencionales basados en la tecnología electrónica; es decir, en la manipulación electrónica del silicio. Este resultado, esperable por otra parte, adquirió tintes dramáticos cuando se comprobó, a partir de resultados de la teoría de la complejidad computacional, que esas limitaciones hacían imposible la resolución de instancias de problemas de la vida real que son especialmente importantes, a menos que sucediera un «milagro»; es decir, que se verificara que $\mathbf{P} = \mathbf{NP}$. La existencia de problemas concretos que para su resolución mecánica necesitaban una cantidad de recursos que excedía las capacidades de cualquier *dispositivo computacional convencional*, independientemente de cuál fuera el procedimiento utilizado para resolverlo, propició la búsqueda de alternativas que ofrecieran la posibilidad de resolver esos mismos problemas de manera *más eficiente*, permitiendo abordar instancias de tamaño *relativamente grande*, de problemas *intratables* o *presuntamente intratables*. Ante esa tesitura surgió la necesidad de buscar nuevos paradigmas computacionales que proporcionaran modelos con algunas singularidades y que, además, pudieran ser implementados mediante máquinas cuyo principal soporte físico no fuera el medio electrónico. De esta manera surgiría la posibilidad de construir máquinas programables de propósito general tales que la velocidad de cálculo y el tamaño de los microprocesadores superaran las barreras inherentes al soporte electrónico, lo que redundaría en una mejora cuantitativa en la resolución de instancias de problemas relevantes cuya complejidad computacional sea muy

elevada.

4.2. Sistemas de membranas que trabajan a modo de células

4.2.1. Sistemas P básicos de transición

Los sistemas P básicos de transición fueron introducidos en el artículo fundacional de la disciplina de *Membrane Computing* [43].

Definición 4.1. *Un sistema P básico de transición de grado* $q \geq 1$ *es una tupla de la forma* $\Pi = (\Gamma, \mu, \{\mathcal{M}_i, \mathcal{R}_i, \rho_i \mid 1 \leq i \leq q\}, i_{out})$*, en donde:*

- Γ *es un alfabeto finito.*
- μ *es un árbol enraizado cuyos nodos están biyectivamente etiquetados por* $1, \ldots, q$ *(la raíz del árbol está etiquetada por* 1*).*
- $\mathcal{M}_i, 1 \leq i \leq q$*, son multiconjuntos sobre* Γ*.*
- $\mathcal{R}_i, 1 \leq i \leq q$*, es un conjunto finito de reglas sobre* Γ *del tipo* $u \rightarrow v$*, siendo* $v = (v_1, here)(v_2, out)(v_3, in_j)\delta'$*, con* u, v_1, v_2, v_3 *multiconjuntos sobre* Γ *y* $\delta' \in \{\lambda, \delta\}$*.*
- $\rho_i, 1 \leq i \leq q$*, es un orden parcial estricto entre reglas de* $\mathcal{R}_i$*; es decir,* $\rho_i = \{(r, r') \mid r, r' \in \mathcal{R}_i\}$ *es una relación binaria antirreflexiva, simétrica y transitiva sobre* $\mathcal{R}_i$*.*
- $i_{out} \in \{0, 1, \ldots, q\}$*.*

Un sistema P básico de transición Π de grado $q \geq 1$ puede ser considerado como un conjunto de q membranas, situado en un determinado *entorno* externo. Las membranas están etiquetadas de manera biyectiva mediante elementos del conjunto $\{1, \ldots, q\}$ y, a su vez, están organizadas a través de una estructura jerárquica de membranas que delimitan regiones o compartimentos, materializada por un árbol enraizado μ cuya raíz es la *membrana* que se denomina *piel*, etiquetada por 1, y de tal manera que: (a) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ representan los multiconjuntos de objetos (símbolos de Γ) inicialmente situados en las q regiones determinadas por las membranas del sistema; (b) $\mathcal{R}_1, \ldots, \mathcal{R}_q$ son conjuntos finitos de reglas sobre Γ asociadas a las membranas del sistema; (c)

$\rho_1, \ldots, \rho_q$ son órdenes parciales estrictos entre reglas de $\mathcal{R}_1, \ldots, \mathcal{R}_q$, respectivamente, de tal manera que si $(r, r') \in \rho_i$ entonces escribiremos $r > r'$ y diremos que la regla $r \in \mathcal{R}_i$ tiene más prioridad que la regla $r' \in \mathcal{R}_i$; y (d) el número natural i_{out} representa la etiqueta de la denominada *zona de salida* del sistema en donde se codificarán los posibles *resultados*. Si $i_{out} = 0$ entonces la salida del sistema se codificará en el *entorno* y si $i_{out} \in \{1, \ldots, q\}$ entonces la salida del sistema se codificará en la membrana etiquetada por i_{out}. Para cada membrana $j \in \{2, \ldots, q\}$ se denotará por $p(j)$ a la *membrana padre* de j en el árbol enraizado μ. Si $p(j) = i$ entonces diremos que j es un *hijo* de la membrana i. Por cuestiones técnicas convendremos en que la membrana padre de la piel será el entorno, es decir, $p(1) = 0$. Las *hojas* del árbol enraizado (es decir, las membranas del sistema que carecen de hijos) se denominan *membranas elementales*.

Una *configuración* o *descripción instantánea* $\mathcal{C}_t$ en un instante t de un sistema P básico de transición $\Pi = (\Gamma, \mu, \{\mathcal{M}_i, \mathcal{R}_i, \rho_i \mid 1 \leq i \leq q\}, i_{out})$ es una tupla $(\mu_t, \mathcal{M}_{p_1}, \ldots, \mathcal{M}_{p_t})$ cuyas componentes son la estructura de membranas μ_t del sistema en el instante t (conteniendo p_t membranas) y los multiconjuntos $\mathcal{M}_{p_1}, \ldots, \mathcal{M}_{p_t}$ asociados a cada una de las membranas que conforman μ_t. La *configuración inicial* $\mathcal{C}_0$ del sistema Π es la tupla $(\mu, \mathcal{M}_1, \ldots, \mathcal{M}_q)$.

El concepto de *aplicabilidad de una regla* a una configuración $\mathcal{C}_t$ hace referencia a si una cierta regla puede ser aplicada, o no, a dicha configuración. Una regla $r \in \mathcal{R}_i$ del tipo $u \to (v_1, here)(v_2, out)(v_3, in_j)\delta^*$ es *aplicable* a una configuración $\mathcal{C}_t$ en un instante t si se verifican las condiciones siguientes:

- En la estructura de membranas correspondiente a $\mathcal{C}_t$ existe una membrana etiquetada por i tal que contiene al multiconjunto u.
- En la estructura de membranas correspondiente a $\mathcal{C}_t$ existe una membrana etiquetada por j tal que su padre es la membrana antes citada; es decir, $p(j) = i$.
- Si $\delta^* = \delta$, entonces i no es la membrana piel ni, en su caso, la membrana de salida.
- No existe $r' \in \mathcal{R}_i$ tal que r' sea aplicable a $\mathcal{C}_t$ y, además, $(r', r) \in \rho_i$; es decir, no existe ninguna regla de $\mathcal{R}_i$ que sea aplicable a $\mathcal{C}_t$ y tenga *mayor* prioridad que la regla r (*versión fuerte* de la prioridad).

Si una regla $r \in \mathcal{R}_i$ del tipo $u \to (v_1, here)(v_2, out)(v_3, in_j)\delta^*$ es *aplicable* a una configuración $\mathcal{C}_t$, entonces la aplicación de dicha regla a una membrana de $\mathcal{C}_t$ etiquetada por i, produce el siguiente efecto:

- El multiconjunto u se elimina de dicha membrana i.
- El multiconjunto v_1 se añade al contenido de esa membrana.
- El multiconjunto v_2 se añade al contenido de la membrana padre de i. En el caso en que i sea la membrana piel, entonces el multiconjunto v_2 se envía al entorno del sistema.
- El multiconjunto v_3 se añade al contenido de una membrana etiquetada por j, hija de la membrana i.
- Si $\delta' = \delta$, entonces esa membrana etiquetada por i se *disuelve* (es decir, desaparece de la estructura de membranas de la configuración $\mathcal{C}_{t+1}$) y su contenido se añade al primer antecesor que no se haya disuelto en el paso $t+1$.

Se dirá que una configuración $\mathcal{C}_t$ del sistema P básico de transición Π produce una configuración $\mathcal{C}_{t+1}$ en un *paso de transición*, lo denotaremos por $\mathcal{C}_t \Rightarrow_\Pi \mathcal{C}_{t+1}$ y diremos que $\mathcal{C}_{t+1}$ es *una configuración siguiente* de $\mathcal{C}_t$, si se puede pasar de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ aplicando las reglas de $\mathcal{R}_1 \cup \cdots \cup \mathcal{R}_q$ de forma *no determinista*, paralela y *maximal* a la configuración $\mathcal{C}_t$. Esto es, la configuración $\mathcal{C}_{t+1}$ se obtiene a partir de la configuración $\mathcal{C}_t$ aplicando un cierto multiconjunto de reglas que verifica las siguientes condiciones: (a) todas las reglas deben ser *aplicables* a $\mathcal{C}_t$; y (b) dicho multiconjunto debe ser *maximal*, en el sentido de que tras la aplicación de todas las reglas del multiconjunto (con las multiplicidades respectivas) a la configuración $\mathcal{C}_t$, no puede quedar en $\mathcal{C}_t$ ningún objeto que pudiera evolucionar por alguna regla del sistema. Teniendo presente que pueden existir distintos multiconjuntos maximales de reglas que son aplicables a la configuración $\mathcal{C}_t$, a la hora de implementar un paso de transición de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ se seleccionará de manera no determinista uno de esos multiconjuntos maximales de reglas.

Una *computación* de un sistema P básico de transición Π es una sucesión (finita o infinita) de configuraciones tal que: (a) el primer término de la sucesión es la configuración inicial del sistema; (b) a partir del segundo término de la sucesión, cada configuración es obtenida de la configuración previa mediante un paso de transición; y (c) si la sucesión es finita (*computación de parada*) entonces el último término de la sucesión es una *configuración de parada*; es decir una configuración tal que ninguna regla del sistema le es aplicable. Todas las computaciones comienzan desde una configuración inicial y evolucionan

mediante pasos de transición. Sólo las computaciones que son de parada proporcionan un *resultado*, el cuál estará codificado por los objetos presentes en la zona de salida i_{out} de la correspondiente configuración de parada.

4.2.2. Sistemas P con membranas activas

En un sistema P básico de transición la estructura de membranas nunca puede incrementar su tamaño; más aún, la aplicación de reglas de disolución propicia que dicho tamaño disminuya. Por otra parte, las reglas de un tal sistema son ejecutadas en paralelo y, en consecuencia, puede proporcionar una cantidad exponencial de objetos en un número lineal de pasos de computación. En efecto, sea un sistema P básico de transición cuya configuración inicial contiene un objeto a_1 en una cierta membrana y tal que $a_i \rightarrow (a_{i+1}^2, here)$, para $1 \leq i \leq n$, son reglas asociadas a dicha membrana. Entonces, al cabo de n pasos de transición se obtendrían 2^n objetos en esa membrana. Pues bien, es importante destacar el hecho siguiente: la posibilidad de construir una cantidad de espacio exponencial (expresado a través del número de objetos) en un número lineal de pasos de transición, no es condición suficiente para poder garantizar que en esos sistemas se puedan resolver problemas **NP**-completos en tiempo polinomial (a menos que $\mathbf{P} = \mathbf{NP}$). Para más detalles, véase la referencia [17].

En [56] se introdujeron unos sistemas de membranas que trabajan a modo de células (los *sistemas P con membranas activas*) que tenían la capacidad de incrementar el tamaño de la estructura inicial de membranas. En dichos sistemas se permite el uso de un nuevo tipo de reglas, denominadas de *división de membranas*, que están inspiradas en la mitosis celular y permiten construir en tiempo polinomial un espacio de trabajo de tamaño exponencial, expresado en términos del número de membranas y el número de objetos. De esta forma, los nuevos sistemas de membranas tendrán la capacidad de proporcionan soluciones polinomiales de problemas computacionalmente duros.

Definición 4.2. *Un* sistema P con membranas activas *de grado* $q \geq 1$ *es una tupla* $\Pi = (\Gamma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$*, en donde:*

- Γ *es un alfabeto finito.*
- μ *es un árbol enraizado cuyos nodos están biyectivamente etiquetados por elementos del conjunto* $H = \{1, \ldots, q\}$ *(la raíz del árbol está etiquetada por 1).*

- $\mathcal{M}_i, 1 \leq i \leq q$, *son multiconjuntos sobre* Γ.
- $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ *de la forma:*
  - *(a)* $[\, a \rightarrow u \,]_h^\alpha$, *para* $h \in H, \alpha \in \{+, -, 0\}, a \in \Gamma$, $u \in \Gamma^\star$ (*reglas de evolución de objetos*).
  - *(b)* $a[\ \ ]_h^{\alpha_1} \rightarrow [\, b \,]_h^{\alpha_2}$, *para* $h \in H \setminus \{1\}, \alpha_1, \alpha_2 \in \{+, -, 0\}$, $a, b \in \Gamma$ (*reglas de comunicación hacia dentro*).
  - *(c)* $[\, a \,]_h^{\alpha_1} \rightarrow b[\ \ ]_h^{\alpha_2}$, *para* $h \in H, \alpha_1, \alpha_2 \in \{+, -, 0\}, a, b \in \Gamma$ (*reglas de comunicación hacia fuera*).
  - *(d)* $[\, a \,]_h^\alpha \rightarrow b$, *para* $h \in H \setminus \{1, i_{out}\}, \alpha \in \{+, -, 0\}, a, b \in \Gamma$ (*reglas de disolución*).
  - *(e)* $[\, a \,]_h^\alpha \rightarrow [\, b \,]_h^{\alpha_1} [\, c \,]_h^{\alpha_2}$, *para* $h \in H \setminus \{1, i_{out}\}$, *h es la etiqueta de una hoja de* μ, $\alpha_1, \alpha_2, \alpha_3 \in \{+, -, 0\}$, $a, b, c \in \Gamma$ (*reglas de división para membranas elementales*).
  - *(f)* $[\,[\,]_{h_1}^{\alpha_1} \ldots [\,]_{h_k}^{\alpha_1} [\,]_{h_{k+1}}^{\alpha_2} \ldots [\,]_{h_n}^{\alpha_2}]_h^\alpha \rightarrow [\,[\,]_{h_1}^{\alpha_3} \ldots [\,]_{h_k}^{\alpha_3}]_h^\beta [\,[\,]_{h_{k+1}}^{\alpha_4} \ldots [\,]_{h_n}^{\alpha_4}]_h^\gamma$, *para* $n > k \geq 1, h_1, \ldots, h_n \in H, h \in H \setminus \{1, i_{out}\}, \alpha, \beta, \gamma, \alpha_1, \alpha_2, \alpha_3, \alpha_4 \in \{+, -, 0\}, \{\alpha_1, \alpha_2\} = \{+, -\}$ (*reglas de división para membranas no elementales*).
- $i_{out} \in \{0, 1, \ldots, q\}$.

Un sistema P con membranas activas Π de grado $q \geq 1$ puede ser considerado como un conjunto de q membranas situado en un determinado *entorno* externo. Las membranas están organizadas a través de una estructura jerárquica de membranas que delimitan regiones o compartimentos, materializada por un árbol enraizado μ cuya raíz es la *membrana* que se denomina *piel*, etiquetada por 1, y de tal manera que: (a) cada nodo del árbol (membrana) está etiquetado por un elemento de $H = \{1, \ldots, q\}$ y posee una carga eléctrica que puede ser positiva (+), negativa (-), o neutra (0); (b) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ representan los multiconjuntos de objetos (símbolos de Γ) inicialmente situados en las q regiones determinadas por las membranas del sistema; (c) $\mathcal{R}$ es un conjunto finito de reglas asociado al sistema; y (d) el número natural i_{out} representa la etiqueta de la denominada *zona de salida* del sistema en donde se codificarán los posibles *resultados*. Si $i_{out} = 0$ entonces la salida del sistema se codificará en el *entorno* y si $i_{out} \in \{1, \ldots, q\}$ entonces la salida del sistema se codificará en la membrana etiquetada por i_{out}. En adelante, cuando hablemos de la etiqueta de una membrana de un tal sistema, nos referiremos al elemento de H que la identifica. Así pues, a diferencia de lo que sucedía en los sistemas P

básicos de transición, las reglas de estos sistemas están asociadas a etiquetas, no a membranas, y las reglas de división permiten que, a lo largo del tiempo, existan distintas membranas que posean la misma etiqueta.

En lo que respecta a la semántica de los sistemas P con membranas activas, los conceptos de configuración (teniendo presente que en el nuevo marco pueden existir distintas membranas con la misma etiqueta), y computación se definen de manera similar a como se hizo con los sistemas P básicos de transición. El punto clave radica en definir cómo se implementa un *paso de transición* de una configuración del sistema a una configuración siguiente. Para ello, hay que definir previamente el concepto de aplicabilidad de las reglas del sistema a una configuración, así como el efecto que produce la aplicación de dichas reglas. Más aún, a diferencia de lo que sucedía en los sistemas P básicos de transición, el concepto de maximalidad en la ejecución de las reglas del sistema, ha de ser matizado.

Diremos que una *regla de evolución de objetos* $[\, a \rightarrow u \,]_h^{\alpha}$ será aplicable a una configuración $\mathcal{C}_t$ del sistema en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por h, su carga eléctrica es α y, además, contiene al objeto a. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ produce la sustitución en dicha membrana de un objeto a por un multiconjunto u de objetos. Así pues, la aplicación de dicha regla no altera ni la etiqueta ni la carga de la membrana en la que se aplica.

Diremos que una regla *de comunicación hacia dentro* $a\,[\quad]_h^{\alpha_1} \rightarrow [\, b \,]_h^{\alpha_2}$ es aplicable a una configuración $\mathcal{C}_t$ del sistema en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por h, distinta de la membrana piel, su carga eléctrica es α y, además, el objeto a aparece en la membrana padre de h. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ produce la eliminación de un objeto a de la membrana padre de h y la aparición de un objeto b en la membrana h a la que se aplica la regla. Además, tras la aplicación de dicha regla, la carga eléctrica de esa membrana pasa a ser β, mientras que su etiqueta no queda alterada.

Diremos que una *regla de comunicación hacia fuera* $[\, a \,]_h^{\alpha_1} \rightarrow b\,[\quad]_h^{\alpha_2}$ es aplicable a una configuración $\mathcal{C}_t$ del sistema en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por h, su carga eléctrica es α y, además, contiene al objeto a. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ produce la eliminación de un objeto a de dicha membrana y

la aparición de un objeto b en su membrana padre (si h es la membrana piel, el objeto b pasaría al entorno del sistema). Además, tras la aplicación de dicha regla la carga eléctrica de esa membrana pasa a ser β, mientras que su etiqueta no queda alterada.

Diremos que una *regla de disolución* $[\,a\,]_h^\alpha \to b$ es aplicable a una configuración $\mathcal{C}_t$ del sistema en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por h, distinta de la piel y de la membrana de salida, si la hubiera, su carga eléctrica es α y, además, contiene al objeto a. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ produce la eliminación de dicha membrana del sistema y su contenido (en donde el objeto a es sustituido por el objeto b) pasará a la membrana que sea el primer antecesor de esa membrana que no haya sido disuelto en el paso $t+1$.

Diremos que una *regla de división de membranas elementales* $[\,a\,]_h^\alpha \to [\,b\,]_h^{\alpha_1}\,[\,c\,]_h^{\alpha_2}$ es aplicable a una configuración $\mathcal{C}_t$ del sistema en un instante t si en dicha configuración existe, al menos, una membrana elemental que está etiquetada por h, distinta de la piel y de la membrana de salida si la hubiera, su carga eléctrica es α y, además, contiene al objeto a. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ produce la creación de dos nuevas membranas con la misma etiqueta h, la primera de ellas con carga β y la segunda con carga γ. El objeto a que dispara la regla desaparece y es sustituido por el objeto b, en la primera de las membranas creadas, y por el objeto c en la segunda de esas membranas. Los restantes objetos que aparecían en la membrana donde se ha aplicado la regla, son replicados («copiados») en las dos membranas creadas.

Diremos que una *regla de división de membranas no elementales*

$$[\,[\,]_{h_1}^{\alpha_1}\dots[\,]_{h_k}^{\alpha_1}\,[\,]_{h_{k+1}}^{\alpha_2}\dots[\,]_{h_n}^{\alpha_2}\,]_h^\alpha \to [\,[\,]_{h_1}^{\alpha_3}\dots[\,]_{h_k}^{\alpha_3}\,]_h^\beta\,[\,[\,]_{h_{k+1}}^{\alpha_4}\dots[\,]_{h_n}^{\alpha_4}\,]_h^\gamma$$

es aplicable a una configuración $\mathcal{C}_t$ del sistema en un instante t si en dicha configuración existe, al menos, una membrana no elemental etiquetada por h, distinta de la piel y de la membrana de salida si la hubiera, tal que su carga eléctrica es α y contiene membranas etiquetadas por $h_1,\dots h_k, h_{k+1},\dots h_n$ de manera que las k primeras tienen carga eléctrica α_1 y las restantes tienen carga eléctrica α_2, siendo $\{\alpha_1,\alpha_2\}=\{+,-\}$. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ etiquetada por h, produce la creación de dos nuevas membranas con la misma etiqueta h de tal manera que: (a) la primera de ellas posee carga eléctrica β y contiene a las membranas etiquetadas $h_1,\dots h_k$ cuyas cargas eléctricas pasan a ser α_3; y (b) la segunda posee carga eléctrica γ y contiene

a las membranas etiquetadas $h_{k+1}, \ldots h_n$ cuyas cargas eléctricas pasan a ser α_4.

Se dirá que una configuración $\mathcal{C}_t$ del sistema P con membranas activas Π produce una configuración $\mathcal{C}_{t+1}$ en un *paso de transición*, lo denotaremos por $\mathcal{C}_t \Rightarrow_\Pi \mathcal{C}_{t+1}$ y diremos que $\mathcal{C}_{t+1}$ es *una configuración siguiente* de $\mathcal{C}_t$, si se puede pasar de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ aplicando las reglas de $\mathcal{R}$ de acuerdo con los siguientes principios:

1. A un objeto arbitrario de una membrana cualquiera sólo se le puede aplicar, a lo sumo, una regla (escogida, en su caso, de forma no determinista).

2. A cada membrana sólo se le puede aplicar, a lo sumo, una regla del tipo (b), (c), (d), (e), o (f) (escogida, en su caso, de forma no determinista) y, en tal situación, se aplicará una única vez; es decir, este tipo de reglas se aplican a cada membrana de manera secuencial.

3. Una regla del tipo (a) se puede aplicar simultáneamente con una regla de los tipos antes citado. Más aún, en todo caso, las reglas del tipo (a) deberán aplicarse siguiendo el criterio de maximalidad (no pueden quedar objetos sin evolucionar a los que se les pudieran aplicar reglas de ese tipo).

Conviene aclarar que si en un paso de transición se le puede aplicar a una membrana, simultáneamente, una regla del tipo (b), (c), (d), (e) o (f) junto con unas reglas de evolución de objetos, entonces podemos imaginar que ese paso de transición consta de dos *micropasos* de tal manera que en el primero de ellos, se aplican las reglas de evolución de objetos y, en el segundo, se aplica, propiamente, esa otra regla.

Algunas características importantes a resaltar de los sistemas P con membranas activas son las siguientes: (a) usan tres tipos de cargas eléctricas; (b) las reglas están asociadas a etiquetas no a membranas; (c) la polarización de una membrana puede ser modificada por la aplicación de una regla pero, en cambio, su etiqueta no puede ser alterada; y (d) las reglas no son cooperativas; es decir, la parte izquierda de las mismas constan de un solo símbolo; y (e) las reglas de dichos sistemas están asociados a etiquetas, no a membranas.

4.2.3. Sistemas P con membranas activas sin polarización

Definición 4.3. *Un sistema P con membranas activas y sin polarización de grado* $q \geq 1$ *es una tupla* $\Pi = (\Gamma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$*, en donde:*

1. Γ *es un alfabeto finito.*

2. μ *es un árbol enraizado cuyos nodos están biyectivamente etiquetados por elementos del conjunto* $H = \{1, \ldots, q\}$ *(la raíz del árbol está etiquetada por* 1*).*

3. $\mathcal{M}_i, 1 \leq i \leq q$*, son multiconjuntos sobre* Γ*.*

4. $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ *de la forma:*

   (a) $[\, a \rightarrow u\,]_h$*, para* $h \in H, a \in \Gamma, u \in \Gamma^*$ *(reglas de evolución de objetos).*

   (b) $a[\ \ ]_h \rightarrow [\, b\,]_h$*, para* $h \in H \setminus \{1\}, a, b \in \Gamma$ *(reglas de comunicación hacia dentro).*

   (c) $[\, a\,]_h \rightarrow b[\ \ ]_h$*, para* $h \in H, a, b \in \Gamma$ *(reglas de comunicación hacia fuera).*

   (d) $[\, a\,]_h \rightarrow b$*, para* $h \in H \setminus \{1, i_{out}\}, a, b \in \Gamma$ *(reglas de disolución).*

   (e) $[\, a\,]_h \rightarrow [\, b\,]_h [\, c\,]_h$*, para* $h \in H \setminus \{1, i_{out}\}$*,* h *es la etiqueta de una hoja de* μ*,* $a, b, c \in \Gamma$ *(reglas de división para membranas elementales).*

   (f) $[\,[\,]_{h_1} \ldots [\,]_{h_k} [\,]_{h_{k+1}} \ldots [\,]_{h_n}\,]_h \rightarrow [\,[\,]_{h_1} \ldots [\,]_{h_k}\,]_h [\,[\,]_{h_{k+1}} \ldots [\,]_{h_n}\,]_h$*, para* $k \geq 1, n > k, h_1, \ldots h_n \in H, h \in H \setminus \{1, i_{out}\}$ *(reglas de división para membranas no elementales).*

5. $i_{out} \in \{0, 1, \ldots, q\}$*.*

Este tipo de sistemas de membranas activas son similares a los anteriores, con la única novedad de que no existen cargas eléctricas asociadas a las membranas (lo cual equivale a considerar la existencia de una única carga eléctrica asociada a cada membrana). La semántica de los sistemas P con membranas activas y sin polarización se define de manera similar a la realizada con los sistemas P con membranas activas y cargas eléctricas.

4.2.4. Sistemas P con reglas symport/antiport y división

Los sistemas P que trabajan a modo de células y usan reglas de comunicación symport/antiport, fueron introducidos en [41] y están inspirados en el fenómeno biológico del transporte de pares de sustancias químicas a traves de las membranas. Además, las reglas de división fueron incorporadas a estos sistemas en [68].

Definición 4.4. *Un sistema P con reglas symport/antiport y división de membranas de grado $q \geq 1$, es una tupla $\Pi = (\Gamma, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{out})$, en donde:*

1. *Γ es un alfabeto finito;*
2. *$\mathcal{E} \subseteq \Gamma$;*
3. *μ es un árbol enraizado cuyos nodos están biyectivamente etiquetados por $1, \ldots, q$ (la raíz del árbol está etiquetada por 1);*
4. *$\mathcal{M}_1, \ldots, \mathcal{M}_q$ son multiconjuntos sobre Γ;*
5. *$\mathcal{R}_i$, $1 \leq i \leq q$, son conjuntos finitos de reglas de comunicación sobre Γ de la forma:*
   (a) *$(u, out) \in \mathcal{R}_i$ o $(u, in) \in \mathcal{R}_i$, siendo u un multiconjunto sobre Γ tal que $|u| > 0$* (Reglas symport)*;*
   (b) *$(u, out; v, in) \in \mathcal{R}_i$, siendo u, v multiconjuntos sobre Γ tales que $|u| > 0$ y $|v| > 0$* (Reglas antiport)*;*
   (c) *$[\, a \,]_i \to [\, b \,]_i [\, c \,]_i \in \mathcal{R}_i$, siendo $a, b, c \in \Gamma$, $i \in \{2, \ldots, q\}$, $i \neq i_{out}$, e i es la etiqueta de una hoja del árbol μ* (Reglas de división)*;*
6. *$i_{out} \in \{0, 1, \ldots, q\}$.*

Un sistema P con reglas con reglas symport/antiport y división celular, de grado $q \geq 1$, $\Pi = (\Gamma, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{out})$, puede ser considerado como un conjunto de q membranas etiquetadas por $1, \ldots, q$ tales que: (a) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ representan los multiconjuntos de objetos colocados inicialmente en las q membranas del sistema; (b) $\mathcal{E}$ es el conjunto de objetos colocados inicialmente en el entorno, cada uno de ellos con un número infinito de copias; (c) $\mathcal{R}_1, \ldots, \mathcal{R}_q$ son conjuntos finitos de reglas sobre Γ tal que el conjunto $\mathcal{R}_i$ está asociada a la membrana i de μ; y (d) i_{out} representa una *zona* distinguida

que codificará la salida del sistema. Usaremos el término *zona* $i (0 \leq i \leq q)$ para referirnos a la membrana i, en el caso $1 \leq i \leq q$, o bien al entorno, en el caso $i = 0$. La longitud de una regla $(u, out) \in \mathcal{R}_i$ o $(u, in) \in \mathcal{R}_i$ (resp., $(u, out; v, in) \in \mathcal{R}_i$) está definida por $|u|$ (resp. $|u| + |v|$).

En la figura 4.2 se puede ver una representación gráfica de las reglas symport y antiport.

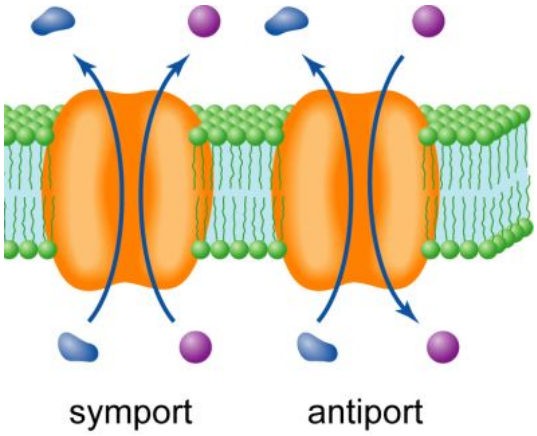


Figura 4.2: Reglas symport y antiport.

Una *configuraciónn* en un instante t de un tal sistema está descrita por la estructura de membranas en dicho instante, los multiconjuntos de objetos de Γ colocados en cada membrana de esa estructura, así como el multiconjunto de objetos de $\Gamma \setminus \mathcal{E}$ colocados en el entorno, en ese instante. La configuración inicial de $\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$ es $(\mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \emptyset)$.

Una regla symport $(u, out) \in \mathcal{R}_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por i y contiene el multiconjunto u. La ejecución de una tal regla $(u, out) \in \mathcal{R}_i$ a esa membrana i de $\mathcal{C}_t$ produce los siguientes efectos: los objetos del multiconjunto representado por u son consumidos de esa membrana y, a su vez, son colocados en la membrana padre de i (que será el entorno, en el caso en que i sea la raíz del árbol μ). Una regla symport $(u, in) \in \mathcal{R}_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por i tal que su membrana padre contiene el multiconjunto u. La ejecución de una tal regla $(u, in) \in \mathcal{R}_i$ a esa membrana i de $\mathcal{C}_t$ produce los siguientes efectos: los objetos del multiconjunto representado por u son consumidos de la membrana padre de i (que será el entorno, en el caso en que i sea la raíz del árbol de μ) y, a su vez, son colocados en dicha

membrana etiquetada por i. Una regla antiport $(u, out; v, in) \in \mathcal{R}_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por i, contiene al multiconjunto u y, además, la zona padre de i contiene al multiconjunto v. La ejecución de una tal regla a esa membrana i de $\mathcal{C}_t$ produce los siguientes efectos: los objetos del multiconjunto representados por u son eliminados de esa membrana y, a su vez, son enviados a la correspondiente zona padre; por su parte, los objetos del multiconjunto representados por v son eliminados de la zona padre de i y, a su vez, son enviados a esa membrana i.

Una regla de división $[\, a \,]_i \rightarrow [\, b \,]_i \, [\, c \,]_i \in \mathcal{R}_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por i, distinta de la raíz de μ y de la membrana de salida si la hubiera, de tal manera que contiene al objeto a. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ produce la creación de dos nuevas membranas con la misma etiqueta i de tal manera que el objeto a que dispara la regla desaparece y es sustituido por el objeto b, en la primera de las membranas creadas, y por el objeto c en la segunda de esas membranas. Los restantes objetos que aparecían en la membrana donde se ha aplicado la regla, son replicados («copiados») en las dos membranas creadas.

Se dirá que una configuración $\mathcal{C}_t$ del sistema P con reglas symport/antiport y división celular Π produce una configuración $\mathcal{C}_{t+1}$ en un *paso de transición*, lo denotaremos por $\mathcal{C}_t \Rightarrow_{\Pi} \mathcal{C}_{t+1}$ y diremos que $\mathcal{C}_{t+1}$ es *una configuración siguiente* de $\mathcal{C}_t$, si se puede pasar de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ aplicando las reglas de $\mathcal{R}$ de acuerdo con los siguientes principios:

- A un objeto arbitrario de una membrana cualquiera sólo se le puede aplicar, a lo sumo, una regla (escogida, en su caso, de forma no determinista).
- A cada membrana sólo se le puede aplicar o bien reglas de comunicación o bien una regla de división. En el caso de aplicarse reglas de comunicación a la configuración $\mathcal{C}_t$, éstas se aplicarán, en su caso, de forma *no determinista*, paralela y *maximal*. En el caso de aplicarse una regla de división a la configuración $\mathcal{C}_t$, ésta será seleccionada de forma no determinista. Así pues, podemos imaginar que cuando se aplica una regla de división a una membrana de una configuración $\mathcal{C}_t$, la propia membrana queda bloqueada a efectos de poder comunicarse con otras membranas vecinas o con el entorno.

Las nuevas membranas resultantes de la división interactuarán con las otras membranas o con el entorno sólo en el siguiente paso de transición, siempre que

no se vuelvan a dividir. Además, esas membranas tendrán las mismas etiquetas que la membrana que se divide y, por tanto, proporcionarán nuevas aristas en el árbol enraizado.

A partir del concepto de paso de transición se define, de manera natural, el concepto de computación del sistema.

4.2.5. Sistemas P con reglas symport/antiport y separación

Seguidamente, se introduce un nuevo tipo de sistemas P con reglas symport/antiport, en donde las reglas de división de membranas son reemplazadas por reglas de separación. Estas reglas están inspiradas en el fenómeno biológico de la fisión celular y fueron incorporadas a estos sistemas en [67].

Definición 4.5. *Un sistema P con reglas symport/antiport y separación de membranas, de grado* $q \geq 1$ *es una tupla* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{out})$*, en donde:*

(a) Γ *es un alfabeto finito;*

(b) $\mathcal{E} \subsetneq \Gamma$*;*

(c) $\{\Gamma_0, \Gamma_1\}$ *es una partición de* Γ*; es decir,* $\Gamma = \Gamma_0 \cup \Gamma_1, \Gamma_0, \Gamma_1 \neq \emptyset$*,* $\Gamma_0 \cap \Gamma_1 = \emptyset$*;*

(d) μ *es un árbol enraizado cuyos nodos están biyectivamente etiquetados por* $1, \ldots, q$ *(la raíz del árbol está etiquetada por* 1*);*

(e) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ *son multiconjuntos sobre* Γ*;*

(f) $\mathcal{R}_i, 1 \leq i \leq q$*, son conjuntos finitos de reglas sobre* Γ *de la forma:*

  (a) $(u, out) \in \mathcal{R}_i$ *o* $(u, in) \in \mathcal{R}_i$*, siendo* u *un multiconjunto sobre* Γ *tal que* $|u| > 0$ (Reglas symport)*;*

  (b) $(u, out; v, in) \in \mathcal{R}_i$*, siendo* u, v *multiconjuntos sobre* Γ *tales que* $|u| > 0$ *y* $|v| > 0$ (Reglas antiport)*;*

  (c) $[\,a\,]_i \rightarrow [\,\Gamma_0\,]_i\,[\,\Gamma_1\,]_i \in \mathcal{R}_i$*, siendo* $a \in \Gamma$*,* $i \in \{2, \ldots, q\}$*,* $i \neq i_{out}$*, e* i *es la etiqueta de una hoja del árbol* μ (Reglas de separación)*;*

(g) $i_{out} \in \{0, 1, \ldots, q\}$.

Una regla de separación $[\,a\,]_i \rightarrow [\,\Gamma_0\,]_i\,[\,\Gamma_1\,]_i \in \mathcal{R}_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una membrana que está etiquetada por i, distinta de la raíz de μ y de la membrana de salida si la hubiera y, además, contiene al objeto a. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ produce la creación de dos nuevas membranas con la misma etiqueta i de tal manera que el objeto a que dispara la regla desaparece y, además, los objetos de la membrana a la que se ha aplicado la regla y pertenecen a Γ_0 (respectivamente, Γ_1) irán a para a la primera (respectivamente, segunda) de las membranas creadas.

Con respecto a la semántica de esta variante, las reglas serán aplicadas de forma no determinista y maximal con la siguiente restricción: cuando una membrana etiquetada por i es separada, entonces la regla de separación será la única regla de $\mathcal{R}_i$ que se aplicará a esa membrana en ese paso de computación. Las nuevas membranas resultantes de la separación interactuarán con las otras membranas o con el entorno sólo en el siguiente paso de transición, siempre que no se vuelvan a separar. Además, esas membranas tendrán las mismas etiquetas que la membrana que se separa y, por tanto, proporcionarán nuevas aristas al árbol enraizado.

4.3. Sistemas de membranas que trabajan a modo de tejidos

Los sistemas de membranas que trabajan a modo de tejidos son dispositivos computacionales de *Membrane Computing*, inspirados en la intercomunicación celular en tejidos y la cooperación entre neuronas [30]. En estos sistemas, la comunicación entre las células está basada en reglas del tipo symport/antiport [42]. Estas reglas de comunicación proporcionan, implícitamente, un grafo dirigido que está asociado al sistema mediante las etiquetas de las células.

Esta sección está dedicada a introducir sistemas de membranas que trabajan a modo de tejidos: (a) los sistemas P básicos de tejidos; (b) los sistemas P de tejidos con división celular; (c) los sistemas P de tejidos con separación celular; y (d) los sistemas P de tejidos con reglas de comunicación evolutiva.

4.3.1. Sistemas P básicos de tejidos

Los sistemas P de tejidos son dispositivos computacionales de *Membrane Computing* inspirados en la intercomunicación celular en tejidos y la cooperación entre neuronas [30]. En estos sistemas, la comunicación entre las células

está basada en reglas del tipo symport/antiport, que fueron incorporadas a los sistemas P en [42]. Estas reglas de comunicación proporcionan, implícitamente, un grafo dirigido que está asociado al sistema mediante las etiquetas de las células.

Definición 4.6. *Un sistema P básico de tejidos de grado* $q \geq 1$ *es una tupla*

$$\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$$

en donde:

- Γ *es un alfabeto finito,* $\mathcal{E} \subseteq \Gamma$ *y* $\mathcal{M}_j, 1 \leq j \leq q$*, son multiconjuntos sobre* Γ*.*
- $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ*, de comunicación symport/antiport, de la forma* $(i, u/v, j)$*, siendo* $i, j \in \{0, 1, \ldots, q\}$*,* $i \neq j, u, v \in M(\Gamma), |u| + |v| > 0$*.*
- $i_{out} \in \{0, 1, \ldots, q\}$*.*

Un sistema P básico de tejidos Π de grado $q \geq 1$ puede considerarse como un conjunto de q células, situadas en un determinado *entorno* externo. Las células están etiquetadas de manera biyectiva mediante elementos del conjunto $\{1, \ldots, q\}$ y, a su vez, están organizadas a través de una estructura jerárquica de grafo dirigido que está definido, implícitamente, a través de las reglas del sistema, de tal manera que: (a) $\mathcal{E}$ representa el conjunto de objetos situados inicialmente en el entorno del sistema, todos ellos apareciendo en un número arbitrario (teóricamente infinito) de copias; (b) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ son multiconjuntos de objetos (elementos en Γ) inicialmente situados en las q células del sistema; (c) $\mathcal{R}$ es un conjunto finito de reglas de comunicación sobre Γ, del tipo symport/antiport, asociadas al sistema; (d) el número natural $i_{out} \in \{0, 1, 2, \ldots, q\}$ representa la etiqueta de la denominada *zona de salida* del sistema en donde se codificarán los posibles *resultados*. Si $i_{out} = 0$ entonces la salida del sistema se codificará en el *entorno* y si $i_{out} \in \{1, \ldots, q\}$ entonces la salida del sistema se codificará en la célula etiquetada por i_{out}.

Una regla de comunicación de la forma $(i, u/v, j)$, con $i, j \in \{0, 1, \ldots, q\}, i \neq j, u, v \in M(\Gamma)$, $|u| + |v| > 0$, se dice que es de *tipo symport* si $u = \lambda$ o $v = \lambda$. Si, en cambio, $u \neq \lambda$ y $v \neq \lambda$, entonces dicha regla de comunicación se dice que es de *tipo antiport*. Cada regla de tipo symport $(i, u/\lambda, j)$, con $i \neq 0, j \neq 0$,

proporciona un arco virtual desde la célula i hasta la célula j y cada regla de tipo *antiport* $(i, u/v, j)$, con $i \neq 0, j \neq 0$, proporciona dos arcos: uno desde la célula i hasta la j y otro desde la célula j hasta la i. Así pues, cada sistema P de tejidos tiene un grafo dirigido subyacente cuyos nodos son las células del sistema y los arcos son obtenidos mediante reglas de comunicación. En este contexto, el entorno puede ser considerado como un nodo virtual de dicho grafo tal que sus conexiones están definidas por las reglas de comunicación de la forma $(i, u/v, j)$, con $i = 0$ ó $j = 0$. La *longitud* de una regla de comunicación $(i, u/v, j)$ se define como el número natural $|u| + |v|$.

Una *configuración* o *descripción instantánea* $\mathcal{C}_t$ en un instante t de un sistema P básico de tejidos $\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$ es una tupla que consta de los multiconjuntos de objetos sobre Γ asociados a cada una de las células del sistema en el instante t, junto con el multiconjuntos de objetos sobre $\Gamma \setminus \mathcal{E}$ asociado al sistema en el instante t. Téngase presente que los objetos del alfabeto $\mathcal{E}$ aparecen inicialmente en una cantidad infinita de copias y, en consecuencia, esa cantidad no queda alterada por la aplicación de reglas de comunicación. La *configuración inicial* $\mathcal{C}_0$ del sistema Π es la tupla $(\mathcal{M}_1, \ldots, \mathcal{M}_q; \emptyset)$.

Una regla de comunicación del tipo symport/antiport es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en $\mathcal{C}_t$ el multiconjunto u está contenido en la zona i y el multiconjunto v está contenido en la zona j. La ejecución de una regla de comunicación $(i, u/v, j)$ a una configuración $\mathcal{C}_t$ produce los siguientes efectos: los objetos del multiconjunto representado por u son enviados desde la zona i hasta la zona j en $\mathcal{C}_t$ y, simultáneamente, los objetos del multiconjunto v son enviados desde la zona j hasta la zona i en $\mathcal{C}_t$.

Se dirá que una configuración $\mathcal{C}_t$ de un sistema P básico de tejidos Π produce una configuración $\mathcal{C}_{t+1}$ en un *paso de transición*, lo denotaremos por $\mathcal{C}_t \Rightarrow_\Pi \mathcal{C}_{t+1}$ y diremos que $\mathcal{C}_{t+1}$ es *una configuración siguiente* de $\mathcal{C}_t$, si se puede pasar de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ aplicando las reglas de $\mathcal{R}$ de forma *no determinista*, paralela y *maximal* a la configuración $\mathcal{C}_t$, de acuerdo con lo indicado previamente en los sistemas P básicos de transición.

El concepto de *computación* de un sistema P básico de tejidos se define de manera análoga a como se hizo en los sistemas P básicos de transición. Todas las computaciones comienzan desde una configuración inicial y evolucionan mediante pasos de transición. Sólo las computaciones que son de parada proporcionan un *resultado*, el cuál estará codificado por los objetos presentes

en la zona de salida i_{out} de la correspondiente configuración de parada. Si $\mathcal{C} = \{\mathcal{C}_i\}_{i<r+1}\,(r \in \mathbb{N})$ es una configuración de parada de Π, entonces la *longitud* de $\mathcal{C}$ es r, es decir, r es el número de configuraciones no iniciales que aparecen en la sucesión finita $\mathcal{C}$, y se denotará por $|\mathcal{C}|$. También se denotará por $\mathcal{C}_i(j)$ el contenido de la célula j en la configuración $\mathcal{C}_i$. Es interesante tener presente que, a lo largo de cada computación, todos los objetos de $\Gamma \setminus \mathcal{E}$ que aparezcan en el entorno tendrán una multiplicidad finita.

4.3.2. Sistemas P de tejidos con división celular

Los sistemas P de tejidos con división celular se definen a partir de los sistemas P básicos de tejidos permitiendo el uso de reglas de división celular, que están inspiradas en el fenómeno biológico de la mitosis celular. Las reglas de división celular fueron incorporadas a estos sistemas en [45].

Definición 4.7. *Un sistema P de tejidos con división celular de grado $q \geq 1$ es una tupla*

$$\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$$

en donde:

- Γ *es un alfabeto finito,* $\mathcal{E} \subseteq \Gamma$ *y* $\mathcal{M}_j$, $1 \leq j \leq q$, *son multiconjuntos sobre* Γ.
- $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ *de la forma:*
  - *(a) Reglas de comunicación symport/antiport:* $(i, u/v, j)$, *siendo* $i, j \in \{0, 1, \ldots, q\}, i \neq j, u, v \in M(\Gamma), |u| + |v| > 0$.
  - *(b) Reglas de división celular:* $[\,a\,]_i \rightarrow [\,b\,]_i\,[\,c\,]_i$, *siendo* $i \in \{1, 2, \ldots, q\}$, $i \neq i_{out}$ *y* $a, b, c \in \Gamma$.
- $i_{out} \in \{0, 1, \ldots, q\}$.

En lo que respecta a la semántica de los sistema P de tejidos con división celular, los conceptos de configuración (teniendo presente que en el nuevo marco pueden existir distintas células con la misma etiqueta) y computación se definen de manera similar a como se hizo con los sistemas P básicos de tejidos. El punto clave radica en definir cómo se implementa un *paso de transición* de una configuración del sistema a una configuración siguiente. Para ello, hay que definir previamente el concepto de aplicabilidad de las reglas del sistema

a una configuración, así como el efecto que produce la aplicación de dichas reglas. Puesto que este concepto ha sido definido previamente para reglas de comunicación, nos centraremos en las reglas de división.

Una regla de división $[\,a\,]_i \rightarrow [\,b\,]_i\,[\,c\,]_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una célula que está etiquetada por i, distinta de la célula de salida si la hubiera y, además, contiene al objeto a. La ejecución de dicha regla a esa célula de $\mathcal{C}_t$ produce la creación de dos nuevas células con la misma etiqueta i, de tal manera que el objeto a que dispara la regla desaparece y es sustituido por el objeto b, en la primera de las células creadas, y por el objeto c en la segunda de esas células. Los restantes objetos que aparecían en la célula donde se ha aplicado la regla, son replicados («copiados») en las dos células creadas. En la figura 4.3 se puede apreciar una representación de la regla de división celular.

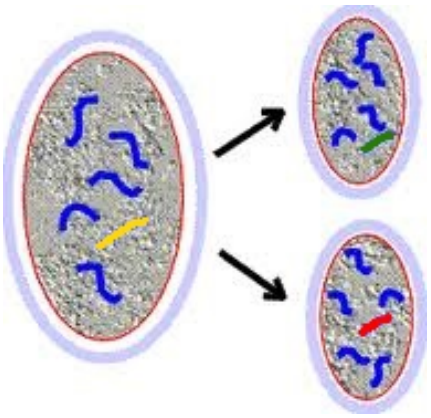


Figura 4.3: Regla de división celular.

Se dirá que una configuración $\mathcal{C}_t$ del sistema P de tejidos con división celular Π produce una configuración $\mathcal{C}_{t+1}$ en un *paso de transición*, lo denotaremos por $\mathcal{C}_t \Rightarrow_\Pi \mathcal{C}_{t+1}$ y diremos que $\mathcal{C}_{t+1}$ es *una configuración siguiente* de $\mathcal{C}_t$, si se puede pasar de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ aplicando las reglas de $\mathcal{R}$ de acuerdo con los siguientes principios:

- A un objeto arbitrario de una célula cualquiera sólo se le puede aplicar, a lo sumo, una regla (escogida, en su caso, de forma no determinista).
- A cada célula sólo se le puede aplicar o bien reglas de comunicación o bien una regla de división. En el caso de aplicarse reglas de comunicación

a la configuración $\mathcal{C}_t$, éstas se aplicarán de forma *no determinista*, paralela y *maximal*, de acuerdo con lo indicado previamente en los sistemas P básicos de tejidos. A la hora de aplicar una regla de división a la configuración $\mathcal{C}_t$, ésta será seleccionada, en su caso, de forma no determinista. Así pues, podemos imaginar que cuando se aplica una regla de división a una célula de una configuración $\mathcal{C}_t$, la propia célula queda bloqueada a efectos de poder comunicarse con otras células vecinas o con el entorno.

Las nuevas células resultantes de la división interactuarán con las otras células o con el entorno sólo en el siguiente paso de transición, siempre que no se vuelvan a dividir. Además, esas células tendrán las mismas etiquetas que la célula que se divide y, por tanto, proporcionarán nuevos arcos en el grafo subyacente al sistema.

4.3.3. Sistemas P de tejidos con separación celular

Los sistemas P de tejidos con separación celular se definen a partir de los sistemas P básicos de tejidos permitiendo el uso de reglas de separación celular, que están inspiradas en el fenómeno biológico de la fisión celular. Las reglas de separación celular fueron incorporadas a estos sistemas en [37].

Definición 4.8. *Un sistema P de tejidos con separación celular de grado* $q \geq 1$ *es una tupla* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$*, en donde:*

- Γ *es un alfabeto finito,* $\mathcal{E} \subseteq \Gamma$ *y* $\mathcal{M}_j$*,* $1 \leq j \leq q$*, son multiconjuntos sobre* Γ*.*
- $\{\Gamma_0, \Gamma_1\}$ *es una partición de* Γ*; es decir,* $\Gamma = \Gamma_0 \cup \Gamma_1, \Gamma_0 \neq \emptyset, \Gamma_1 \neq \emptyset$ *y* $\Gamma_0 \cap \Gamma_1 = \emptyset$*.*
- $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ *de la forma:*
  - *(a) Reglas de comunicación symport/antiport:* $(i, u/v, j)$*, siendo* $i, j \in \{0, 1, \ldots, q\}, i \neq j, u, v \in M(\Gamma), |u| + |v| > 0$*.*
  - *(b) Reglas de separación celular:* $[\,a\,]_i \rightarrow [\,\Gamma_0\,]_i [\,\Gamma_1\,]_i$*, siendo* $i \in \{1, 2, \ldots, q\}, i \neq i_{out}$ *y* $a \in \Gamma$*.*

En lo que respecta a la semántica de los sistema P de tejidos con separación celular, los conceptos de configuración (teniendo presente que en el nuevo marco pueden existir distintas células con la misma etiqueta) y computación

se definen de manera similar como se hizo con los sistemas P básicos de tejidos. El punto clave radica en definir cómo se implementa un *paso de transición* de una configuración del sistema a una configuración siguiente. Para ello, hay que definir previamente el concepto de aplicabilidad de las reglas del sistema a una configuración, así como el efecto que produce la aplicación de dichas reglas. Nos centraremos en las reglas de separación.

Una regla de separación $[\,a\,]_i \rightarrow [\,\Gamma_0\,]_i\,[\,\Gamma_1\,]_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una célula que está etiquetada por i, distinta de la célula de salida si la hubiera y, además, contiene al objeto a. La ejecución de dicha regla a esa célula de $\mathcal{C}_t$ produce la creación de dos nuevas células con la misma etiqueta i de tal manera que el objeto a que dispara la regla desaparece y, además, los objetos de la célula de $\mathcal{C}_t$ a la que se ha aplicado la regla y pertenecen a Γ_0 (respectivamente, Γ_1) irán a parar a la primera (respectivamente, a la segunda) de las células creadas. En la figura 4.4 se puede apreciar una representación de la regla de separación celular.

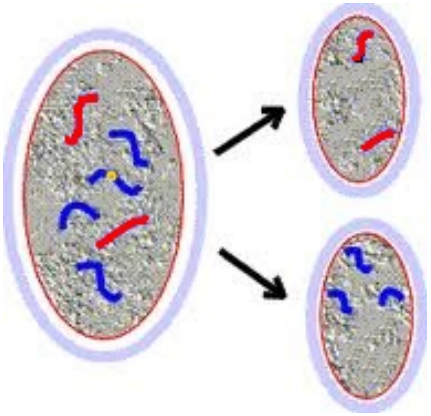


Figura 4.4: Regla de separación celular.

En este punto conviene resaltar un hecho importante: mientras que en la aplicación de las reglas de división se produce una *replicación* de objetos en las nuevas células creadas, en la aplicación de las reglas de separación se produce, simplemente, una *distribución* de objetos.

Se dirá que una configuración $\mathcal{C}_t$ del sistema P de tejidos con separación elular Π produce una configuración $\mathcal{C}_{t+1}$ en un *paso de transición*, lo denotaremos por $\mathcal{C}_t \Rightarrow_\Pi \mathcal{C}_{t+1}$ y diremos que $\mathcal{C}_{t+1}$ es *una configuración siguiente* de

$\mathcal{C}_t$, si se puede pasar de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ aplicando las reglas de $\mathcal{R}$ de acuerdo con los siguientes principios:

- A un objeto arbitrario de una célula cualquiera sólo se le puede aplicar, a lo sumo, una regla (escogida, en su caso, de forma no determinista).
- A cada célula sólo se le puede aplicar o bien reglas de comunicación o bien una regla de separación. En el caso de aplicarse reglas de comunicación a la configuración $\mathcal{C}_t$, éstas se aplicarán de forma *no determinista*, paralela y *maximal*, de acuerdo con lo indicado previamente en los sistemas P básicos de tejidos. A la hora de aplicar una regla de separación a la configuración $\mathcal{C}_t$, ésta será seleccionada, en su caso, de forma no determinista. Así pues, podemos imaginar que cuando se aplica una regla de separación a una célula de una configuración $\mathcal{C}_t$, la propia célula queda bloqueada a efectos de poder comunicarse con otras células vecinas o con el entorno.

Las nuevas células resultantes de la separación interactuarán con las otras células o con el entorno sólo en el siguiente paso de transición, siempre que no se vuelvan a separar. Además, esas células tendrán las mismas etiquetas que la célula que se separa y proporcionarán nuevos arcos en el grafo subyacente al sistema.

4.3.4. Sistemas P de tejidos con reglas de comunicación evolutiva

En los sistemas P de tejidos con reglas symport/antiport, la comunicación entre las células del sistema o las células y el entorno, se produce de tal manera que los objetos son «trasladados» de una zona a otra sin que éstos tengan la posibilidad de evolucionar a lo largo de dicho «desplazamiento». Los sistemas P de tejidos con reglas de comunicación evolutiva fueron introducidos en [60] y es una generalización de los sistemas P de tejidos con reglas symport/antiport, en los cuales se permite la evolución de los objetos al aplicar las reglas de comunicación del sistema. En [60], se ha estudiado la eficiencia computacional de dichos sistemas en el caso de que, además, se permiten reglas de división celular.

Definición 4.9. *Un sistema P de tejidos con reglas de comunicación evolutiva y división celular de grado* $q \geq 1$ *es una tupla* $\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$*, en donde:*

- Γ *es un alfabeto finito,* $\mathcal{E} \subseteq \Gamma$ *y* $\mathcal{M}_j, 1 \leq j \leq q$*, son multiconjuntos sobre* Γ*;*
- $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ *de la forma:*
  - *Reglas de comunicación evolutiva de tipo symport:* $[\, u\,]_i [\quad]_j \rightarrow [\quad]_i [\, u'\,]_j$*, siendo* $0 \leq i, j \leq q, i \neq j, u \in M_f^+(\Gamma), u' \in M_f(\Gamma)$*;*
  - *Reglas de comunicación evolutiva de tipo antiport:* $[\, u\,]_i [\, v\,]_j \rightarrow [\, v'\,]_i [\, u'\,]_j$*, where* $0 \leq i, j \leq q, i \neq j, u, v \in M_f^+(\Gamma)$*,* $u', v' \in M_f(\Gamma)$*;*
  - *Reglas de división celular:* $[\, a\,]_i \rightarrow [\, b\,]_i [\, c\,]_i$*, siendo* $i \in \{1, \ldots, q\}, i \neq i_{out}$ *y* $a, b, c \in \Gamma$*;*
- $i_{out} \in \{0, 1, \ldots, q\}$.

Un sistema P de tejidos con reglas de comunicación evolutiva y división celular de grado $q \geq 1$ $\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$, puede ser considerado como un conjunto de q células etiquetadas por $1, \ldots, q$ tales que: (a) $\mathcal{E}$ representa el conjunto de objetos colocados inicialmente en el entorno, cada uno de ellos con un número arbitrario (teóricamente infinito) de copias; (b) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ representa los multiconjuntos de objetos colocados inicialmente en las q células del sistema; y (c) i_{out} representa una *zona* distinguida que codificará la salida del sistema. Usaremos el término *zona* i $(0 \leq i \leq q)$ para referirnos a la célula i, en el caso $1 \leq i \leq q$, o bien al entorno, en el caso $i_{out} = 0$.

Una *configuración* en un instante t de un tal sistema está descrita por los multiconjuntos de objetos de Γ colocados en cada célula, en ese instante, así como el multiconjunto de objetos de $\Gamma \setminus \mathcal{E}$ colocados en el entorno, en ese instante. La configuración inicial de $\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$ es $(\mathcal{M}_1, \ldots, \mathcal{M}_q, \emptyset)$.

Una regla de comunicación evolutiva de tipo symport $[\, u\,]_i [\quad]_j \rightarrow [\quad]_i [\, u'\,]_j$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una zona que está etiquetada por i y contiene el multiconjunto u. La ejecución de la regla $[\, u\,]_i [\quad]_j \rightarrow [\quad]_i [\, u'\,]_j$ a una tal zona de $\mathcal{C}_t$ produce los siguientes efectos: los objetos del multiconjunto representado por u son consumidos y los objetos del multiconjunto representado por u' son enviados desde la zona i hasta la zona j en $\mathcal{C}_t$. Una regla de comunicación evolutiva de tipo antiport $[\, u\,]_i [\, v\,]_j \rightarrow [\, v'\,]_i [\, u'\,]_j$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en en dicha configuración existe, al menos, una zona que está

etiquetada por i y contiene el multiconjunto u y una zona j que contiene el multiconjunto v. La ejecución de la regla $[\,u\,]_i\,[\,v\,]_j \rightarrow [\,v'\,]_i\,[\,u'\,]_j$ a una tal zona de $\mathcal{C}_t$ produce los siguientes efectos: los objetos de los multiconjuntos representados por u y v son consumidos; los objetos del multiconjunto representado por u' son enviados desde la zona i hasta la zona j en $\mathcal{C}_t$ y los objetos del multiconjunto representado por v' son enviados desde la zona j hasta la zona i en $\mathcal{C}_t$. Téngase presente que la regla $[\,u\,]_i\,[\quad]_j \rightarrow [\quad]_i\,[\,u'\,]_j$ se puede considerar como un caso particular de la regla $[\,u\,]_i\,[\,v\,]_j \rightarrow [\,v'\,]_i\,[\,u'\,]_j$ (tómese $v = v' = \lambda$).

Por su parte, los conceptos de aplicabilidad de las reglas de división y su ejecución se definen de manera similar al caso de los sistemas P de tejidos ordinarios.

En [60] se define la *longitud* (o *tamaño*) de una regla de comunicación evolutiva $[\,u\,]_i\,[\,v\,]_j \rightarrow [\,v'\,]_i\,[\,u'\,]_j$ como sigue: $length(r) = |u| + |v| + |u'| + |v'|$. No obstante, a lo largo de este trabajo, usaremos también otro concepto de longitud, introducido en [39]. Específicamente, la longitud de una tal regla r será un par ordenado de números naturales, definido como sigue: $length'(r) = (|u| + |v|, |u'| + |v'|)$.

Se dirá que una configuración $\mathcal{C}_t$ del sistema P de tejidos reglas de comunicación evolutiva y división celular Π produce una configuración $\mathcal{C}_{t+1}$ en un *paso de transición*, lo denotaremos por $\mathcal{C}_t \Rightarrow_\Pi \mathcal{C}_{t+1}$ y diremos que $\mathcal{C}_{t+1}$ es *una configuración siguiente* de $\mathcal{C}_t$, si se puede pasar de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ aplicando las reglas de $\mathcal{R}$ de acuerdo con los siguientes principios:

- A un objeto arbitrario de una célula cualquiera sólo se le puede aplicar, a lo sumo, una regla (escogida de forma no determinista).

- A cada célula sólo se le puede aplicar o bien reglas de comunicación o bien una regla de división. En el caso de aplicarse reglas de comunicación a la configuración $\mathcal{C}_t$, éstas se aplicarán de forma *no determinista*, paralela y *maximal*, de acuerdo con lo indicado previamente en los sistemas P básicos de tejidos. En el caso de aplicarse una regla de división a la configuración $\mathcal{C}_t$, ésta será seleccionada de forma no determinista. Así pues, podemos imaginar que cuando se aplica una regla de división a una célula de una configuración $\mathcal{C}_t$, la propia célula queda bloqueada a efectos de poder comunicarse con otras células vecinas o con el entorno.

Las nuevas células resultantes de la división interactuarán con las otras células o con el entorno sólo en el siguiente paso de transición, siempre que no se vuelvan a dividir. Además, esas células tendrán las mismas etiquetas que

la célula que se divide y, por tanto, proporcionarán nuevos arcos en el grafo subyacente al sistema.

A partir del concepto de paso de transición se define, de manera natural, el concepto de computación del sistema.

Definición 4.10. *Un sistema P de tejidos con reglas de comunicación evolutiva y separación celular, de grado* $q \geq 1$*, es una tupla* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$*, en donde:*

- Γ *es un alfabeto finito,* $\mathcal{E} \subseteq \Gamma$ *y* $\mathcal{M}_j, 1 \leq j \leq q$*, son multiconjuntos sobre* Γ*;*
- $\{\Gamma_0, \Gamma_1\}$ *es una partición de* Γ*; es decir,* $\Gamma = \Gamma_0 \cup \Gamma_1, \Gamma_0, \Gamma_1 \neq \emptyset, \Gamma_0 \cap \Gamma_1 = \emptyset$*;*
- $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ *de la forma:*
  - *Reglas de comunicación evolutiva de tipo symport:* $[\,u\,]_i[\quad]_j \rightarrow [\quad]_i[\,u'\,]_j$*, siendo* $0 \leq i, j \leq q, i \neq j, u \in M_f^+(\Gamma), u' \in M_f(\Gamma)$*;*
  - *Reglas de comunicación evolutiva de tipo antiport:* $[\,u\,]_i[\,v\,]_j \rightarrow [\,v'\,]_i[\,u'\,]_j$*, siendo* $0 \leq i, j \leq q, i \neq j, u, v \in M_f^+(\Gamma)$*,* $u', v' \in M_f(\Gamma)$*;*
  - *Reglas de separación celular:* $[\,a\,]_i \rightarrow [\,\Gamma_0\,]_i[\,\Gamma_1\,]_i$*, siendo* $i \in \{1, 2, \ldots, q\}$*,* $i \neq i_{out}$ *y* $a \in \Gamma$*.*
- $i_{out} \in \{0, 1, \ldots, q\}$*.*

Los conceptos de aplicabilidad de las reglas de comunicación evolutiva de tipo symport y antiport, así como de su ejecución, han sido definidos previamente. Por su parte, los conceptos de aplicabilidad de las reglas de separación y su ejecución se definen de manera similar al caso de los sistemas P de tejidos ordinarios.

Con respecto a la semántica de esta variante, las reglas serán aplicadas de forma no determinista y maximal con la siguiente restricción: cuando una célula etiquetada por i es separada, entonces la regla de separación será la única regla de $\mathcal{R}_i$ que se aplicará a esa célula en ese paso de computación. Las nuevas células resultantes de la separación interactuarán con las otras células o con el entorno sólo en el siguiente paso de transición, siempre que no se vuelvan a separar. Además, esas células tendrán las mismas etiquetas que la célula que se separa y, por tanto, proporcionarán nuevos arcos en el grafo subyacente al sistema.

4.4. Complejidad computacional en Membrane Computing

El objetivo de esta sección es introducir los elementos básicos de una teoría de la complejidad computacional en el paradigma de la computación celular con membranas.

Para definir los primeros conceptos de complejidad computacional, se ha de tener presente que la resolución de problemas abstractos de decisión equivalen al reconocimiento de los lenguajes formales asociados a los mismos. Por tanto, en primer lugar, será necesario considerar nuevos modelos de computación celular con membranas, denominados *reconocedores*, que tengan la habilidad de reconocer lenguajes sobre alfabetos arbitrarios. Es importante hacer notar que los dispositivos computacionales en el paradigma de la computación celular son no deterministas y, por ello, será importante establecer diferencias significativas con las clásicas máquina de Turing no deterministas.

Recordemos que la resolución de problemas de decisión mediante MTNDs se materializa mediante el concepto de *aceptación de una instancia* del problema. Específicamente, una instancia de un problema de decisión es aceptada por una MTND si existe, al menos, una computación de la máquina con ese dato de entrada, que es de aceptación; es decir, que es de parada y devuelve *sí*. En este punto, conviene hacer notar que esta noción clásica de aceptación de un dato de entrada por MTNDs *no captura el verdadero concepto de algorítmo* [16], en el sentido de que el resultado de una computación de una tal máquina no será *fiable*, en general. Sólo lo será cuando la respuesta de la misma sea afirmativa. Por ello, la definición que se va a presentar de aceptación de un dato de entrada por un sistema de membranas va a diferir sustancialmente del concepto de aceptación antes reseñado para MTNDs.

4.4.1. Sistemas de membranas reconocedores

A lo largo de este trabajo, el término *sistema de membranas* será utilizado para referirnos, indistintamente, a cualquier sistema P que trabaja a modo de células o a modo de tejidos, introducidos en las secciones 2 y 3 de este capítulo.

Definición 4.11. *Un sistema de membranas reconocedor de orden* $q \geq 1$ *es una tupla*

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

en donde:

- Π *es un sistema P de membranas de orden* $q \geq 1$*. cuyo alfabeto de trabajo es* Γ *y cuyos multiconjuntos inciales son* $\mathcal{M}_1, \ldots, \mathcal{M}_q$*.*
- Γ *contiene dos elementos distinguidos* `yes` *y* `no`*.*
- Σ *es un alfabeto (denominado alfabeto de entrada) estrictamente contenido en* Γ*. En el caso de sistemas de membranas que trabajan a modo de tejidos, el alfabeto* $\mathcal{E}$ *del entorno ha de ser un subconjunto de* $\Gamma \setminus \Sigma$*.*
- $\mathcal{M}_1, \ldots, \mathcal{M}_q$ *son multiconjuntos sobre* $\Gamma \setminus \Sigma$ *de tal manera que, al menos una copia del símbolo* `yes` *o del símbolo* `no` *está presente en alguno de esos multiconjuntos.*
- $i_{in} \in \{1, \ldots, q\}$ *es la etiqueta de una membrana ó célula distinguida (membrana o célula de entrada).*
- $i_{out} = 0$*; es decir, la zona de salida es el entorno del sistema.*
- *Todas las computaciones de* Π *son de parada.*
- *Si* $\mathcal{C}$ *es una computación de* Π *entonces, o bien el símbolo* `yes` *o bien el símbolo* `no` *(pero no ambos) son enviados al entorno y, además, en el último paso de la computación.*

Si $(\Pi, \Gamma, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$ es un sistema de membranas reconocedor, entonces para cada multiconjunto m sobre el alfabeto de entrada Σ se considera el sistema de membranas Π con multiconjunto de entrada m. Dicho sistema se notará por $\Pi + m$ y está caracterizado por el hecho de que los multiconjuntos asociados a la configuración inicial del mismo es la siguiente: $\mathcal{M}_1, \ldots, \mathcal{M}_{i_{in}} + m, \ldots, \mathcal{M}_p$. Es decir, dicha configuración se obtiene a partir de la configuración inicial $(\mathcal{M}_1, \ldots, \mathcal{M}_{i_{in}}, \ldots, \mathcal{M}_q)$ de Π, añadiendo el multiconjunto m al multiconjunto inicial $\mathcal{M}_{i_{in}}$.

A continuación se define una función *Output* cuyo dominio es el conjunto de las computaciones de Π. Dicha función formalizará los resultados o salidas de las computaciones del sistema. Si $\mathcal{C}$ es una computación del sistema Π, entonces $Output(\mathcal{C}) =$ `yes` (respectivamente, $Output(\mathcal{C}) =$ `no`) si el objeto `yes` (resp. el objeto `no`) aparece en el entorno asociado a la configuración de parada de $\mathcal{C}$ pero, en cambio, no aparece en el entorno asociado a cualquier configuración de $\mathcal{C}$ que no sea de parada. Se dirá que una computación $\mathcal{C}$ del sistema Π es de *aceptación* (respectivamente, de *rechazo*) si $Output(\mathcal{C}) =$ `yes` (respectivamente, $Output(\mathcal{C}) =$ `no`).

Es interesante hacer notar que en los sistemas de membranas reconocedores todas las computaciones son o bien de aceptación o bien de rechazo. Ahora bien, en principio, a partir de un multiconjunto de entrada m pueden existir computaciones que sean de aceptación y otras que sean de rechazo. Si en un sistema de membranas reconocedor Π, para un cierto multiconjunto m de entrada se verifica que o bien todas las computaciones de $\Pi + m$ son de aceptación, o bien todas las computaciones de $\Pi + m$ son de rechazo, entonces se dice que el sistema de membranas reconocedor $\Pi + m$ es *confluente*. Precisamente, los sistemas de membranas reconocedores que son confluentes *capturan el verdadero concepto de algoritmo* en el sentido de que, a pesar de ser dispositivos no deterministas, el resultado de cualquiera de las computaciones de un tal sistema con multiconjunto de entrada m, será *fiable*, en tanto que su respuesta coincidirá con la de cualquier otra computación arbitraria del mismo que esté asociada a dicho multiconjunto de entrada m.

4.4.2. Clases de complejidad polinomial en sistemas de membranas

Los sistemas de membranas son dispositivos computacionales tales que tienen una descripción finita; es decir, existe una cantidad inicial fija de recursos iniciales, expresada a través del número de membranas o células, de los tamaños de los alfabetos y de los conjuntos de reglas. Por ello, a diferencia de lo que sucede con las máquinas de Turing (deterministas o no deterministas) en donde existe una cinta con infinitas celdas o casillas, para resolver un problema de decisión (que, usualmente, consta de un número infinito de instancias), será necesario una familia infinita numerable de sistemas de membranas, de tal manera que cada uno de esos sistemas se encarguen de procesar un número finito de instancias del problema.

Seguidamente, se va a introducir el concepto de *tratabilidad* de problemas de decisión en Membrane Computing, a través de sistemas de membranas reconocedores. Dicho con otras palabras, se va a definir qué significa *resolver eficientemente* (es decir, en tiempo polinomial) un problema de decisión mediante dicho tipo de sistemas. Para ello, conviene precisar, previamente, algunos conceptos importantes.

Definición 4.12. *Se dice que una familia* $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ *de sistemas de membranas reconocedores es* polinomialmente uniforme por máquinas de Turing *si existe una MTD que trabaja en tiempo polinomial y construye el sistema* $\Pi(n)$ *a partir del número natural* $n \in \mathbb{N}$*, expresado en codificación 1-aria.*

A la hora de definir el concepto de resolución de un problema de decisión $X = (I_X, \theta_X)$ a través de una familia de sistemas de membranas reconocedores, se trata de que los sistemas de la familia procesen un conjunto finito de instancias del problema; concretamente, todas aquellas instancias que posean un determinado «tamaño fijo» que será fijado por una cierta función computable s. En tales circunstancias, un tal sistema procesará todas las instancias $u \in I_X$ con el mismo «tamaño» siempre que dicho sistema trabaje con un multionjunto de entrada que será codificado/obtenido a través de otra función computable, que notaremos por *cod*. Esta idea justifica la siguiente definición de lo que entenderemos por *codificación en tiempo polinomial* de un problema de decisión en una familia de sistemas de membranas reconocedores.

Definición 4.13. *Sea $X = (I_X, \theta_X)$ un problema de decisión y $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ una familia de sistemas de membranas reconocedores. Una* codificación polinomial *del problema X en la familia $\mathbf{\Pi}$ es un par ordenado (cod, s) de funciones computables en tiempo polinomial sobre I_X tal que: (a) para cada instancia $u \in I_X$, $s(u)$ es un número natural (obtenido por medio de un esquema de codificación razonable); (b) para cada $k \in \mathbb{N}$, el conjunto $s^{-1}(k)$ es finito; y (c) para cada $u \in I_X$, $cod(u)$ es un multiconjunto de entrada del sistema $\Pi(s(u))$.*

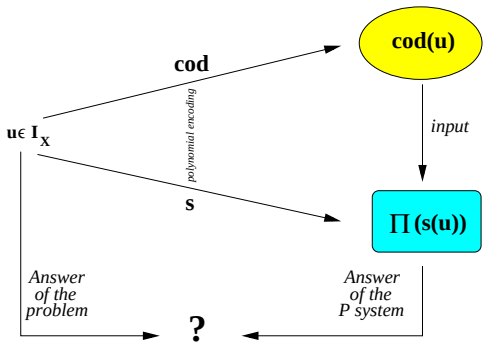


Figura 4.5: Codificación polinomial de un problema en una familia de sistemas de membranas

Proposición 4.1. *Si X_1, X_2 son problemas de decisión, f es una reducibilidad en tiempo polinomial de X_1 en X_2, y (cod, s) es una codificación polinomial de X_2 en la familia $\mathbf{\Pi}$. Entonces, $(cod \circ f, s \circ f)$ es una codificación polinomial de X_1 en la familia $\mathbf{\Pi}$.*

Para más detalles, véase [51].

Definición 4.14. *Sea $X = (I_X, \theta_X)$ un problema de decisión, $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ una familia de sistemas de membranas reconocedores y (cod, s) una codificación polinomial del problema X en la familia $\mathbf{\Pi}$.*

- *Diremos que la familia $\mathbf{\Pi}$ es adecuada con respecto a (X, cod, s) si para cada instancia $u \in I_X$ tal que existe, al menos, una computación de aceptación del sistema $\Pi(s(u)) + cod(u)$, se tiene que $\theta_X(u) = 1$.*
- *Diremos que la familia $\mathbf{\Pi}$ es completa con respecto a (X, cod, s) si para cada instancia $u \in I_X$ tal que $\theta_X(u) = 1$, se tiene que toda computación del sistema $\Pi(s(u)) + cod(u)$ es una computación de aceptación.*

Obsérvese que si una familia $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ de sistemas de membranas reconocedores es adecuada y completa con respecto a (X, cod, s), entonces para cada instancia $u \in I_X$ se tiene que el sistema $\Pi(s(u)) + cod(u)$ es confluente.

A continuación se define el concepto de resolubilidad uniforme y eficiente (en tiempo polinomial) de un problema de decisión en el marco de Membrane Computing.

Definición 4.15. *Sea $\mathcal{R}$ una clase arbitraria de sistemas de membranas reconocedores. Diremos que un problema de decisión $X = (I_X, \theta_X)$ es resoluble de manera uniforme y en tiempo polinomial, por una familia $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ de sistemas de $\mathcal{R}$, y lo notaremos por $X \in \mathbf{PMC}_{\mathcal{R}}$, si se verifica las siguiente condiciones:*

(a) La familia $\mathbf{\Pi}$ es polinomialmente uniforme por maquinas de Turing.

(b) Existe una codificación polinomial (cod, s) de X en $\mathbf{\Pi}$ tal que:

(b_1) La familia $\mathbf{\Pi}$ polinomialmente acotada con respecto a (X, cod, s); es decir, existe un número natural $k \in \mathbb{N}$ tal que para cada instancia $u \in I_X$, toda computación de $\Pi(s(u)) + cod(u)$ realiza, a lo sumo, $|u|^k$ pasos de computación.

(b_2) La familia $\mathbf{\Pi}$ es adecuada y completa con respecto a (X, cod, s).

Es interesante hacer notar que si $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ es una familia de sistemas reconocedores que resuelve un problema de decisión X, entonces, para cada instancia u del problema se verifica que el sistema $\Pi(s(u)) + cod(u)$ es confluente. De este hecho se deduce, a su vez, de forma inmediata, que la clase de complejidad $\mathbf{PMC}_{\mathcal{R}}$ es cerrada bajo complementario. Además, se puede

demostrar que la clase de complejidad $\mathbf{PMC}_{\mathcal{R}}$ es cerrada bajo reducibilidad en tiempo polinomial; es decir, si X_1 y X_2 son problemas de decisión tales que X_1 es reducible en tiempo polinomial a X_2 y, además, $X_2 \in \mathbf{PMC}_{\mathcal{R}}$, entonces se tiene que $X_1 \in \mathbf{PMC}_{\mathcal{R}}$ (para más detalle, véase [51]).

Por otra parte, recordemos que si una MTND, M, resuelve un problema de decisión $X = (I_X, \theta_X)$ entonces para cada instancia $u \in I_X$ se tiene que $\theta_X(u) = 1$ si y sólo si existe, al menos, una computación de la máquina M con dato de entrada u que es de aceptación. En cambio, si una familia $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ de sistemas de membranas reconocedores resuelve un problema de decisión $X = (I_X, \theta_X)$ y (cod, s) es una codificación polinomial de X en $\mathbf{\Pi}$, de acuerdo con dicha definición, entonces para cada instancia $u \in I_X$ se tiene que $\theta_X(u) = 1$ si y sólo si toda computación del sistema $\Pi(s(u)) + cod(u)$ es de aceptación. Como se puede observar, se han establecido diferencias significativas en lo que respecta a la definición del concepto de *aceptación* de una instancia de un problema de decisión por medio de sistemas de membranas reconocedoras, en relación con la definición clásica dada para MTNDs.

Proposición 4.2. *Si $\mathcal{R}$ es una clase arbitraria de sistemas de membranas reconocedores entonces se verifica que $\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{R}}$.*

Demostración. Sea $X = (I_X, \theta_X)$ un problema de decisión que pertenece a la clase de complejidad $\mathbf{P}$. Entonces existe una MTD, M, que trabaja en tiempo polinomial y resuelve el problema X; es decir, para cada instancia $u \in I_X$ se tiene que $\theta_X(u) = 1$ si y sólo si la máquina M con dato de entrada u para y devuelve `yes`.

Consideremos una familia $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ de sistemas de membranas de la clase $\mathcal{R}$ en donde el el sistema $\Pi(1)$ está descrito como sigue:

- En el caso de sistemas que trabajan a modo de célula se considera una única membrana (etiquetada por 1) cuyo alfabeto de trabajo es $\Gamma = \{\mathtt{yes}, \mathtt{no}\}$ y las únicas reglas del sistema son $[\,\mathtt{yes}\,]_1 \to \mathtt{yes}[\;\;]_1$ y $[\,\mathtt{no}\,]_1 \to \mathtt{no}[\;\;]_1[\,\mathtt{no}\,]_1$.
- En el caso de sistemas que trabajan a modo de tejidos se considera una única célula (etiquetada por 1) cuyo alfabeto de trabajo es $\Gamma = \{\mathtt{yes}, \mathtt{no}\}$ y las únicas reglas del sistema son $(1, \mathtt{yes}/\lambda, 0)$ y $(1, \mathtt{no}/\lambda, 0)$.

Se considera la codificación polinomial (cod, s) del problema X en la familia $\mathbf{\Pi}$ definida como sigue: para cada instancia $u \in I_X$,

$$s(u) = 1 \quad \text{y} \quad cod(u) = \begin{cases} \mathtt{yes} & \text{si} \quad M(u) \text{ es una computación de aceptación} \\ \mathtt{no} & \text{si} \quad M(u) \text{ es una computación de rechazo} \end{cases}$$

En tal situación, para cada instancia $u \in I_X$ el sistema $\Pi(1) + cod(u)$ encargado de procesar u verifica lo siguiente: (a) si $M(u)$ es una computación de aceptación, entonces la membrana/célula etiquetada por 1 contiene inicialmente el multiconjunto $\{\texttt{yes}\}$ y, por tanto, el sistema para tras realizar un paso de computación, respondiendo `yes`; y (b) si $M(u)$ es una computación de rechazo, entonces la membrana/célula etiquetada por 1 contiene inicialmente el multiconjunto $\{\texttt{no}\}$ y, por tanto, el sistema para tras realizar un paso de computación, respondiendo `no`. En consecuencia, la familia $\mathbf{\Pi}$ así definida resuelve el problema X de manera uniforme y en un solo paso de transición. □

Capítulo 5

Fronteras de la eficiencia en computación celular con membranas

En el capítulo 3 se describió una nueva metodología para atacar el problema **P** *versus* **NP** basada en la obtención de fronteras de la eficiencia de modelos de computación, expresadas en términos de ingredientes sintácticos y/o semánticos de dichos modelos. El objetivo de este capítulo consiste en presentar los resultados más importantes que ya se han establecidos, en relación con las fronteras de la eficiencia en el paradigma de la computación celular con membranas.

5.1. Sistemas P que trabajan a modo de células

En esta sección se van a detallar las fronteras antes citadas que ya han sido obtenidas en sistemas de membranas reconocedores que trabajan a modo de células.

5.1.1. Sistemas P básicos de transición

En [17] se ha establecido que las máquinas de Turing deterministas pueden ser simuladas eficientemente a través de familias de sistemas P básicos de transición reconocedores (cuya clase se denota por $\mathcal{T}$). Así pues, se tiene que $\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{T}}$. Además, en la referencia antes citada se ha probado que si un

problema de decisión es resoluble en tiempo polinomial por una familia de sistemas de membranas de la clase $\mathcal{T}$, entonces existe una máquina de Turing determinista que resuelve dicho problema en tiempo polinomial; es decir, se ha establecido que $\mathbf{PMC}_{\mathcal{T}} \subseteq \mathbf{P}$. En consecuencia, resulta que $\mathbf{PMC}_{\mathcal{T}} = \mathbf{P}$ y, por tanto, los sistemas P básicos de transición reconocedores proporcionan modelos de computación que son no eficientes.

5.1.2. Sistemas P con membranas activas

En [70] se ha probado que los sistemas P reconocedores con membranas activas y con cargas eléctricas que no usan reglas de división (cuya clase se representa por $\mathcal{NAM}$), son modelos de computación no eficientes; es decir, se ha establecido que $\mathbf{PMC}_{\mathcal{NAM}} = \mathbf{P}$. Por otra parte, en [51, 47, 48], entre otras referencias, se han proporcionado soluciones uniformes en tiempo polinomial de problemas **NP**-completos a través de familias de sistemas P reconocedores con membranas activas y con cargas eléctricas que usan reglas de división pero sólo para membranas elementales (cuya clase se representa por $\mathcal{DAM}(-ne)$). Además, si en el marco de los sistemas P reconocedores con membranas activas y con cargas eléctricas se usan reglas de separación de membranas elementales, en lugar de reglas de división de membranas, como mecanismo para construir eficientemente una cantidad exponencial de espacio (expresada en el número de objetos y el número de membranas), entonces también es posible dar soluciones uniformes en tiempo polinomial de problemas **NP**-completos a través de familias de dichos sistemas (en [36] se ha proporcionado una tal solución para el problema `SAT`).

Así pues, en el marco de los sistemas P reconocedores con membranas activas y con cargas eléctricas, pasar de prohibir el uso de reglas de división y de separación para membranas elementales ($\mathcal{NAM}$) a permitir su uso ($\mathcal{DAM}(-ne)$ o $\mathcal{SAM}(-ne)$), proporciona una frontera entre la no eficiencia y la presumible eficiencia.

Por su parte, si el número de cargas eléctricas en los sistemas P reconocedores con membranas activas pasa de tres a dos, entonces no hay cambio alguno en lo que respecta a la no eficiencia o a la presumible eficiencia de dichos modelos computacionales (ver [2] para más detalles). Ahora bien, como se va a ver en la siguiente subsección, la situación es muy distinta cuando se pasa de usar tres cargas eléctricas a permitir sólo el uso de una carga, lo cual equivale, de hecho, a considerar sistemas P reconocedores con membranas activas y sin polarizaciones.

5.1.3. Sistemas P con membranas activas y sin cargas eléctricas

Sea $\mathcal{DAM}^0$ la clase de los sistemas P reconocedores con membranas activas, sin polarizaciones y que usan reglas de división de membranas. En esta subsección se notará por $\mathcal{DAM}^0(\alpha, \beta)$ la clase de todos los sistemas del tipo antes citado y tales que:

(a) Si $\alpha = +d$ (respectivamente $\gamma = -d$) entonces se permiten las reglas de disolución (respectivamente, se prohíben).

(b) Si $\beta = +ne$ (respectivamente, $-ne$) entonces se permiten reglas de división para membranas elementales y no elementales (respectivamente, sólo para membranas elementales).

A comienzos de 2005, Gh. Păun escribió lo siguiente (problema **F** de [44]):

My favorite question (related to complexity aspects in P systems with active membranes and with electrical charges) is that about the number of polarizations. Can the polarizations be completely avoided? The feeling is that this is not possible – and such a result would be rather sound: passing from no polarization to two polarizations amounts to passing from non–efficiency to efficiency.

Es decir, la impresión del profesor Păun es que en el marco inicial de los los sistemas P reconocedores con membranas activas (que sólo usan reglas de división para membranas elementales) pasar de no usar ninguna carga eléctrica a usar dos significa pasar de la no eficiencia a la eficiencia (bajo el supuesto de que $\mathbf{P} \neq \mathbf{NP}$). Esta es la denominada *conjetura de Păun* y que, formalmente, puede ser formulada, en términos de clases de complejidad de sistemas de membranas como sigue: $\mathbf{P} = \mathbf{PMC}_{\mathcal{DAM}^0(+d,-ne)}$.

Por tanto, admitiendo que $\mathbf{P} \neq \mathbf{NP}$, una *respuesta afirmativa* a dicha cuestión indicaría que la habilidad de los sistemas de membranas de $\mathcal{DAM}^0(+d, -ne)$ para crear una cantidad exponencial de espacio (expresado en términos del número de objetos y el número de membranas) en tiempo polinomial, no sería suficiente para poder resolver de manera eficiente problemas computacionalmente duros. Por contra, una *respuesta negativa* a dicha cuestión, proporcionaría una frontera entre la tratabilidad y la intratabilidad de problemas: el uso o no de reglas de división para membranas elementales.

Es importante hacer notar que, hasta el momento en que se hizo pública la citada conjetura, el papel jugado por las reglas de disolución en el análisis

de la eficiencia computacional de sistemas de membranas, había sido meramente «testimonial». Hasta el punto de que, en el mismo año 2005, el creador de la disciplina Gh. Păun había afirmado, en un contexto informal, que las reglas de disolución eran *aparentemente inocente* desde el punto de vista de la complejidad computacional.

Los sistemas P reconocedores con membranas activas, sin cargas eléctricas, sin reglas de disolución pero que permiten reglas de división sólo para membranas no elementales y, además, el uso de catalizadores biestables (cuya clase se denota por $\mathcal{BAM}(-d,-ne)$) permiten resolver problemas **NP**-completos de manera uniforme y en tiempo polinomial (ver [50] para más detalles). Teniendo presente que $\mathbf{P} = \mathbf{PMC}_{\mathcal{DAM}^0(-d,+ne)}$, se deduce que en el marco de los sistemas P con membranas activas, sin cargas eléctricas, sin reglas de disolución y con reglas de división sólo para membranas elementales, pasar de prohibir el uso de catalizadores biestables a permitirlo $\mathcal{BAM}(-d,-ne)$, constituye una frontera entre la no eficiencia y la presumible eficiencia.

Los sistemas P reconocedores con membranas activas, sin polarizaciones y sin reglas de disolución que usan reglas que permiten el *cambio de etiqueta* en las membranas, son presumiblemente eficientes (ver [3] para más detalles). En consecuencia, en el marco de los sistemas P con membranas activas, sin cargas eléctricas y sin reglas de disolución, que sólo permiten reglas de división para membranas elementales, prohibir o permitir el uso de reglas cuya aplicación puede propiciar el cambio de las etiquetas de las membranas, proporciona una nueva frontera entre la no eficiencia y la presumible eficiencia.

En [18] se ha probado que $\mathbf{P} = \mathbf{PMC}_{\mathcal{DAM}^0(-d,+ne)}$, lo cual puede considerarse como una *respuesta parcial afirmativa* a la conjetura de Păun, en el sentido de que si se impone la condición adicional de prohibir reglas de disolución, entonces dicha conjetura es verdadera. En esta situación y teniendo en cuenta que hasta ese momento ese tipo de reglas había jugado un papel secundario desde el punto de vista de la eficiencia computacional, todo hacía pensar que dicha conjetura sería también verdadera en el caso de que se permitiera esas reglas. No obstante, en la referencia citada anteriormente se proporcionó una solución uniforme y en tiempo polinomial del problema **NP**-completo `Subset Sum` a través de una familia de sistemas reconocedores de $\mathcal{DAM}^0(+d,+ne)$. Este resultado puede ser interpretado como una *respuesta parcial negativa* a la conjetura de Păun, en el sentido de que si en el marco $\mathcal{DAM}^0(+d)$ se permite el uso de reglas de división de membranas no elementales, entonces la citada conjetura es falsa (bajo el supuesto de que $\mathbf{P} \neq \mathbf{NP}$).

Por tanto, en el marco de los sistemas P reconocedores con membranas activas y sin polarizaciones que permiten el uso de reglas de división para

membranas elementales y no elementales, pasar de prohibir el uso de reglas de disolución a permitirlo, proporciona una frontera entre la no eficiencia y la presumible eficiencia.

En la Figura 5.1 aparecen resumidas todas las fronteras de la tratabilidad de problemas que han sido obtenidas en los sistemas de membranas que trabajan a modo de células.

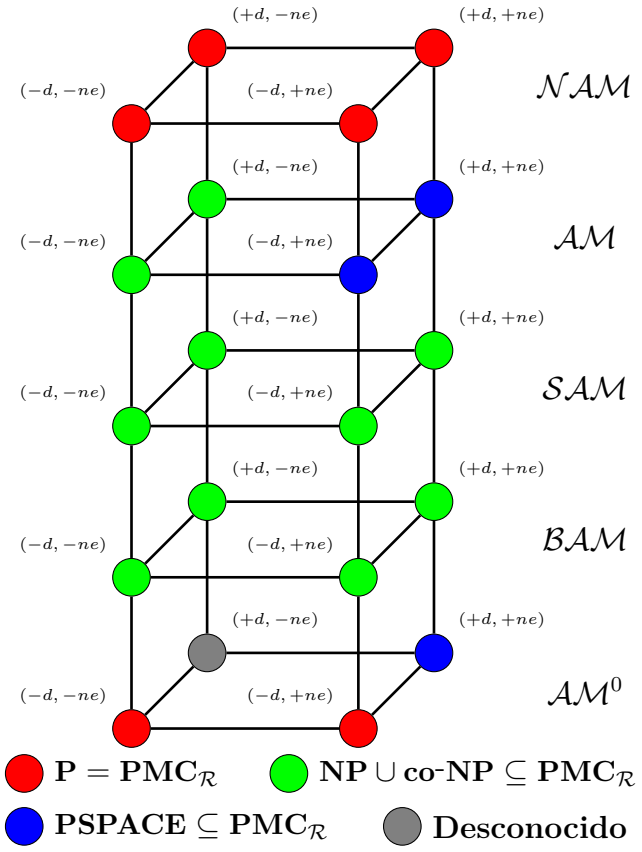


Figura 5.1: Fronteras de la eficiencia en sistemas P con membranas activas

5.1.4. Sistemas P con reglas symport/antiport

En esta sección se van a detallar las fronteras de la eficiencia que ya han sido obtenidas en sistemas de membranas reconocedores que trabajan a modo de células y sus reglas de comunicación son del tipo symport/antiport. Para ello, vamos a comenzar fijando las notaciones que van a ser utilizadas para sistemas P reconocedores.

- $\mathcal{CC}$: con reglas symport/antiport.
- $\mathcal{CDC}$: con reglas symport/antiport y reglas de división de membranas ($\widehat{\mathcal{CDC}}$, análogo pero sin entorno).
- $\mathcal{CSC}$: con reglas symport/antiport y reglas de separación de membranas ($\widehat{\mathcal{CSC}}$, análogo pero sin entorno).
- Para cada $k \geq 1$, $\mathcal{CDC}(k)$: con reglas de división de membranas y reglas de comunicación de longitud, a lo sumo k (análogo para $\widehat{\mathcal{CDC}}(k)$).
- Para cada $k \geq 1$, $\mathcal{CSC}(k)$: con reglas de separación de membranas y reglas de comunicación de longitud, a lo sumo, k (análogo para $\widehat{\mathcal{CSC}}(k)$).

En [28] se estableció que los sistemas P reconocedores con reglas de división o de separación de membranas y reglas de comunicación de longitud 1, son modelos de computación no eficientes. Por tanto, en particular se tiene que $\mathbf{PMC}_{\mathcal{CC}} = \mathbf{P}$. En [35, 67] se proporcionó una solución polinomial y uniforme del problema `HAM-CYCLE` mediante una familia de sistemas de membranas de $\mathcal{CDC}(2)$, justificando que esta clase de modelos de computación es presumiblemente eficiente. De ahí resulta que en el marco de los sistemas P reconocedores con reglas symport/antiport y reglas de división, una frontera entre la no eficiencia y la presumible eficiencia se obtiene al pasar de usar sólo reglas de comunicación de longitud 1, a permitir el uso de reglas de comunicación de longitud a lo sumo 2.

Sin embargo, la situación es bien distinta cuando en los sistemas P reconocedores que usan reglas de comunicación de longitud a lo sumo 2, se consideran reglas de separación de membranas en lugar de reglas de división; específicamente, en [68] se estableció que únicamente problemas de la clase **P** podían ser resuelto en tiempo polinomial y de manera uniforme mediante familias de sistemas de membranas de $\mathcal{CSC}(2)$. Teniendo presente que en [29] se proporcionó una solución polinomial y uniforme del problema `SAT` mediante una familia de sistemas de membranas de $\mathcal{CSC}(3)$, resulta que en el marco de los sistemas P reconocedores con reglas symport/antiport y reglas de separación, una frontera entre la no eficiencia y la presumible eficiencia se obtiene al pasar de usar sólo reglas de comunicación de longitud a lo sumo 2, a permitir el uso de reglas de comunicación de longitud a lo sumo 3.

En lo que respecta al papel que, desde el punto de vista de la complejidad computacional, juega el entorno en los sistemas P reconocedores con reglas symport/antiport cabe destacar dos resultados importantes. Por una parte,

en [34] se probó que el entorno es irrelevante cuando se usan reglas de división de membranas; específicamente, se demostró que para cada número natural $k \geq 1$ se tiene que $\mathbf{PMC}_{\mathcal{CDC}(k)} = \mathbf{PMC}_{\widehat{\mathcal{CDC}}(k)}$. Por otra parte, en [26] se estableció la relevancia del mismo cuando se usan reglas de separación de membranas; específicamente, se demostró que para cada número natural $k \geq 1$ se tiene que $\mathbf{P} = \mathbf{PMC}_{\widehat{\mathcal{CSC}}(k)}$. De estos resultados se deducen, fácilmente, nuevas fronteras entre la no eficiencia y la presumible eficiencia:

- Pasar de $\widehat{\mathcal{CSC}}(2)$ a $\widehat{\mathcal{CDC}}(2)$ (frontera expresada a través del tipo de reglas: separación *versus* división).
- Pasar de $\widehat{\mathcal{CSC}}(3)$ a $\mathcal{CSC}(3)$ (frontera expresada a través de la existencia o no del entorno).

En las Figuras 5.2 y 5.3 aparecen resumidas todas las fronteras de la tratabilidad de problemas que han sido obtenidas en los sistemas de membranas con reglas symport/antiport.

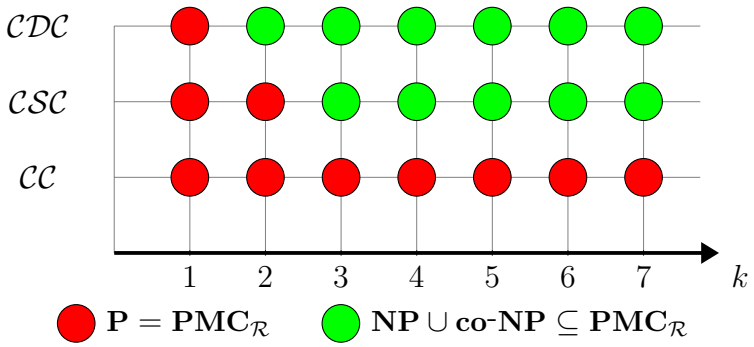


Figura 5.2: Fronteras de la eficiencia en sistemas P con reglas symport/antiport con entorno

5.2. Sistemas P que trabajan a modo de tejidos

En esta sección se van a detallar las fronteras de la eficiencia que ya han sido obtenidas en sistemas de membranas reconocedores que trabajan a modo de tejidos. Para ello, vamos a comenzar fijando las notaciones que van a ser utilizadas para dichos sistemas de membranas.

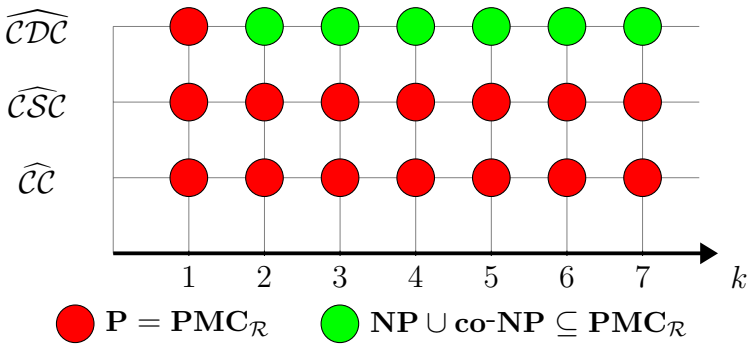


Figura 5.3: Fronteras de la eficiencia en sistemas P con reglas symport/antiport sin entorno

- $\mathcal{TC}$: básicos.
- $\mathcal{TDC}$: con reglas de división celular ($\widehat{\mathcal{TDC}}$: análogo, pero sin entorno).
- $\mathcal{TSC}$: con reglas de división celular ($\widehat{\mathcal{TSC}}$: análogo, pero sin entorno).
- Para cada $k \geq 1$, $\mathcal{TDC}(k)$: con reglas de división celular y reglas de comunicación de longitud, a lo sumo, k (análogo para $\widehat{\mathcal{TDC}}(k)$).
- Para cada $k \geq 1$, $\mathcal{TSC}(k)$: con reglas de división celular y reglas de comunicación de longitud, a lo sumo, k (análogo para $\widehat{\mathcal{TSC}}(k)$).

Para los sistemas P de tejidos reconocedores con reglas de comunicación evolutiva se utilizarán las siguientes notaciones:

- $\mathcal{TDEC}$: con reglas de división celular ($\widehat{\mathcal{TDEC}}$: análogo, pero sin entorno).
- $\mathcal{TSEC}$: con reglas de división celular ($\widehat{\mathcal{TSEC}}$: análogo, pero sin entorno).
- Para cada $k \geq 1$, $\mathcal{TDEC}(k)$: con reglas de división celular y reglas de comunicación evolutiva de longitud, a lo sumo, k (análogo para $\widehat{\mathcal{TDEC}}(k)$).
- Para cada $k \geq 1$, $\mathcal{TSEC}(k)$: con reglas de división celular y reglas de comunicación evolutiva de longitud, a lo sumo, k (análogo para $\widehat{\mathcal{TSEC}}(k)$).
- Para cada $k_1, k_2 \geq 1$, $\mathcal{TDEC}(k_1, k_2)$: con reglas de división celular y reglas de comunicación evolutiva cuya LHS tiene longitud, a lo sumo, k_1 y cuya RHS tiene longitud, a lo sumo, k_2 (análogo para $\widehat{\mathcal{TDEC}}(k_1, k_2)$).

- Para cada $k_1, k_2 \geq 1$, $\mathcal{TSEC}(k_1, k_2)$: con reglas de división celular y reglas de comunicación evolutiva cuya LHS tiene longitud, a lo sumo, k_1 y cuya RHS tiene longitud, a lo sumo, k_2 (análogo para $\widehat{\mathcal{TSEC}}(k_1, k_2)$).

5.2.1. Sistemas P de tejidos symport/antiport y con entorno

En [13] se estableció que los sistemas P básicos de tejidos reconocedores podían ser simulados eficientemente por sistemas P básicos de transición reconocedores (que trabajan a modo de células). Por tanto, resulta que $\mathbf{PMC}_{\mathcal{TC}} \subseteq \mathbf{PMC}_{\mathcal{T}}$. Ahora bien, teniendo presente que $\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{TC}}$ y que $\mathbf{P} = \mathbf{PMC}_{\mathcal{T}}$, se concluye finalmente que $\mathbf{P} = \mathbf{PMC}_{\mathcal{TC}}$. Así pues, los sistemas P básicos de tejidos reconocedores proporcionan modelos de computación no eficientes.

En lo que respecta a los sistemas P de tejidos reconocedores y con división celular, en [13] se ha probado la no eficiencia de dichos sistemas de tejidos cuando las reglas de comunicación no usan cooperación; es decir, se ha probado que $\mathbf{P} = \mathbf{PMC}_{\mathcal{TDC}(1)}$. Por otra parte, en [38] se ha probado la no eficiencia de los sistemas P de tejidos reconocedores y con separación celular que usan reglas de comunicación de longitud, a lo sumo, 2; es decir, $\mathbf{P} = \mathbf{PMC}_{\mathcal{TSC}(2)}$.

En lo que respecta a la presumible eficiencia de estos modelos de computación, por una parte, en [55] se ha proporcionado una solución uniforme en tiempo polinomial del problema `HAM-CYCLE` mediante familias de sistemas de $\mathcal{TDC}(2)$. Por otra, en [54] se ha proporcionado una solución uniforme en tiempo polinomial del problema `SAT` mediante familias de sistemas de $\mathcal{TSC}(3)$.

Por tanto, en el marco de los sistemas P de tejidos con división celular, el paso de usar reglas de comunicación no cooperativas a permitir reglas de comunicación de longitud a lo sumo 2, equivale a pasar de la no eficiencia a la presumible eficiencia. Por su parte, en el marco de los sistemas P de tejidos con separación celular, el paso de usar reglas de comunicación de longitud, a lo sumo 2, a permitir el uso de reglas de comunicación de longitud, a lo sumo 3, equivale pasar de la no eficiencia a la presumible eficiencia.

Más aún, en el marco de los sistemas P de tejidos reconocedores que usan reglas de comunicación de longitud a lo sumo 2, pasar de usar reglas de separación celular ($\mathcal{TSC}(2)$) a permitir el uso de reglas de división ($\mathcal{TDC}(2)$), constituye una frontera entre la no eficiencia y la presumible eficiencia. Así pues, esta frontera está especificada a través del *tipo de reglas* que se usan en los modelos computacionales (pasar de prohibir reglas de división o de separación a permitirlas). Es interesante observar que, comparando estas dos últimas fronteras, se deduce lo siguiente: desde el punto de vista de la complejidad

computacional, la *replicación de objetos* (base de la semántica de las reglas de división celular) es más potente que la *distribución de objetos* (base de la semántica de las reglas de separación celular). Más aún, se podría afirmar que la replicación es *estrictamente más potente* que la distribución si y sólo si se verifica que $\mathbf{P} \neq \mathbf{NP}$.

5.2.2. Sistemas P de tejidos symport/antiport y sin entorno

Aquí, se analiza el papel que juega el entorno en los sistemas P de tejidos reconocedores, desde el punto de vista de la complejidad computacional.

En [49] se ha probado que toda familia de sistemas P de tejidos reconocedores con división celular de $\mathcal{TDC}(k)$ ($k \geq 1$) que resuelve un problema de decisión X de manera uniforme y en tiempo polinomial, puede ser simulada por una familia de sistemas P de tejidos reconocedores con división celular y sin entorno de $\widehat{\mathcal{TDC}}(k)$ que resuelve, así mismo, de manera uniforme y en tiempo polinomial, el mismo problema X. Es decir, se tiene que $\mathbf{PMC}_{\mathcal{TDC}(k)} = \mathbf{PMC}_{\widehat{\mathcal{TDC}}(k)}$, para cada $k \geq 1$. Por tanto, en el marco de los sistemas P de tejidos reconocedores y con división celular, el **entorno** juega un **papel irrelevante** desde el punto de vista de la complejidad computacional.

La cuestión que nos planteamos seguidamente es qué sucede si consideramos reglas de separación celular en lugar de reglas de división celular. En [27] se ha probado (usando la técnica algorítmica) que únicamente se pueden resolver problemas de la clase **P** mediante familias de sistemas P de tejidos reconocedores sin entorno que usan reglas de separación celular (con independencia de la longitud de las reglas de comunicación que se permitan en esos sistemas de membranas). Es decir, se tiene que $\mathbf{P} = \mathbf{PMC}_{\widehat{\mathcal{TSC}}}$. Por tanto, en el marco de los sistemas P de tejidos reconocedores y con separación celular, el **entorno** juega un **papel relevante** desde el punto de vista de la complejidad computacional.

Por tanto, nuevas fronteras de la tratabilidad de problemas son las siguientes:

- En sistemas P de tejidos reconocedores sin entorno: pasar del uso de reglas de separación celular ($\widehat{\mathcal{TSC}}$) a permitir el uso de reglas de división celular ($\widehat{\mathcal{TDC}}$).

- En sistemas P de tejidos reconocedores sin entorno que usan reglas de comunicación de longitud a lo sumo k (para cada $k \geq 2$): pasar del uso

de reglas de separación celular ($\widehat{\mathcal{TSC}}(k)$) a permitir el uso de reglas de división celular ($\widehat{\mathcal{TDC}}(k)$).

- En sistemas P de tejidos reconocedores sin entorno que usan reglas de división celular: pasar de usar sólo reglas de comunicación de longitud 1 ($\widehat{\mathcal{TDC}}(1)$) a permitir el uso de reglas de comunicación de longitud a lo sumo 2 ($\widehat{\mathcal{TDC}}(2)$).
- En sistemas P de tejidos reconocedores que usan reglas de separación celular de longitud a lo sumo 3: pasar de no poseer entorno ($\widehat{\mathcal{TSC}}(3)$) a tenerlo ($\mathcal{TSC}(3)$).

En las Figuras 5.4 y 5.5 aparecen resumidas todas las fronteras de la tratabilidad de problemas que han sido obtenidas en sistemas de membranas que trabajan a modo de tejidos.

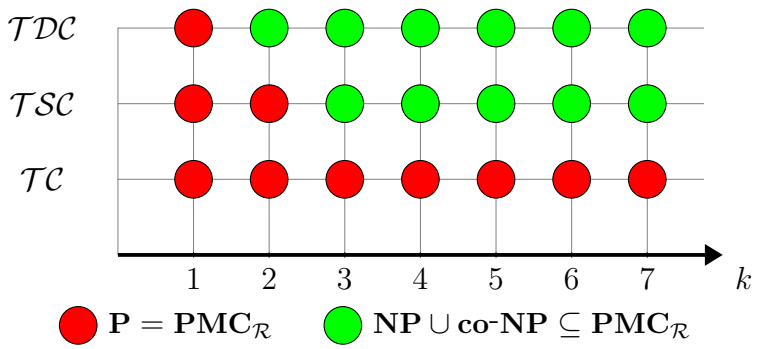


Figura 5.4: Fronteras de la eficiencia en sistemas P de tejidos con entorno

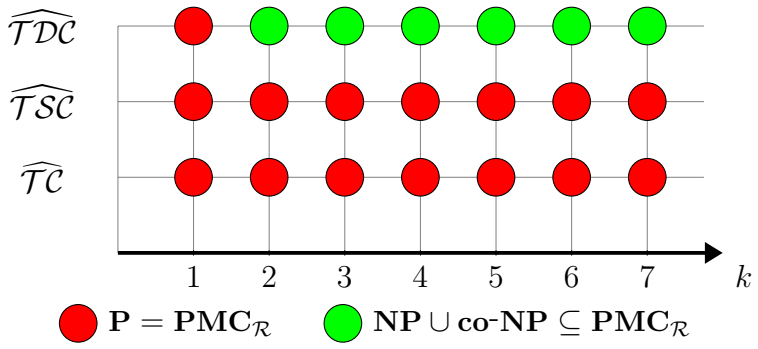


Figura 5.5: Fronteras de la eficiencia en sistemas P de tejidos sin entorno

5.2.3. Sistemas P de tejidos con reglas de comunicación evolutiva

En primer lugar, conviene tener presente que si $\mathcal{X} \subseteq \{\mathcal{D}, \mathcal{S}\}$ entonces se tiene que $\mathcal{TXC} \subseteq \mathcal{TXEC}$ ya que, por una parte, toda regla symport $(i, u/\lambda, j)$ puede ser considerada como una regla de comunicación evolutiva del tipo $[\,u\,]_i[\quad]_j \rightarrow [\quad]_i[\,u\,]_j$; y, por otra, toda regla antiport $(i, u/v, j)$ puede ser considerada como una regla de comunicación evolutiva del tipo $[\,u\,]_i[\,v\,]_j \rightarrow [\,v\,]_i[\,u\,]_j$.

Además, se verifican las relaciones siguientes:

- $\mathbf{PMC}_{\mathcal{TXC}(k)} \subseteq \mathbf{PMC}_{\mathcal{TXEC}(k,k)} \subseteq \mathbf{PMC}_{\mathcal{TXEC}(2k)}$.
- $\mathbf{PMC}_{\mathcal{TXEC}(k_1,k_2)} \subseteq \mathbf{PMC}_{\mathcal{TXEC}(k_1+k_2)}$.
- Si $k \geq 2$ entonces $\mathbf{PMC}_{\mathcal{TXEC}(k)} \subseteq \mathbf{PMC}_{\mathcal{TXEC}(k,k-1)}$.

En [60] se ha probado que los sistemas P de tejidos reconocedores con reglas de comunicación evolutiva y división celular que, inicialmente, tienen una sóla célula y, o bien usan reglas de evolución symport de longitud, a lo sumo 3, o bien usan reglas de evolución antiport de longitud, a lo sumo 4, son modelos universales, en el sentido de tener la misma potencia computacional que las máquinas de Turing deterministas. Además, en el artículo antes citado se ha probado, por un aparte, la no eficiencia de los sistemas P de tejidos de $\mathcal{TDEC}(2)$ (es decir, $\mathbf{P} = \mathbf{PMC}_{\mathcal{TDEC}(2)}$) y, por otra, la presumible eficiencia de los sistemas P de tejidos de $\mathcal{TDEC}(4)$, estableciendo que $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{TDEC}(4)}$. Por tanto, en el marco de los sistemas de $\mathcal{TDEC}(k)$, pasar de longitud $k = 2$ a longitud $k = 4$ equivale a pasar de la no eficiencia a la presumible eficiencia.

En [39] se han establecido los siguientes resultados:

- La no eficiencia de los sistemas P de tejidos de $\mathcal{TSEC}(n, 1)$ y $\mathcal{TSEC}(1, n)$, para cada $n \geq 1$; es decir, que $\mathbf{P} = \mathbf{PMC}_{\mathcal{TSEC}(n,1)} = \mathbf{PMC}_{\mathcal{TSEC}(1,n)}$.
- La presumible eficiencia de los sistemas P de tejidos de $\mathcal{TSEC}(n, 2)$, para cada $n \geq 3$, estableciendo que $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{TSEC}(3,2)}$.

Por tanto, en el marco de los sistemas de $\mathcal{TDEC}(n, k)$, para cada $n \geq 3$, pasar de longitud $k = 1$ a longitud $k = 2$ equivale a pasar de la no eficiencia a la presumible eficiencia.

En las Figuras 5.6, 5.7, 5.8 y 5.9 aparecen resumidas todas las fronteras de la tratabilidad de problemas que han sido obtenidas en los sistemas P de tejidos con reglas de comunicación evolutiva.

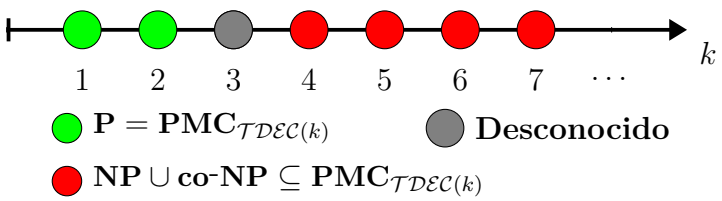


Figura 5.6: Fronteras de la eficiencia en sistemas P de tejidos con reglas de comunicación evolutiva y división celular (métrica [60])

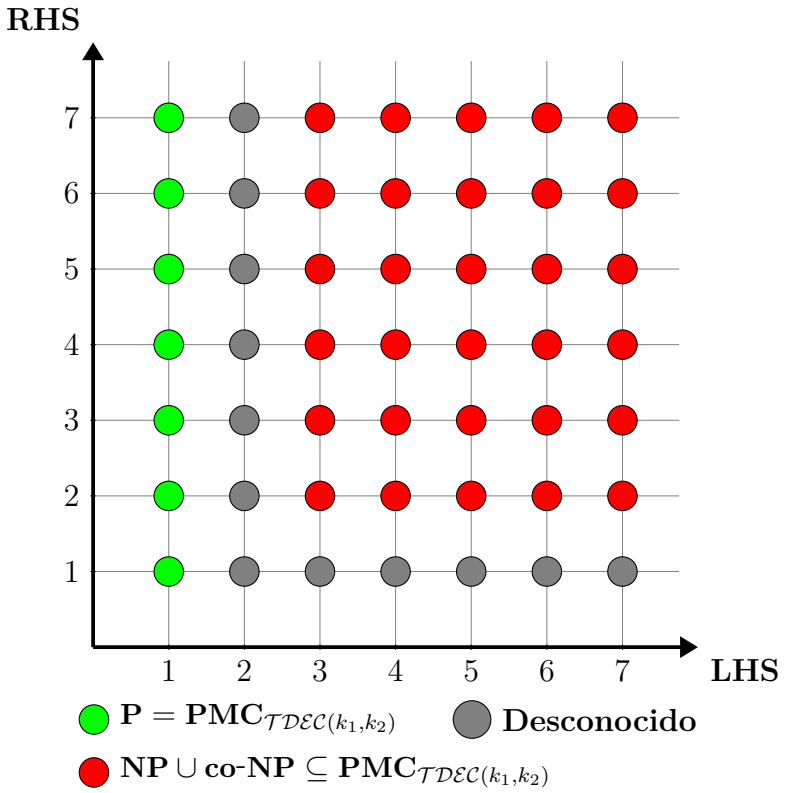


Figura 5.7: Fronteras de la eficiencia en sistemas P de tejidos con reglas de comunicación evolutiva y división celular (métrica [39])

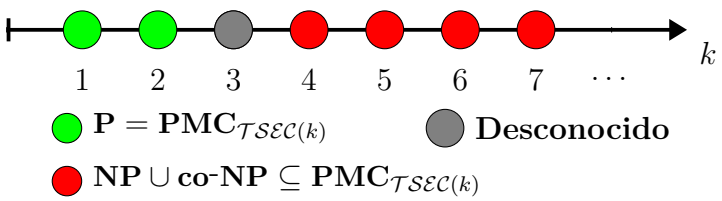


Figura 5.8: Fronteras de la eficiencia en sistemas P de tejidos con reglas de comunicación evolutiva y separación celular (métrica [60])

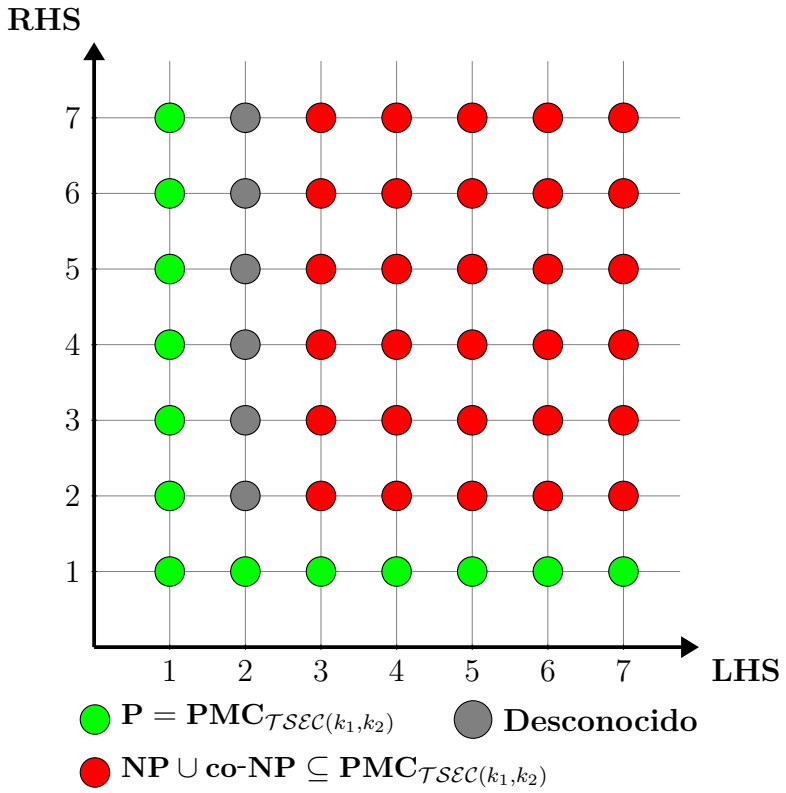


Figura 5.9: Fronteras de la eficiencia en sistemas P de tejidos con reglas de comunicación evolutiva y separación celular (métrica [39])

Parte II

Cooperación mínimal

Capítulo 6

Cooperación minimal en reglas de evolución

Las reacciones químicas con un único reactivo suelen aparecer ocasionalmente en la «Naturaleza viva». Más aún, el reactivo que dispara la reacción suele ser un compuesto que se ha formado por una reacción previa. En el caso de que ese compuesto sea inestable, tenderá a descomponerse de nuevo, bien en sus componentes originales o bien en unos nuevos. Además, teniendo en cuenta que dentro de las células ocurren cerca de mil millones de reacciones químicas por segundo, es lógico pensar que en muchas de ellas concurran dos o más elementos como reactivos. A nivel formal de un paradigma computacional basado en reglas de reescritura, esto se puede traducir en una «cooperación» entre objetos que aperecerán en la correspondiente parte izquierda de la regla.

En el paradigma computacional de *Membrane Computing*, los sistemas P con membranas activas tienen una serie de ingredientes sintácticos que son interesantes de resaltar:

(a) La estructura subyacente es un árbol enraizado etiquetado; es decir, son sistemas que trabajan a modo de células.

(b) Las membranas del sistema tienen asociadas etiquetas y, además, cargas eléctricas entre tres posibles polarizaciones (neutra, positiva y negativa).

(c) Las reglas están asociadas a etiquetas, no a membranas.

(d) Las reglas son no cooperativas; es decir, las partes izquierdas de dichas reglas contienen exactamente un símbolo de un alfabeto.

(e) La aplicación de una regla del sistema puede alterar la carga eléctrica de la membrana a la que se aplica pero no puede cambiar la etiqueta asociada a la misma.

(f) A excepción de uno de los tipos de reglas (de evolución de objetos), las restantes constan de un único símbolo en la correspondiente parte derecha; es decir, son reglas de *producción minimal*.

(g) No existen prioridades entre las reglas.

Además, la semántica tiene unas características peculiares, en donde el paralelismo maximal se usa de una manera muy restringida.

Es bien conocido que se pueden resolver eficientemente problemas computacionalmente duros a través de familias de sistemas P con membranas activas, aunque parece que algunos ingredientes utilizados pudieran ser supérfluos (por ejemplo, las reglas de disolución, las tres polarizaciones, etc.). Por ello, se introdujeron otros sistemas de membranas deducidos de los anteriores eliminando las cargas eléctricas asociadas a las membranas.

En el marco de los sistemas P con membranas activas y sin polarizaciones se demostró que las reglas de disolución jugaban un papel (aparentemente) crucial a la hora de proporcionar soluciones eficientes de problemas computacionalmente duros. En particular, se demostró la no eficiencia de ese tipo de sistemas, incluído el caso en que se permitiera usar reglas de división de membranas elementales y no elementales. Si se analiza con detalle la semántica asociada a las reglas de disolución, se puede observar que su aplicación afecta a objetos que no aparecen en la expresión sintáctica de la regla.

El objetivo de este capítulo consiste en estudiar la incorporación de la cooperación («en su mínima expresión») en las reglas de dichos sistemas como elemento «sustitutivo» de las reglas de disolución, a fin de conseguir la presumible eficiencia de dichos sistemas y, en consecuencia, proporcionar nuevas fronteras de la tratabilidad de problemas abstractos (bajo el supuesto de que **P** fuese distinto de **NP**).

6.1. $\mathcal{DAM}^0(-d)$: Cooperación en reglas de evolución de objetos

Recordemos que $\mathcal{DAM}^0(-d)$ es la clase de los sistemas P reconocedores con membranas activas, sin polarización, sin reglas de disolución y con reglas de división de membranas. En esta sección se van a considerar sistemas de

membranas de $\mathcal{DAM}^0(-d)$ en los que se permitan la cooperación en las reglas de evolución de objetos.

Definición 6.1. *Un sistema P con membranas activas de grado $q \geq 1$, sin polarización, con reglas de división y con cooperación minimal en las reglas de evolución de objetos, es una tupla $\Pi = (\Gamma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$ tal que Π es un sistema P con membranas activas, sin polarizaciones, con reglas de división tal que las las reglas de evolución de objetos son del siguiente tipo:*

(a') $[\, u \to v \,]_h$, *para* $h \in H = \{1, \ldots, q\}, u, v \in M(\Gamma), 1 \leq |u| \leq 2$

Entre los sistemas P antes introducidos vamos a especificar distintos tipos de cooperación minimal en las reglas de evolución de objetos.

- Cooperación minimal primaria ***(pmc)**: las reglas de evolución son del tipo $[\, u \to v \,]_h$, para $h \in H, u, v \in M(\Gamma), 1 \leq |u| \leq 2, 1 \leq |v| \leq 2$.*
- Cooperación minimal acotada ***(bmc)**: las reglas de evolución son del tipo $[\, u \to v \,]_h$, para $h \in H, u, v \in M(\Gamma), 1 \leq |v| \leq |u| \leq 2$.*
- Cooperación minimal y producción minimal ***(mcmp)**: las reglas de evolución son del tipo $[\, u \to a \,]_h$, para $h \in H, u \in M(\Gamma), a \in \Gamma, 1 \leq |u| \leq 2$.*

Denotaremos entonces por $\mathcal{DAM}^0(\alpha, \beta, \delta, \gamma)$ la clase de sistemas P reconocedores con membranas activas y con reglas de división, donde:

- $\alpha = pmc$ (respectivamente, bmc, $mcmp$) cuando se permita el uso de *cooperación minimal primaria* (respectivamente, *cooperación minimal acotada*, *cooperación minimal y producción minimal*).
- $\beta = \pm c$ cuando se permita (respectivamente, se prohíba) el uso de reglas de comunicación.
- $\delta = \pm d$ cuando se permita (respectivamente, se prohíba) el uso de reglas de disolución.
- $\gamma = \pm n$ cuando se permita el uso de reglas de división sólo para membranas elementales (respectivamente, para membranas elementales y no elementales).

A lo largo del desarrollo del Trabajo de Fin de Máster [33] previo a esta Tesis Doctoral, se proporciona una solución eficiente y uniforme del problema SAT usando reglas de evolución de objetos con *cooperación minimal acotada*, incluso prohibiendo las reglas de disolución y permitiendo división sólo en membranas elementales. Es decir, se tiene que $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{DAM}^0(bmc,+c,-d,-n)}$ [63].

Dado que la *cooperación minimal y producción minimal* es más restrictiva que la cooperación minimal acotada, parece interesante explorar si la primera proporciona una menor eficiencia computacional. A continuación vamos a responder negativamente a esta cuestión detallando una solución eficiente y uniforme del problema SAT en ese nuevo marco.

6.1.1. Una solución eficiente al problema SAT mediante sistemas de $\mathcal{DAM}^0(mcmp, +c, -d, -n)$

En esta sección se va a establecer que el problema SAT puede ser resuelto en tiempo polinomial y de manera uniforme por una familia de sistemas P reconocedores con membranas activas, sin polarización, sin reglas de disolución y con reglas de división sólo para membranas elementales pero que permiten cooperación minimal y producción minimal en las reglas de evolución de objetos.

A continuación, se describe una familia de sistemas de membranas de $\mathcal{DAM}^0(mcmp, +c, -d, -n)$ que resuelve el problema SAT en tiempo polinomial y de manera uniforme.

Para cada $n, p \in \mathbb{N}$, consideramos el sistema P reconocedor $\Pi(\langle n, p\rangle) = (\Gamma, \Sigma, H, \mu, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, i_{in}, i_{out})$ de $\mathcal{DAM}^0(mcmp, +c, -d, -n)$, definido como sigue:

1. Alfabeto de trabajo Γ:
   $\{\texttt{yes}, \texttt{no}, \alpha, \beta', \beta'', \gamma, \gamma', \gamma'', \#\} \cup \{a_{i,k} \mid 1 \leq i \leq n, 1 \leq k \leq i\} \cup$
   $\{t_{i,k}, f_{i,k} \mid 1 \leq i \leq n, i \leq k \leq n+p-1\} \cup$
   $\{\beta_k \mid 0 \leq k \leq n+2p+1\} \cup \{c_j \mid 1 \leq j \leq p\} \cup \{d_j \mid 2 \leq j \leq p\} \cup$
   $\{T_{i,k}, F_{i,k} \mid 1 \leq i \leq n, 0 \leq k \leq n-1\} \cup \{T_i, F_i \mid 1 \leq i \leq n\} \cup$
   $\{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 0 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n+p\}$

2. Alfabeto de entrada Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,j,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.

3. $H = \{1, 2\}$.

4. $\mu = [\,[\quad]_2\,]_1$; es decir, $\mu = (V, E)$ donde $V = \{1, 2\}$ y $E = \{(1, 2)\}$.

5. $\mathcal{M}_1 = \{\alpha, \beta_0\}$, $\mathcal{M}_2 = \{a_{i,1}, T^p_{i,0}, F^p_{i,0} \mid 1 \leq i \leq n\}$.

6. El conjunto $\mathcal{R}$ tiene las siguientes reglas:

   1.1 Reglas para un contador general.
   $[\,\beta_k \to \beta_{k+1}\,]_1$, para $0 \leq k \leq n+2p$
   $[\,\beta_{n+2p+1k} \to \beta'\,]_1$

   1.2 Reglas para devolver una respuesta afirmativa.
   $[\,\alpha\gamma \to \gamma'\,]_1$
   $[\,\gamma' \to \gamma''\,]_1$
   $[\,\gamma''\,]_1 \to \texttt{yes}\,[\quad]_1$

   1.3 Reglas para devolver una respuesta negativa.
   $[\,\alpha\beta' \to \beta''\,]_1$
   $[\,\beta''\,]_1 \to \texttt{no}\,[\quad]_1$

   2.1 Reglas para generar las 2^n valoraciones de verdad.
   $[\,a_{i,i}\,]_2 \to [\,t_{i,i}\,]_2\,[\,f_{i,i}\,]_2$, para $1 \leq i \leq n$
   $[\,a_{i,k} \to a_{i,k+1}\,]_2$, para $2 \leq i \leq n \wedge 1 \leq k \leq i-1$

   2.2 Reglas para producir exactamente p copias de cada valoración de verdad.
   $$\left.\begin{array}{l}[\,t_{i,k} \to t_{i,k+1}\,]_2\\ [\,f_{i,k} \to f_{i,k+1}\,]_2\end{array}\right\} 1 \leq i \leq n-1 \wedge i \leq k \leq n-1$$
   $$\left.\begin{array}{l}[\,T_{i,k} \to T_{i,k+1}\,]_2\\ [\,F_{i,k} \to F_{i,k+1}\,]_2\end{array}\right\} 1 \leq i \leq n, 0 \leq k \leq n-2$$
   $$\left.\begin{array}{l}[\,T_{i,n-1} \to T_i\,]_2\\ [\,F_{i,n-1} \to F_i\,]_2\end{array}\right\} 1 \leq i \leq n$$
   $$\left.\begin{array}{l}[\,t_{i,k}F_i \to t_{i,k+1}\,]_2\\ [\,f_{i,k}T_i \to f_{i,k+1}\,]_2\end{array}\right\} 1 \leq i \leq n \wedge n \leq k \leq n+p-2$$
   $$\left.\begin{array}{l}[\,t_{i,n+p-1}F_i \to \#\,]_2\\ [\,f_{i,n+p-1}T_i \to \#\,]_2\end{array}\right\} 1 \leq i \leq n$$

   2.3 Reglas para preparar la fórmula para chequear las cláusulas.
   $$\left.\begin{array}{l}[\,x_{i,j,k} \to x_{i,j,k+1}\,]_2\\ [\,\overline{x}_{i,j,k} \to \overline{x}_{i,j,k+1}\,]_2\\ [\,x^*_{i,j,k} \to x^*_{i,j,k+1}\,]_2\end{array}\right\} 1 \leq i \leq n,\ 1 \leq j \leq p,\ 0 \leq k \leq n+p-1$$

   2.4 Reglas para chequear qué cláusulas son satisfechas.

$$\left.\begin{array}{l}[\, T_i x_{i,j,n+p} \to c_j \,]_2 \\ [\, T_i \overline{x}_{i,j,n+p} \to \# \,]_2 \\ [\, T_i x^*_{i,j,n+p} \to \# \,]_2 \\ [\, F_i x_{i,j,n+p} \to \# \,]_2 \\ [\, F_i \overline{x}_{i,j,n+p} \to c_j \,]_2 \\ [\, F_i x^*_{i,j,n+p} \to \# \,]_2 \end{array}\right\} 1 \leq i \leq n \wedge 1 \leq j \leq p$$

2.5 Reglas para chequear si todas las cláusulas son satisfechas por una valoración de verdad.

$[\, c_1 c_2 \to d_2 \,]_2$

$[\, d_j c_{j+1} \to d_{j+1} \,]_2$, para $2 \leq j \leq p-1$

2.6 Regla para preparar una respuesta afirmativa.

$[\, d_p \,]_2 \to \gamma\, [\quad]_2$

7. La membrana de entrada es la membrana etiquetada por 2 ($i_{in} = 2$) y la región de salida es el entorno ($i_{out} = env$).

Es interesante hacer notar que una fórmula proposicional $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_p, C_j = l_{j_1} \vee \cdots \vee l_{j_{r_j}}, 1 \leq j \leq p, l_{j_k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$, instancia del problema SAT será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$, en donde $s(\varphi) = \langle n, p \rangle$ y $cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$

La solución propuesta captura, en el marco $\mathcal{DAM}^0(mcmp, +c, -d, -n)$, un algoritmo de fuerza bruta que resuelve el problema SAT. Específicamente, la solución se estructura en las siguientes fases:

- *Fase de generación*: Usando las reglas de división de 2.1, se generan todas las posibles valoraciones de verdad para las variables $\{x_1, \ldots, x_n\}$ asociadas a φ. Específicamente, se generan 2^n membranas etiquetadas por 2, cada una conteniendo una valoración de verdad diferente. Esta fase tarda n pasos de computación, siendo n el número de variables de φ.

- *Fase de producción de copias suficientes para cada valoración de verdad*: En esta fase se crean p copias de cada valoración de verdad en las membranas correspondientes para permitir el chequeo de cada literal con el valor de verdad en cada cláusula en cada membrana. Esta fase dura exactamente p pasos de computación.

- *Primera fase de chequeo*: Gracias a la aplicación de las reglas de 2.4, las cuales se aplican en paralelo, se generan objetos c_j en las membranas

etiquetadas por 2 cuya valoración de verdad hace verdadera la cláusula C_j. Esta fase dura exactamente 1 paso de computación.

- *Segunda fase de chequeo*: En esta fase, por medio de las reglas de 2.5, se chequea si todas las cláusulas $C_1, \ldots, C_p$ de la fórmula φ son satisfechas por las valoraciones de verdad codificadas en las membranas etiquetadas por 2. Esta fase tarda $p-1$ pasos de computación, siendo p el número de cláusulas de φ.

- *Fase de salida*: El sistema usa reglas de 1.2 y 1.3 para devolver una respuesta positiva o negativa. Esta fase dura 4 pasos de computación, independientemente de si la respuesta es positiva o negativa.

6.1.1.1. Verificación formal

A continuación, vamos a verificar formalmente la solución propuesta para el problema `SAT`.

Fase de generación

En esta fase, se generan todas las valoraciones de verdad asociadas a la fórmula Booleana φ, aplicando las reglas de división de 2.1 en las membranas etiquetadas por 2. De tal manera que en el paso $i, 1 \leq i \leq n$ de esta fase, una regla de división es disparada por el objeto $a_{i,i}$, produciendo dos nuevas membranas con el resto del contenido duplicado en las dos nuevas membranas creadas. Esta fase termina cuando todas las valoraciones con respecto a objetos $t_{i,n}, f_{i,n}, 1 \leq i \leq n$ han sido generadas.

Proposición 6.1. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(*a*) *Para cada* $k, 1 \leq k \leq n-1$, *en la configuración* $\mathcal{C}_k$ *tenemos* $\mathcal{C}_k(1) = \{\alpha, \beta_k\}$ *(siendo* $\mathcal{C}_i(t)$ *el contenido de la membrana* i *en el instante* t*) y hay* 2^k *membranas etiquetadas por 2 tales que contienen: el conjunto* $\{a_{i,k+1} \mid k+1 \leq i \leq n\}$, *el conjunto* $cod_k(\varphi)$, *el multiconjunto* $\{T^p_{i,k}, F^p_{i,k} \mid 1 \leq i \leq n\}$ *y un subconjunto diferente* $\{r_{1,k}, \ldots, r_{k,k}\}$, *siendo* $r \in \{t, f\}$.

(*b*) *En la configuración* $\mathcal{C}_n$ *tenemos* $\mathcal{C}_n(1) = \{\alpha, \beta_n\}$ *y hay* 2^n *membranas etiquetadas por 2 tales que contienen: el conjunto* $cod_n(\varphi)$, *el multiconjunto* $\{T^p_i, F^p_i \mid 1 \leq i \leq n\}$ *y un subconjunto diferente* $\{r_{1,n}, \ldots, r_{n,n}\}$, *siendo* $r \in \{t, f\}$.

Demostración. Esta afirmación será demostrada por inducción.

(*a*) Por inducción sobre k. El caso base $k = 1$ se cumple teniendo en cuenta que la configuración $\mathcal{C}_1$ es obtenida desde la configuración $\mathcal{C}_0$ por la aplicación de las reglas $[\,\beta_0 \to \beta_1\,]_1, [\,a_{1,1}\,]_2 \to [\,t_{1,1}\,]_2[\,f_{1,1}\,]_2$, $[\,T_{i,0} \to T_{i,1}\,]_2$, $[\,F_{i,0} \to F_{i,1}\,]_2$, para $1 \leq i \leq n$, $[\,x_{i,j,0} \to x_{i,j,1}\,]_2$, $[\,\overline{x}_{i,j,0} \to \overline{x}_{i,j,1}\,]_2$ y $[\,x^*_{i,j,0} \to x^*_{i,j,1}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

Supongamos que el resultado es válido para $k, 1 \leq k < n-1$. Veamos entonces si el resultado es válido para $k+1$.

Por una parte, en la configuración $\mathcal{C}_k$ tenemos $\mathcal{C}_k(1) = \{\alpha, \beta_k\}$ y hay 2^k membranas etiquetadas por 2 tales que contienen el conjunto $\{a_{i,k+1} \mid k+1 \leq i \leq n\}$, el conjunto $cod_k(\varphi)$, el multiconjunto $\{T^p_{i,k}, F^p_{i,k} \mid 1 \leq i \leq n\}$ y un subconjunto diferente $\{r_{1,k}, \ldots, r_{k,k}\}$, siendo $r \in \{t, f\}$.

Por otra parte, la configuración $\mathcal{C}_{k+1}$ se obtiene de la configuración $\mathcal{C}_k$ por la aplicación de las reglas $[\,\beta_k \to \beta_{k+1}\,]_1$, $[\,a_{k+1,k+1}\,]_2 \to [\,t_{k+1,k+1}\,]_2[\,f_{k+1,k+1}\,]_2$, $[\,t_{i,k} \to t_{i,k+1}\,]_2$, $[\,f_{i,k} \to f_{i,k+1}\,]_2$, para $1 \leq i \leq k$, $[\,a_{i,k+1} \to a_{i,k+2}\,]_2$, para $k+2 \leq i \leq n$, $[\,T_{i,k} \to T_{i,k+1}\,]_2$, $[\,F_{i,k} \to F_{i,k+1}\,]_2$, para $1 \leq i \leq n$, $[\,x_{i,j,k} \to x_{i,j,k+1}\,]_2$, $[\,\overline{x}_{i,j,k} \to \overline{x}_{i,j,k+1}\,]_2$ y $[\,x^*_{i,j,k} \to x^*_{i,j,k+1}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

Por lo tanto, el resultado es válido para $k+1$.

(*b*) Aplicando (a) a $k = n-1$ en la configuración $\mathcal{C}_{n-1}$ tenemos que $\mathcal{C}_{n-1}(1) = \{\alpha, \beta_{n-1}\}$ y que hay 2^{n-1} membranas etiquetadas por 2 tales que contienen el objeto $a_{n,n}$, el conjunto $cod_{n-1}(\varphi)$, el multiconjunto $\{T^p_{i,k}, F^p_{i,k} \mid 1 \leq i \leq n\}$ y un subconjunto diferente $\{r_{1,n-1}, \ldots, r_{n-1,n-1}\}$, siendo $r \in \{t, f\}$.

Entonces, (b) se sucede dado que la configuración $\mathcal{C}_n$ se obtiene de la configuración $\mathcal{C}_{n-1}$ por la aplicación de las reglas $[\,\beta_{n-1} \to \beta_n\,]_1$, $[\,a_{n,n}\,]_2 \to [\,t_{n,n}\,]_2[\,f_{n,n}\,]_2$, $[\,t_{i,n-1} \to t_{i,n}\,]_2$, $[\,f_{i,n-1} \to f_{i,n}\,]_2$, for $1 \leq i \leq n-1$, $[\,x_{i,j,n-1} \to x_{i,j,n}\,]_2$, $[\,\overline{x}_{i,j,n-1} \to \overline{x}_{i,j,n}\,]_2$ y $[\,x^*_{i,j,n-1} \to x^*_{i,j,n}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

□

Fase de producción de copias suficientes para cada valoración de verdad

En cada membrana etiquetada por 2 se van a generar exactamente p copias de la valoración de verdad codificada por dicha membrana, siendo p el número

de cláusulas de la fórmula φ. Téngase presente que en la configuración inicial hay p copias de todos los objetos $T_i, F_i, 1 \leq i \leq n$. Estas copias se han ido replicando en las 2^n membranas gracias a las reglas de división. Usando reglas de evolución con cooperación entre estos objetos y los valores de verdad t_i y f_i asociados a cada membrana etiquetada por 2 hará que estos los objetos F_i o T_i, respectivamente, sean eliminados de las correspondientes membranas. Esta fase tarda exactamente p pasos de computación.

En la siguiente proposición, se demuestra la materialización de este proceso.

Proposición 6.2. *Sea $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ una computación del sistema $\Pi(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$.*

(*a*) *Para cada $k, 1 \leq k \leq p-1$, en la configuración $\mathcal{C}_{n+k}$ tendremos $\mathcal{C}_{n+k}(1) = \{\alpha, \beta_{n+k}\}$ y habrán 2^n membranas etiquetadas por 2 tales que cada una de ella contiene el conjunto $cod_{n+k}(\varphi)$, un subconjunto diferente $\{r_{1,n+k}, \ldots, r_{n,n+k}\}$, siendo $r \in \{t, f\}$ y el multiconjunto correspondiente $\{R_1^p, \overline{R}_1^{p-k}, \ldots, R_n^p, \overline{R}_n^{p-k}\}$ que verifica lo siguiente: para cada $k, 1 \leq i \leq n$, si $r_{i,n+k} = t_{i,n+k}$ entonces $R_i = T_i$ y $\overline{R}_i = F_i$; si $r_{i,n+k} = f_{i,n+k}$ entonces $R_i = F_i$ y $\overline{R}_i = T_i$.*

(*b*) *En la configuración $\mathcal{C}_{n+p}$ tenemos $\mathcal{C}_{n+p}(1) = \{\alpha, \beta_{n+p}\}$ y hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene el conjunto $cod_{n+p}(\varphi)$, n copias del objeto $\#$ y un subconjunto diferente $\{R_1^p, \ldots, R_n^p\}$ siendo $R \in \{T, F\}$.*

Demostración. Se demostrará, de nuevo, por inducción.

(*a*) Por inducción sobre k. El caso base $k = 1$ se cumple teniendo en cuenta que la configuración $\mathcal{C}_{n+1}$ es obtenida de la configuración $\mathcal{C}_n$ por la aplicación de las reglas $[\, \beta_n \to \beta_{n+1} \,]_1$, $[\, t_{i,n} F_i \to t_{i,n+1} \,]_2$, $[\, f_{i,n} T_i \to f_{i,n+1} \,]_2$, para $1 \leq i \leq n$, $[\, x_{i,j,n} \to x_{i,j,n+1} \,]_2$, $[\, \overline{x}_{i,j,n} \to \overline{x}_{i,j,n+1} \,]_2$ y $[\, x^*_{i,j,n} \to x^*_{i,j,n+1} \,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

Supongamos que el resultado es válido para $k, 1 \leq k < p-1$. Veamos si el resultado es válido para $k+1$.

Por una parte, en la configuración $\mathcal{C}_{n+k}$ tenemos $\mathcal{C}_{n+k}(1) = \{\alpha, \beta_{n+k}\}$ y hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene el conjunto $cod_{n+k}(\varphi)$, un subconjunto diferente $\{r_{1,n+k}, \ldots, r_{n,n+k}\}$, siendo $r \in \{t, f\}$ y el multiconjunto correspondiente $\{R_1^p, \overline{R}_1^{p-k}, \ldots, R_n^p, \overline{R}_n^{p-k}\}$ que verifica lo siguiente: para cada $k, 1 \leq i \leq n$, si $r_{i,n+k} = t_{i,n+k}$ entonces $R_i = T_i$ y $\overline{R}_i = F_i$; si $r_{i,n+k} = f_{i,n+k}$ entonces $R_i = F_i$ y $\overline{R}_i = T_i$;

Por otra parte, la configuración $\mathcal{C}_{n+k+1}$ se obtiene de la configuración $\mathcal{C}_{n+k}$ por la aplicación de las reglas $[\,\beta_{n+k} \to \beta_{n+k+1}\,]_1$, $[\,t_{i,n+k}F_i \to t_{i,n+k+1}\,]_2$, $[\,f_{i,n+k}T_i \to f_{i,n+k+1}\,]_2$, para $1 \leq i \leq n$, $[\,x_{i,j,n+k} \to x_{i,j,n+k+1}\,]_2$, $[\,\overline{x}_{i,j,n+k} \to \overline{x}_{i,j,n+k+1}\,]_2$ y $[\,x^*_{i,j,n+k} \to x^*_{i,j,n+k+1}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

Por lo tanto, el resultado es válido para $k+1$.

(*b*) Aplicando (a) a $k = p-1$, en la configuración $\mathcal{C}_{n+p-1}$ tenemos $\mathcal{C}_{n+p-1}(1) = \{\alpha, \beta_{n+p-1}\}$ y hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene el conjunto $cod_{n+p-1}(\varphi)$, un subconjunto diferente $\{r_{1,n+p-1}, \ldots, r_{n,n+p-1}\}$, siendo $r \in \{t, f\}$ y el multiconjunto correspondiente $\{R^p_1, \overline{R}^1_1, \ldots, R^p_n, \overline{R}^1_n\}$ que verifica lo siguiente: para cada $i, 1 \leq i \leq n$, si $r_{i,n+p-1} = t_{i,n+p-1}$ entonces $R_i = T_i, \overline{R}_i = F_i$ y si $r_{i,n+p-1} = f_{i,n+p-1}$ entonces $R_i = F_i, \overline{R}_i = T_i$.

Entonces, (b) se sucede dado que la configuración $\mathcal{C}_{n+p}$ se obtiene de la configuración $\mathcal{C}_{n+p-1}$ por la aplicación de las reglas $[\,\beta_{n+p-1} \to \beta_{n+p}\,]_1$, $[\,t_{i,n+p-1}F_i \to \#\,]_2$, $[\,f_{i,n+p-1}T_i \to \#\,]_2$, para $1 \leq i \leq n$, $[\,x_{i,j,n+p-1} \to x_{i,j,n+p}\,]_2$, $[\,\overline{x}_{i,j,n+p-1} \to \overline{x}_{i,j,n+p}\,]_2$ y $[\,x^*_{i,j,n+p-1} \to x^*_{i,j,n+p}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

□

Primera fase de chequeo

En esta fase se trata de determinar qué cláusulas son satisfechas por cada valoración de verdad codificada en cada una de las membranas etiquetadas por 2. Para ello, las reglas de 2.4 son aplicadas de tal manera que se produce un objeto c_j en una membrana etiquetada por 2 si y sólo si la valoración de verdad codificada en dicha membrana hace verdadera la cláusula C_j. Esta fase tarda exactamente 1 paso de computación.

Proposición 6.3. *Sea $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ una computación del sistema $\Pi(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$. En la configuración $\mathcal{C}_{n+p+1}$ tenemos $\mathcal{C}_{n+p+1}(1) = \{\alpha, \beta_{n+p+1}\}$ y hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene r_j copias de objetos c_j, para $1 \leq j \leq p$ y $0 \leq t_j \leq n$, si y sólo si la valoración de verdad asociada a dicha membrana tiene exactamente r_j literales que hacen verdadera la cláusula C_j, y $np - (r_1 + \cdots + r_p)$ copies of object #.*

Demostración. Es suficiente darse cuenta que la configuración $\mathcal{C}_{n+p+1}$ se obtiene de la configuración $\mathcal{C}_{n+p}$ por la aplicación de las reglas $[\,\beta_{n+p} \to \beta_{n+p+1}\,]_1$, $[\,T_i x_{i,j,n+p} \to c_j\,]_2$, $[\,T_i \overline{x}_{i,j,n+p} \to \#\,]_2$, $[\,T_i x^*_{i,j,n+p} \to \#\,]_2$, $[\,F_i x_{i,j,n+p} \to \#\,]_2$, $[\,F_i \overline{x}_{i,j,n+p} \to c_j\,]_2$ y $[\,F_i x^*_{i,j,n+p} \to \#\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

□

Segunda fase de chequeo

En esta fase, se trata de determinar aquellas membranas etiquetadas por 2 que codifican una valoración de verdad que satisface todas las cláusulas de la fórmula φ. Para ello, se aplican las reglas de 2.5 de tal manera que aparecerá un objeto d_j $(2 \leq j \leq p)$ en una membrana etiquetada por 2 si y sólo si la valoración de verdad codificada en dicha membrana hace verdaderas las cláusulas $C_1, \ldots, C_j$. Por lo tanto, la fórmula será satisfecha por una valoración de verdad asociada a una membrana si y sólo si aparece un objeto d_p en dicha membrana. Esta fase consume exactamente $p - 1$ pasos de computación.

Proposición 6.4. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(*a*) *Para cada* $k, 1 \leq k \leq p-1$, *en la configuración* $\mathcal{C}_{(n+p+1)+k}$ *tenemos* $\mathcal{C}_{(n+p+1)+k}(1) = \{\alpha, \beta_{(n+p+1)+k}\}$ *y hay* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene un objeto* d_{k+1} *si y sólo si la valoración de verdad asociada a dicha membrana hace verdaderas las cláusulas* $C_1, \ldots, C_{k+1}$.

(*b*) *La fórmula* φ *is satisfactible si y sólo si en la configuración* $\mathcal{C}_{n+2p}$ *hay alguna membrane etiquetada por 2 tal que tiene un objeto* d_p.

Demostración. De nuevo, para demostrar que en p pasos se consigue esto, se utilizará inducción.

(*a*) Por inducción sobre k. Para el caso base $k = 1$ es suficiente darse cuenta que la configuración $\mathcal{C}_{n+p+2}$ se obtiene de la configuración $\mathcal{C}_{n+p+1}$ por la aplicación de las reglas $[\,\beta_{n+p+1} \to \beta_{n+p+2}\,]_1$ y $[\,c_1 c_2 \to d_2\,]_2$.

Supongamos que el resultado es válido para $k, 1 \leq k < p-1$. Entonces, en la configuración $\mathcal{C}_{(n+p+1)+k}$ tenemos $\mathcal{C}_{(n+p+1)+k}(1) = \{\alpha, \beta_{(n+p+1)+k}\}$ y hay 2^n membranas etiquetadas por 2 tales que contienen un objeto d_{k+1} si y sólo si la valoración de verdad codificada en dicha membrana hace verdaderas las cláusulas $C_1, \ldots, C_{k+1}$.

Teniendo en cuenta que la configuración $\mathcal{C}_{(n+p+1)+k+1}$ se obtiene de la configuración $\mathcal{C}_{(n+p+1)+k}$ por la aplicación de las reglas $[\,\beta_{(n+p+1)+k} \to \beta_{(n+p+1)+k+1}\,]_1$ y $[\,d_{k+1}c_{k+2} \to d_{k+2}\,]_2$, deducimos que el resultado es válido para $k+1$.

(b) Para probar (b), hay que señalar que la fórmula φ es satisfactible si y sólo si existe una valoración de verdad σ que hace verdadera la fórmula φ; es decir, que hacer verdaderas las cláusulas $C_1, \ldots, C_p$. De (a) deducimos que φ es satisfactible si y sólo si en la configuración $\mathcal{C}_{n+2p}$ existe alguna membrana etiquetada por 2 que contiene al objeto d_p.

□

Fase de salida

La fase de salida comienza en el paso $(n+2p+1)$, y tarda exactamente 4 pasos, independientemente de devolver una respuesta afirmativa o negativa.

- *Respuesta afirmativa*: Si la fórmula φ es satisfactible entonces al menos una de las valoraciones de verdad codificadas en las membranas etiquetadas por 2 hace verdaderas todas las cláusulas. Por lo tanto, una copia del objeto d_p aparecerá en dichas membranas en la configuración $\mathcal{C}_{n+2p}$. Entonces, aplicando las reglas $[\,d_p\,]_2 \to \gamma\,[\quad]_2$ y $[\,\beta_{n+2p} \to \beta_{n+2p+1}\,]_1$, los objetos γ y β_{n+2p+1} son producidos en la membrana piel. En el siguiente paso, por la aplicación de las reglas $[\,\alpha\gamma \to \gamma'\,]_1$ y $[\,\beta_{n+2p+1} \to \beta'\,]_1$, los objetos γ' y β' son producidos en la membrana piel. En el paso $n+2p+3$, aplicando la regla $[\,\gamma' \to \gamma''\,]_1$, el objeto γ'' será producido en la misma membrana (hay que notar que el objeto β' no puede interactuar con el objeto α). Finalmente, en el último paso de computación, aplicando la regla $[\,\gamma''\,]_1 \to \texttt{yes}\,[\quad]_1$, el objeto `yes` es enviado al entorno. Aquí, la computación para y devuelve una respuesta positiva.

- *Respuesta negativa*: Si la fórmula φ no es satisfactible, entonces ninguna de las valoraciones de verdad codificadas en las membranas etiquetadas por 2 hará verdadera la fórmula φ. Por lo tanto, no aparecerá un objeto d_p en ninguna de las membranas etiquetadas por 2 en la configuración $\mathcal{C}_{n+2p}$. En el paso $n+2p+1$, sólo la regla $[\,\beta_{n+2p} \to \beta_{n+2p+1}\,]_1$ será aplicable a la configuración $\mathcal{C}_{n+2p}$. Entonces, $\mathcal{C}_{n+2p+1}(1) = \{\alpha, \beta_{n+2p+1}\}$. En el siguiente paso, aplicando la regla $[\,\beta_{n+2p+1} \to \beta'\,]_1$ tenemos $\mathcal{C}_{n+2p+2}(1) = \{\alpha, \beta'\}$. En el paso de computación $n+2p+3$, la regla $[\,\alpha\beta' \to \beta''\,]_1$ produce un

objeto β'' en la membrana piel. Finalmente, en el último paso de computación, mediante la aplicación de la regla $[\,\beta''\,]_1 \rightarrow \texttt{no}\,[\quad]_1$ un objeto `no` es enviado al entorno. Entonces, la computación para y devuelve una respuesta negativa.

6.1.1.2. Resultado obtenido

Teorema 6.1. $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{DAM}^0(mcmp,+c,-d,-n)}$

Demostración. La familia de sistemas P presentada verifica lo siguiente:

(*a*) Todos los sistemas de la familia $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ pertenece a $\mathcal{DAM}^0(mcmp, +c, -d, -n)$.

(*b*) La familia $\mathbf{\Pi}$ es polinomialmente uniforme por máquinas de Turing dado que para cada $n, p \in \mathbb{N}$, la cantidad de recursos necesarios para construir $\Pi(\langle n, p\rangle)$ es de orden polinomial con respecto a n y p:

- Tamaño del alfabeto: $3np^2 + 3n^2p + 5np + \frac{7n^2}{2} + \frac{5n}{2} + 4p + 10 \in \Theta(max\{np^2, n^2p\})$.
- Número inicial de membranas: $2 \in \Theta(1)$.
- Número inicial de objetos en las membranas: $2np + n + 2 \in \Theta(np)$.
- Número de reglas: $3np^2 + 3n^2p + 8np + \frac{7n^2}{2} + \frac{n}{2} + 3p + 7 \in \Theta(max\{np^2, n^2p\})$.
- Máximo número de objetos envueltos en cada regla: $3 \in \Theta(1)$.

(*c*) El par (cod, s) de funciones computables en tiempo polinomial cumplen lo siguiente: para cada fórmula φ del problema `SAT`, $s(\varphi)$ es un número natural, $cod(\varphi)$ es un multiconjunto del sistema $\Pi(s(\varphi))$, y para cada $t \in \mathbb{N}$, $s^{-1}(t)$ es un conjunto finito.

(*d*) La familia $\mathbf{\Pi}$ es polinomialmente acotada: de hecho, para cada fórmula φ de `SAT`, el sistema P determinista $\Pi(s(\varphi)) + cod(\varphi)$ tarda exactamente $n + 2p + 4$ pasos de computación, siendo n el número de variables y p el número de cláusulas de la fórmula φ.

(*e*) La familia $\mathbf{\Pi}$ es adecuada con respecto a (X, cod, s): de hecho, para cada fórmula φ, si la computación de $\Pi(s(\varphi)) + cod(\varphi)$ es una computación de aceptación, entonces φ es satisfactible.

(f) La familia $\mathbf{\Pi}$ es completa con respecto de (X, cod, s): de hecho, para cada fórmula φ tal que sea satisfactible, la computación de $\Pi(s(\varphi)) + cod(\varphi)$ es una computación de aceptación.

Por lo tanto, la familia $\mathbf{\Pi}$ de sistemas P presentada resuelve el problema SAT en tiempo polinomial y de manera uniforme.

□

Corolario 6.1. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{DAM}^0(mcmp,+c,-d,-n)}$

Demostración. Basta con señalar que SAT es un problema **NP**-completo, $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{DAM}^0(mcmp,+c,-d,-n)}$ y que la clase $\mathbf{PMC}_{\mathcal{DAM}^0(mcmp,+c,-d,-n)}$ es cerrada bajo reducibilidad en tiempo polinomial y bajo complementario.

□

6.1.2. Nuevas fronteras obtenidas

Se tienen los siguientes resultados:

- $\mathbf{P} = \mathbf{PMC}_{\mathcal{DAM}^0(+e,+c,-d,-n)}$
- $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{DAM}^0(mcmp,+c,-d,-n)}$

Es decir, los sistemas de membranas de $\mathcal{DAM}^0(+e, +c, -d, -n)$ son no eficientes, mientras que los sistemas de membranas de $\mathcal{DAM}^0(mcmp, +c, -d, -n)$ son presumiblemente eficientes.

En consecuencia, en el marco de los sistemas P con membranas activas, sin polarización, sin disolución y con reglas de división sólo para membranas elementales, pasar de usar reglas de evolución de objetos no cooperativas a usar reglas de evolución de objetos que hacen uso de *cooperación minimal y producción minimal*, equivale a pasar de la no eficiencia a la presumible eficiencia.

6.2. $\mathcal{SAM}^0(-d)$: Cooperación en reglas de evolución de objetos

Las reglas de separación son un mecanismo usado, al igual que las reglas de división, para generar un espacio exponencial en términos de membranas en tiempo polinomial. La diferencia reside en la capacidad de *duplicar* los objetos que se encontraban en la membrana original de las reglas de división, ya que

en las reglas de separación, el contenido original se *distribuye* entre las dos nuevas membranas creadas.

Definición 6.2. *Un sistema P con membranas activas sin polarización con cooperación minimal en reglas de evolución y separación de grado* $q \geq 1$ *es una tupla* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, H, H_0, H_1, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$*, donde* Π *es un sistema P con membranas activas sin polarización y reglas de separación y reglas de evolución cooperativas, con* $H_0 \cup H_1 = H$ *y* $H_0 \cap H_1 = \emptyset$*, cuyas reglas de separación se definen de la siguiente forma:*

(e) $[\,a\,]_h \rightarrow [\,\Gamma_0\,]_h\,[\,\Gamma_1\,]_h$*, para* $h \in H \setminus \{1, i_{out}\}$*, h es la etiqueta de una hoja de* μ*,* $a \in \Gamma$ *(reglas de separación para membranas elementales).*

(f) $[\,[\,]_{h_1}[\,]_{h_2}\,]_h \rightarrow [\,\Gamma_0\,[\,]_{h_1}\,]_h\,[\,\Gamma_1\,[\,]_{h_2}\,]_h$*, para* $k \geq 1, n > k, h \in H \setminus \{1, i_{out}\}, h_1 \in H_0, h_2 \in H_1$ *(reglas de separación de membranas no elementales).*

Diremos que una *regla de separación de membranas elementales* $[\,a\,]_h \rightarrow [\,\Gamma_0\,]_h\,[\,\Gamma_1\,]_h$ es aplicable a una configuración $\mathcal{C}_t$ del sistema en un instante t si en dicha configuración existe, al menos, una membrana elemental que está etiquetada por h, distinta de la piel y de la membrana de salida si la hubiera y contiene al objeto a. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ produce la creación de dos nuevas membranas con la misma etiqueta h. El objeto a que dispara la regla desaparece y los restantes objetos que aparecían en la membrana donde se ha aplicado la regla, son distribuidos entre las dos membranas creadas. En una de ellas, se colocan todos los elementos correspondientes de Γ_0, y en la otra los correspondientes de Γ_1.

Diremos que una *regla de separación de membranas no elementales* $[\,[\,]_{h_1}[\,]_{h_2}\,]_h \rightarrow [\,\Gamma_0\,[\,]_{h_1}\,]_h\,[\,\Gamma_1\,[\,]_{h_2}\,]_h$ es aplicable a una configuración $\mathcal{C}_t$ del sistema en un instante t si en dicha configuración existe, al menos, una membrana no elemental etiquetada por h, distinta de la piel y de la membrana de salida si la hubiera que contiene membranas etiquetadas por $h_1 \in H_0$ y $h_2 \in H_1$. La ejecución de dicha regla a esa membrana de $\mathcal{C}_t$ etiquetada por h, produce la creación de dos nuevas membranas con la misma etiqueta h de tal manera que: (a) la primera de ellas contiene a los objetos correspondientes de Γ_0 y a las membranas pertenecientes a H_0; y (b) la segunda contiene a los objetos correspondientes de Γ_1 y a las membranas pertenecientes a H_1.

Para denotar las familias de sistemas de membranas estudiados aquí, usamos $\mathcal{SAM}^0(\alpha, \beta, \delta, \gamma)$ con los parámetros $\alpha, \beta, \delta, \gamma$ especificados en la sección

anterior.

En [66], artículo producido desde el Trabajo de Fin de Máster [33] previo a esta Tesis Doctoral, mediante el uso de la técnica algorítmica, se demuestra que dado un sistema P reconocedor de membranas activas que usa *cooperación minimal acotada* en reglas de evolución de objetos, incluso con reglas de disolución y reglas de separación para membranas tanto elementales como no elementales, puede ser simulada una computación del mismo mediante un algoritmo que trabaja en tiempo y espacio polinomiales. Esto da lugar al resultado $\mathbf{P} = \mathbf{PMC}_{\mathcal{SAM}^0(bmc,+c,+d,+n)}$. Esto parece ser debido a la incapacidad de estos sistemas a generar una cantidad de objetos exponencial en tiempo polinomial, por ejemplo, con reglas del tipo $[\, a \to bc\,]_h$. De hecho, como se demostrará a continuación, la incorporación de estas reglas al marco de la *cooperación minimal acotada*, teniendo una *cooperación minimal primaria*, en el marco de los sistemas P con membranas activas con separación sólo de membranas no elementales y sin reglas de disolución, permite a estos sistemas resolver problemas **NP**-completos.

6.2.1. Una solución eficiente al problema SAT mediante sistemas de $\mathcal{SAM}^0(pmc, +c, -d, -n)$

A continuación, se proporciona una solución polinomial uniforme al problema SAT por una familia $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ de sistemas P reconocedores de $\mathcal{SAM}^0(pmc, +c, -d, -n)$. Cada sistema $\Pi(t)$ procesará todas las fórmulas Booleanas φ en forma normal conjuntiva formadas por n variables y p cláusulas, donde $t = \langle n, p\rangle$, dado el multiconjunto de entrada correspondiente $cod(\varphi)$ al sistema (en la membrana correspondiente).

Para cada $n, p \in \mathbb{N}$, consideramos el sistema reconocedor de grado 2 de $\Pi(\langle n, p\rangle) = (\Gamma, \Gamma_0, \Gamma_1, \Sigma, H, \mu, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, i_{in}, i_{out})$ como sigue:

1. Alfabeto de trabajo Γ:

$\{\texttt{yes}, \texttt{no}, \alpha, \beta', \beta'', \gamma, \gamma', \gamma'', \#, S\} \cup \{\beta_k \mid 0 \leq k \leq 3n + 2p + 2\} \cup$
$\{a_i, a'_i, b_i \mid 1 \leq i \leq n\} \cup \{b'_i \mid 1 \leq i \leq n-1\} \cup$
$\{c_{j,k} \mid 1 \leq j \leq p-1, j \leq k \leq p-1\} \cup \{c_j \mid 1 \leq j \leq p\} \cup$
$\{d_j \mid 2 \leq j \leq p\} \cup \{t_{i,2k}, f_{i,2k} \mid 1 \leq i \leq n-1, i \leq k \leq n-1\} \cup$
$\{T_{i,k}, F'_{i,k} \mid 1 \leq i \leq n, 2i-1 \leq k \leq 2n\} \cup$
$\{T'_{i,k}, F_{i,k} \mid 1 \leq i \leq n-1, 2i+1 \leq k \leq 2n\} \cup$
$\{T^*_{i,j}, F^*_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq p\} \cup$
$\{A_{i,i}, A_{i+1,i}, A'_{i+1,i}, A'_{i+1,i+1}, B_{i,i}, B_{i+1,i}, B'_{i+1,i}, B'_{i+1,i+1} \mid 1 \leq i \leq n-1\} \cup$
$\{A_{n,n}, B_{n,n}\} \cup \{x_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq p\} \cup$
$\{x_{i,j,3k+1}, x_{i,j,3k+2}, x_{i,j,3k+3} \mid 1 \leq i \leq n, 1 \leq j \leq p, 0 \leq k \leq n-1\} \cup$
$\{\overline{x}_{i,j,3k+1}, \overline{x}_{i,j,3k+2}, \overline{x}_{i,j,3k+3} \mid 1 \leq i \leq n, 1 \leq j \leq p, 0 \leq k \leq n-1\} \cup$
$\{{x^*}_{i,j,3k+1}, {x^*}_{i,j,3k+2}, {x^*}_{i,j,3k+3} \mid 1 \leq i \leq n, 1 \leq j \leq p, 0 \leq k \leq n-1\} \cup$
$\{x'_{i,j,3k}, x'_{i,j,3k+1}, x'_{i,j,3k+2} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n-1\} \cup$
$\{\overline{x}'_{i,j,3k}, \overline{x}'_{i,j,3k+1}, \overline{x}'_{i,j,3k+2} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n-1\} \cup$
$\{{x^*}'_{i,j,3k}, {x^*}'_{i,j,3k+1}, {x^*}'_{i,j,3k+2} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n-1\} \cup$
$\{x'_{i,j,3n}, \overline{x}'_{i,j,3n}, {x^*}'_{i,j,3n} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.

2. Alfabeto de entrada Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,j,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$;

3. $\Gamma_0 = \Gamma \setminus \Gamma_1$, $\Gamma_1 = \{T'_{i,k} \mid 1 \leq i \leq n-1, 2i+1 \leq k \leq 2n\} \cup \{F'_{i,k} \mid 1 \leq i \leq n, 2i-1 \leq k \leq 2n\} \cup \{A'_{i+1,i}, A'_{i+1,i+1}, B'_{i+1,i}, B'_{i+1,i+1} \mid 1 \leq i \leq n-1\} \cup \{x'_{i,j,3k}, x'_{i,j,3k+1}, x'_{i,j,3k+2} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n-1\} \cup \{\overline{x}'_{i,j,3k}, \overline{x}'_{i,j,3k+1}, \overline{x}'_{i,j,3k+2} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n-1\} \cup \{{x^*}'_{i,j,3k}, {x^*}'_{i,j,3k+1}, {x^*}'_{i,j,3k+2} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n-1\} \cup \{x'_{i,j,3n}, \overline{x}'_{i,j,3n}, {x^*}'_{i,j,3n} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.

4. $H = \{1, 2\}$.

5. $\mu = [\,[\quad]_2\,]_1$; es decir, $\mu = (V, E)$ donde $V = \{1, 2\}$ y $E = \{(1, 2)\}$.

6. $\mathcal{M}_1 = \{\alpha, \beta_0\}, \mathcal{M}_2 = \{A_{1,1}, B_{1,1}\}$

7. El conjunto $\mathcal{R}$ consiste en las siguientes reglas:

   1.1 Reglas del contador general.
   $[\,\beta_k \to \beta_{k+1}\,]_1$, para $0 \leq k \leq 3n + 2p + 1$
   $[\,\beta_{3n+2p+2} \to \beta'\,]_1$

   1.2 Reglas para producir una respuesta afirmativa.
   $[\,\alpha\gamma \to \gamma'\,]_1$
   $[\,\gamma' \to \gamma''\,]_1$
   $[\,\gamma''\,]_1 \to \texttt{yes}\,[\quad]_1$

1.3 Reglas para producir una respuesta negativa.

$[\,\alpha\beta' \to \beta''\,]_1$

$[\,\beta''\,]_1 \to \texttt{no}\,[\quad]_1$

2.1 Regla para producir dos nuevas membranas a partir de una membrana.

$[\,S\,]_2 \to [\,\Gamma_0\,]_2\,[\,\Gamma_1\,]_2$

2.2 Reglas para generar las valoraciones de verdad.

$$\left.\begin{array}{l}[\,A_{i,i} \to a_i a'_i\,]_2\\ [\,B_{i,i} \to b_i b'_i\,]_2\end{array}\right\}\ 1 \le i \le n-1$$

$$\left.\begin{array}{l}[\,A'_{i,i} \to a_i a'_i\,]_2\\ [\,B'_{i,i} \to b_i b'_i\,]_2\end{array}\right\}\ 2 \le i \le n-1$$

$[\,A_{n,n} \to a_n a'_n\,]_2$

$[\,A'_{n,n} \to a_n a'_n\,]_2$

$[\,B_{n,n} \to b_n\,]_2$

$[\,B'_{n,n} \to b_n\,]_2$

$$\left.\begin{array}{l}[\,a_i \to T_{i,2i-1}A_{i+1,i}\,]_2\\ [\,a'_i \to F'_{i,2i-1}A'_{i+1,i}\,]_2\\ [\,b_i \to B_{i+1,i}S\,]_2\\ [\,b'_i \to B'_{i+1,i}\,]_2\end{array}\right\}\ 1 \le i \le n-1$$

$[\,a_n \to T_{n,2n-1}\,]_2$

$[\,a'_n \to F'_{n,2n-1}\,]_2$

$[\,b_n \to S\,]_2$

$$\left.\begin{array}{l}[\,T_{i,2j+1} \to T_{i,2j+2}\,]_2\\ [\,T'_{i,2j+1} \to T'_{i,2j+2}\,]_2\\ [\,F_{i,2j+1} \to F_{i,2j+2}\,]_2\\ [\,F'_{i,2j+1} \to F'_{i,2j+2}\,]_2\end{array}\right\}\ 1 \le i \le n-1, i \le j \le n-1$$

$$\left.\begin{array}{l}[\,T_{i,2i-1} \to T_{i,2i}\,]_2\\ [\,F'_{i,2i-1} \to F'_{i,2i}\,]_2\end{array}\right\}\ 1 \le i \le n$$

$$\left.\begin{array}{l}[\,A_{i+1,i} \to A_{i+1,i+1}\,]_2\\ [\,A'_{i+1,i} \to A'_{i+1,i+1}\,]_2\\ [\,B_{i+1,i} \to B_{i+1,i+1}\,]_2\\ [\,B'_{i+1,i} \to B'_{i+1,i+1}\,]_2\end{array}\right\}\ 1 \le i \le n-1$$

$$\left.\begin{array}{l}[\,T_{i,2j} \to t_{i,2j}\,]_2\\ [\,T'_{i,2j} \to t_{i,2j}\,]_2\\ [\,F_{i,2j} \to f_{i,2j}\,]_2\\ [\,F'_{i,2j} \to f_{i,2j}\,]_2\end{array}\right\}\ 1 \le i \le n-2, i+1 \le j \le n-1$$

$$\left.\begin{array}{l}[\,T_{i,2i} \to t_{i,2i}\,]_2 \\ [\,F'_{i,2i} \to f_{i,2i}\,]_2\end{array}\right\}\ 1 \le i \le n-1$$

$$\left.\begin{array}{l}[\,t_{i,2j} \to T_{i,2j+1}T'_{i,2j+1}\,]_2 \\ [\,f_{i,2j} \to F_{i,2j+1}F'_{i,2j+1}\,]_2\end{array}\right\}\ 1 \le i \le n-1, i \le j \le n-1$$

2.3 Reglas para preparar la fórmula φ para chequear las cláusulas.

$$\left.\begin{array}{l}[\,x_{i,j,3k} \to x_{i,j,3k+1}\,]_2 \\ [\,\overline{x}_{i,j,3k} \to \overline{x}_{i,j,3k+1}\,]_2 \\ [\,x^*_{i,j,3k} \to x^*_{i,j,3k+1}\,]_2 \\ [\,x_{i,j,3k+1} \to x_{i,j,3k+2}\,]_2 \\ [\,\overline{x}_{i,j,3k+1} \to \overline{x}_{i,j,3k+2}\,]_2 \\ [\,x^*_{i,j,3k+1} \to x^*_{i,j,3k+2}\,]_2 \\ [\,x_{i,j,3k+2} \to x_{i,j,3k+3}x'_{i,j,3k+3}\,]_2 \\ [\,\overline{x}_{i,j,3k+2} \to \overline{x}_{i,j,3k+3}\overline{x}'_{i,j,3k+3}\,]_2 \\ [\,x^*_{i,j,3k+2} \to x^*_{i,j,3k+3}{x^*}'_{i,j,3k+3}\,]_2\end{array}\right\}\ \begin{array}{l}1 \le i \le n, \\ 1 \le j \le p, \\ 0 \le k \le n-1\end{array}$$

$$\left.\begin{array}{l}[\,x'_{i,j,3k} \to x'_{i,j,3k+1}\,]_2 \\ [\,\overline{x}'_{i,j,3k} \to \overline{x}'_{i,j,3k+1}\,]_2 \\ [\,{x^*}'_{i,j,3k} \to {x^*}'_{i,j,3k+1}\,]_2 \\ [\,x'_{i,j,3k+1} \to x'_{i,j,3k+2}\,]_2 \\ [\,\overline{x}'_{i,j,3k+1} \to \overline{x}'_{i,j,3k+2}\,]_2 \\ [\,{x^*}'_{i,j,3k+1} \to {x^*}'_{i,j,3k+2}\,]_2 \\ [\,x'_{i,j,3k+2} \to x_{i,j,3k+3}x'_{i,j,3k+3}\,]_2 \\ [\,\overline{x}'_{i,j,3k+2} \to \overline{x}_{i,j,3k+3}\overline{x}'_{i,j,3k+3}\,]_2 \\ [\,{x^*}'_{i,j,3k+2} \to {x^*}_{i,j,3k+3}{x^*}'_{i,j,3k+3}\,]_2\end{array}\right\}\ \begin{array}{l}1 \le i \le n, \\ 1 \le j \le p, \\ 1 \le k \le n-1\end{array}$$

$$\left.\begin{array}{l}[\,x_{i,j,3n} \to x_{i,j}\,]_2 \\ [\,\overline{x}_{i,j,3n} \to \overline{x}_{i,j}\,]_2 \\ [\,x^*_{i,j,3n} \to x^*_{i,j}\,]_2 \\ [\,x'_{i,j,3n} \to x_{i,j}\,]_2 \\ [\,\overline{x}'_{i,j,3n} \to \overline{x}_{i,j}\,]_2 \\ [\,{x^*}'_{i,j,3n} \to x^*_{i,j}\,]_2\end{array}\right\}\ 1 \le i \le n, 1 \le j \le p$$

2.4 Reglas para preparar las valoraciones de verdad para chequear las cláusulas.

$$\left.\begin{array}{l}[\,T_{i,2n} \to T^*_{i,1}\,]_2 \\ [\,T'_{i,2n} \to T^*_{i,1}\,]_2 \\ [\,F_{i,2n} \to F^*_{i,1}\,]_2 \\ [\,F'_{i,2n} \to F^*_{i,1}\,]_2\end{array}\right\}\ 1 \le i \le n-1$$

$$[\,T_{n,2n} \to T^*_{n,1}\,]_2$$

$$[\,F'_{n,2n} \to F^*_{n,1}\,]_2$$

2.5 Reglas para chequear las cláusulas.

$$\left.\begin{array}{l}[\, T^*_{i,j}x_{i,j} \to T^*_{i,j+1}c_{j,j}\,]_2\\ [\, T^*_{i,j}\overline{x}_{i,j} \to T^*_{i,j+1}\,]_2\\ [\, T^*_{i,j}x^*_{i,j} \to T^*_{i,j+1}\,]_2\\ [\, F^*_{i,j}x_{i,j} \to F^*_{i,j+1}\,]_2\\ [\, F^*_{i,j}\overline{x}_{i,j} \to F^*_{i,j+1}c_{j,j}\,]_2\\ [\, F^*_{i,j}x^*_{i,j} \to F^*_{i,j+1}\,]_2\end{array}\right\} 1 \le i \le n, 1 \le j \le p-1$$

$$\left.\begin{array}{l}[\, T^*_{i,p}x_{i,p} \to c_p\,]_2\\ [\, T^*_{i,p}\overline{x}_{i,p} \to \#\,]_2\\ [\, T^*_{i,p}x^*_{i,p} \to \#\,]_2\\ [\, F^*_{i,p}x_{i,p} \to \#\,]_2\\ [\, F^*_{i,p}\overline{x}_{i,p} \to c_p\,]_2\\ [\, F^*_{i,p}x^*_{i,p} \to \#\,]_2\end{array}\right\} 1 \le i \le n$$

$[\, c_{j,k} \to c_{j,k+1}\,]_2$, para $1 \le j \le p-2, j \le k \le p-2$

$[\, c_{j,p-1} \to c_j\,]_2$, para $1 \le j \le p-1$

2.6 Reglas para detectar si una valoración de verdad hace verdadera la fórmula φ.

$[\, c_1c_2 \to d_2\,]_2$

$[\, d_jc_{j+1} \to d_{j+1}\,]_2$, para$2 \le j \le p-1$

$[\, d_p\,]_2 \to \gamma\,[\quad]_2$

8. La membrana de entrada es la membrana etiquetada por 2 ($i_{in} = 2$) y la región de salida es el entorno ($i_{out} = env$).

Obsérvese que para cada $u \in I_{\mathtt{SAT}}$, el sistema $\Pi(s(u)) + cod(u)$ es determinista.

Consideramos la codificación polinomial (cod, s) de SAT en $\mathbf{\Pi}$ definida como sigue: sea $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_p, C_j = l_{j_1} \vee \cdots \vee l_{j_{r_j}}, 1 \le j \le p, l_{j_k} \in \{x_i, \neg x_i \mid 1 \le i \le n\}$, instancia del problema SAT será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$, en donde $s(\varphi) = \langle n, p\rangle$ y $cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$.

Además, denotamos por $cod_k(\varphi)$ el conjunto $cod(\varphi)$ cuando el tercer subíndice de todos sus elementos es igual a k y por $cod'_k(\varphi)$ su versión primada.

Denotamos por $(cod^*(\varphi))^p_j$ la codificación de las cláusulas $C_j, \ldots, C_p$; es decir, la expresión que tiene desde la j-ésima hasta la p-ésima fila, y donde los objetos correspondientes sólo conservan los dos primeros subíndices.

La fórmula Booleana φ será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$. A continuación, daremos una descripción informal de como funciona el sistema.

La solución propuesta sigue un algoritmo de fuerza bruta en el marco de sistemas P reconocedores de membranas activas, cooperación minimal primaria en reglas de evolución y separación sólo para membranas elementales, y consiste en las siguientes fases:

- *Fase de generación*: Usando las reglas de separación, junto con las reglas de 2.2, 2.3 y 2.4 que «simulan» las reglas de división, todas las valoraciones de verdad para las variables $\{x_1, \ldots, x_n\}$ asociadas a φ son producidas. Específicamente, se generan 2^n membranas etiquetadas por 2, cada una de ellas codificando una valoración de verdad diferente. Esta fase tarda $3n+1$ pasos de computación exactamente (el último paso está dedicado a preparar el sistema para comenzar la siguiente fase), siendo n el número de variables de φ.

- *Primera fase de chequeo*: Mediante el uso de las reglas de 2.5, se chequea si una cláusula de φ es satisfecha o no por cada valoración de verdad codificada en cada membrana etiquetada por 2. Esta fase tarda exactamente p pasos de computación, siendo p el número de cláusulas de φ.

- *Segunda fase de chequeo*: En esta fase se verifica en las membranas etiquetadas por 2 si las valoraciones de verdad codificadas en las mismas hacen verdaderas todas las cláusulas de φ y, por tanto, la fórmula en sí. Para ello se van disparando las reglas de 2.6. Esta fase tarda $p-1$ pasos de computación.

- *Fase de salida*: El sistema usa las reglas de 1.2 y 1.3 dependiendo si la fórmula φ es satisfactible o no, para enviar al entorno el objeto adecuado en el último paso de computación. Esta fase dura exactamente 4 pasos de computación, independientemente de si la fórmula sea satisfactible o no.

6.2.1.1. Verificación formal

A continuación, vamos a verificar formalmente la solución propuesta para el problema `SAT`.

Fase de generación

En esta fase, todas las valoraciones de verdad asociadas a la fórmula φ serán generadas por la aplicación de las reglas de 2.1, 2.2, 2.3 y 2.4 en las membranas etiquetadas por 2, de tal manera que en los pasos $3i$ $(1 \leq i \leq n-1)$ de esta fase,

la regla de separación asociada al objeto S será disparada, produciendo dos nuevas membranas con todos los objetos de la membrana original distribuidos en las nuevas membranas creadas con la misma etiqueta que la original. En el último paso de esta fase los objetos producidos son las diferentes valoraciones de verdad codificadas por los objetos $T_{i,1}$ y $F_{i,1}$, para $1 \leq i \leq n$ y los objetos $x_{i,j}, \overline{x}_{i,j}, x^*_{i,j}, 1 \leq i \leq n, 1 \leq j \leq p$.

Proposición 6.5. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

- *En la configuración* $\mathcal{C}_1$*:*
  - $\mathcal{C}_1(1) = \{\alpha, \beta_1\}$.
  - *Hay una membrana etiquetada con 2 que contiene los objetos* a_1, a'_1, b_1, b'_1 *y el multiconjunto* $cod_1(\varphi)$.
- *En la configuración* $\mathcal{C}_2$*:*
  - $\mathcal{C}_2(1) = \{\alpha, \beta_2\}$.
  - *En la configuración* $\mathcal{C}_2$ *hay una membrana etiquetada por 2 que contiene el objeto* $T_{1,1}, F'_{1,1}, A_{2,1}, A'_{2,1}, B_{2,1}, B'_{2,1}, S$ *y el multiconjunto* $cod_2(\varphi)$.

Demostración. Se va a probar a continuación.

- En la configuración $\mathcal{C}_0$:
  - $\mathcal{C}_0(1) = \{\alpha, \beta_0\}$.
  - Hay una membrana etiquetada por 2 que contiene los objetos $A_{1,1}, B_{1,1}$ y el multiconjunto $cod_0(\varphi)$.
- La configuración $\mathcal{C}_1$ se obtiene de la configuración $\mathcal{C}_0$ por la aplicación de las reglas $[\,\beta_0 \to \beta_1\,]_1, [\,A_{1,1} \to a_1 a'_1\,]_2, [\,x'_{i,j,3k+2} \to x_{i,j,3k+3} x'_{i,j,3k+3}\,]_2$, $[\,B_{1,1} \to b_1 b'_1\,]_2, [\,x_{i,j,0} \to x_{i,j,1}\,]_2, [\,\overline{x}_{i,j,0} \to \overline{x}_{i,j,1}\,]_2$ y $[\,x^*_{i,j,0} \to x^*_{i,j,1}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.
- La configuración $\mathcal{C}_2$ se obtiene de la configuración $\mathcal{C}_1$ por la aplicación de las reglas $[\,\beta_1 \to \beta_2\,]_1, [\,a_1 \to T_{1,1} A_{2,1}\,]_2, [\,a'_1 \to F'_{1,1} A'_{2,1}\,]_2, [\,b_1 \to B_{2,1} S\,]_2, [\,b'_1 \to B'_{2,1}\,]_2, [\,x_{i,j,1} \to x_{i,j,2}\,]_2, [\,\overline{x}_{i,j,1} \to \overline{x}_{i,j,2}\,]_2$ y $[\,x^*_{i,j,1} \to x^*_{i,j,2}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

□

Proposición 6.6. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a) *Para cada* $k, 1 \leq k \leq n-2$, *lo siguiente es válido:*

- *En la configuración* $\mathcal{C}_{3k}$*:*
  - $\mathcal{C}_{3k}(1) = \{\alpha, \beta_{3k}\}$.
  - *Hay* 2^k *membranas etiquetadas por 2 such that:*
    - 2^{k-1} *de estas membranas contienen el multiconjunto* $cod_{3k}(\varphi)$, *objetos* $A_{k+1,k+1}, B_{k+1,k+1}$ *y un subconjunto diferente* $\{R_{1,2k}, \ldots, R_{k-1,2k}, T_{k,2k}\}$, *siendo* $R \in \{T, F\}$.
    - 2^{k-1} *de estas membranas contienen el multiconjunto* $cod'_{3k}(\varphi)$, *objetos* $A'_{k+1,k+1}, B'_{k+1,k+1}$ *y un subconjunto diferente* $\{R'_{1,2k}, \ldots, R'_{k-1,2k}, F'_{k,2k}\}$, *siendo* $R \in \{T, F\}$.
- *En la configuración* $\mathcal{C}_{3k+1}$*:*
  - $\mathcal{C}_{3k+1}(1) = \{\alpha, \beta_{3k+1}\}$.
  - *Hay* 2^k *membranas etiquetadas por 2 tales que cada una de ellas contiene los objetos* $a_{k+1}, a'_{k+1}, b_{k+1}, b'_{k+1}$. *Además,*
    - 2^{k-1} *de estas membranas contiene el multiconjunto* $cod_{3k+1}(\varphi)$ *y un subconjunto diferente* $\{r_{1,2k}, \ldots, r_{k-1,2k}, t_{k,2k}\}$, *siendo* $r \in \{t, f\}$.
    - 2^{k-1} *de estas membranas contiene el multiconjunto* $cod'_{3k+1}(\varphi)$ *y un subconjunto diferente* $\{r'_{1,2k}, \ldots, r'_{k-1,2k}, f_{k,2k}\}$, *siendo* $r \in \{t, f\}$.
- *En la configuración* $\mathcal{C}_{3k+2}$*:*
  - $\mathcal{C}_{3k+2}(1) = \{\alpha, \beta_{3k+2}\}$.
  - *Hay* 2^k *membranas etiquetadas por 2 tales que contienen los objetos* $T_{k+1,2(k+1)-1}, F'_{k+1,2(k+1)-1}, A_{k+2,k+1}, A'_{k+2,k+1}, B_{k+2,k+1}, B'_{k+2,k+1}$ *y* S. *Además,*
    - 2^{k-1} *de estas membranas contiene el multiconjunto* $cod_{3k+2}(\varphi)$ *y un subconjunto diferente* $\{R_{1,2(k+1)-1}, R'_{1,2(k+1)-1}, \ldots, R_{k,2(k+1)-1}, R'_{k,2(k+1)-1}\}$, *siendo* $R \in \{T, F\}$.
    - 2^{k-1} *de estas membranas contiene el multiconjunto* $cod'_{3k+2}(\varphi)$ *y un subconjunto diferente* $\{R_{1,2(k+1)-1}, R'_{1,2(k+1)-1}, \ldots, R_{k-1,2(k+1)-1}, R'_{k-1,2(k+1)-1}\}$, *siendo* $R \in \{T, F\}$.

(*b*) *En la configuración* $\mathcal{C}_{3n-3} = \mathcal{C}_{3(n-1)}$*:*

- $\mathcal{C}_{3n-3}(1) = \{\alpha, \beta_{3n-3}\}$.
- *Hay* 2^{n-1} *membranas etiquetadas por 2 tales que:*
  - 2^{n-2} *de estas membranas contiene el multiconjunto* $cod_{3n-3}(\varphi)$*, los objetos* $A_{n,n}, B_{n,n}$ *y un subconjunto diferente* $\{R_{1,2(n-1)}, \ldots, R_{n-2,2(n-1)}, T_{n-1,2(n-1)}\}$*, siendo* $R \in \{T, F\}$.
  - 2^{n-2} *de estas membranas contiene el multiconjunto* $cod'_{3n-3}(\varphi)$*, los objetos* $A'_{n,n}, B'_{n,n}$ *y un subconjunto diferente* $\{R'_{1,2(n-1)}, \ldots, R'_{n-2,2(n-1)}, F'_{n-1,2(n-1)}\}$*, siendo* $R \in \{T, F\}$.

(*c*) *En la configuración* $\mathcal{C}_{3n-2} = \mathcal{C}_{3(n-1)+1}$*:*

- $\mathcal{C}_{3n-2}(1) = \{\alpha, \beta_{3n-2}\}$.
- *Hay* 2^{n-1} *membranas etiquetadas por 2 tales que contienen los objetos* a_n, a'_n, b_n*. Además,*
  - 2^{n-2} *de estas membranas contiene el multiconjunto* $cod_{3n-2}(\varphi)$ *y un subconjunto diferente* $\{r_{1,2(n-1)}, \ldots, r_{n-2,2(n-1)}, t_{n-1,2(n-1)}\}$*, siendo* $r \in \{t, f\}$.
  - 2^{k-1} *de estas membranas contiene el multiconjunto* $cod'_{3n-2}(\varphi)$ *y un subconjunto diferente* $\{r_{1,2(n-1)}, \ldots, r_{n-2,2(n-1)}, f_{n-1,2(n-1)}\}$*, siendo* $r \in \{t, f\}$.

(*d*) *En la configuración* $\mathcal{C}_{3n-1} = \mathcal{C}_{3(n-1)+2}$*:*

- $\mathcal{C}_{3n-1}(1) = \{\alpha, \beta_{3n-1}\}$.
- *Hay* 2^{n-1} *membranas etiquetadas por 2 tales que contienen los objetos* $T_{n,2n-1}, F'_{n,2n-1}, S$ *y un subconjunto diferente* $\{R_{1,2n-1}, R'_{1,2n-1}, \ldots, R_{n-2,2n-1}, R'_{n-2,2n-1}\}$*, siendo* $R \in \{T, F\}$.
  Además,
  - 2^{n-2} *de estas membranas contains los objetos* $T_{n-1,2n-1}, T'_{n-1,2n-1}$ *y el multiconjunto* $cod_{3n-1}(\varphi)$.
  - 2^{n-2} *de estas membranas contains los objetos* $F_{n-1,2n-1}, F'_{n-1,2n-1}$ *y el multiconjunto* $cod'_{3n-1}(\varphi)$.

(*e*) *En la configuración* $\mathcal{C}_{3n}$*:*

- $\mathcal{C}_{3n}(1) = \{\alpha, \beta_{3n}\}$.

- *Hay* 2^n *membranas etiquetadas por 2 tales que:*
  - 2^{n-1} *de estas membranas contiene el multiconjunto* $cod_{3n}(\varphi)$ *y un subconjunto diferente* $\{R_{1,2n}, \ldots, R_{n-1,2n}, T_{n,2n}\}$, *siendo* $R \in \{T, F\}$.
  - 2^{n-1} *de estas membranas contiene el multiconjunto* $cod'_{3n}(\varphi)$ *y un subconjunto diferente* $\{R'_{1,2n}, \ldots, R'_{n-1,2n}, F'_{n,2n}\}$, *siendo* $R \in \{T, F\}$.

(f) *En la configuración* $\mathcal{C}_{3n+1}$*:*

- $\mathcal{C}_{3n+1}(1) = \{\alpha, \beta_{3n+1}\}$.
- *Hay* 2^n *membranas etiquetadas por 2 tales que each of them contains multiset* $cod(\varphi)$ *y un subconjunto diferente* $\{R^*_{1,1}, \ldots, R^*_{n,1}\}$, *siendo* $R \in \{T, F\}$.

Demostración. Se demostrará por inducción sobre k.

(a) Por inducción sobre k. Para probar el caso base $k = 1$ basta tener presente la Proposición 6.5 y observar que:

- La configuración $\mathcal{C}_3$ se obtiene de la configuración $\mathcal{C}_2$ por la aplicación de las reglas $[\,\beta_2 \to \beta_3\,]_1, [\,T_{1,1} \to T_{1,2}\,]_2, [\,F'_{1,1} \to F'_{1,2}\,]_2, [\,A_{2,1} \to A_{2,2}\,]_2, [\,A'_{2,1} \to A'_{2,2}\,]_2, [\,B_{2,1} \to B_{2,2}\,]_2, [\,B'_{2,1} \to B'_{2,2}\,]_2, [\,x_{i,j,2} \to x_{i,j,3}\,]_2, [\,\overline{x}_{i,j,2} \to \overline{x}_{i,j,3}\,]_2$ y $[\,x^*_{i,j,2} \to x^*_{i,j,3}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.
- La configuración $\mathcal{C}_4$ se obtiene de la configuración $\mathcal{C}_3$ por la aplicación de las reglas $[\,\beta_3 \to \beta_4\,]_1, [\,T_{1,2} \to t_{1,2}\,]_2, [\,F'_{1,2} \to f_{1,2}\,]_2, [\,A_{2,2} \to a_2 a'_2\,]_2, [\,A'_{2,2} \to a_2 a'_2\,]_2, [\,B_{2,2} \to b_2 b'_2\,]_2, [\,B'_{2,2} \to b_2 b'_2\,]_2, [\,x_{i,j,3} \to x_{i,j,4}\,]_2, [\,\overline{x}_{i,j,3} \to \overline{x}_{i,j,4}\,]_2$ y $[\,x^*_{i,j,3} \to x^*_{i,j,4}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.
- La configuración $\mathcal{C}_5$ se obtiene de la configuración $\mathcal{C}_4$ por la aplicación de las reglas $[\,\beta_4 \to \beta_5\,]_1$, $[\,t_{1,2} \to T_{1,3}T'_{1,3}\,]_2, [\,f_{1,2} \to F_{1,3}F'_{1,3}\,]_2$, $[\,a_2 \to T_{2,3}A_{3,2}\,]_2, [\,a'_2 \to F'_{2,3}A'_{3,2}\,]_2, [\,b_2 \to B_{3,2}S\,]_2, [\,b'_2 \to B'_{3,2}\,]_2$, $[\,x_{i,j,4} \to x_{i,j,5}\,]_2, [\,\overline{x}_{i,j,4} \to \overline{x}_{i,j,5}\,]_2$ y $[\,x^*_{i,j,4} \to x^*_{i,j,5}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

Supongamos ahora, por hipotesis de induccion, que el resultado es cierto para k, siendo $1 \leq k < n - 2$. On the one hand, at configuration $\mathcal{C}_{3k+2}$ we have:

- $\mathcal{C}_{3k+2}(1) = \{\alpha, \beta_{3k+2}\}$.
- Hay 2^k membranas etiquetadas por 2 tales que contienen los objetos $T_{k+1,2k+1}, F'_{k+1,2k+1}, A_{k+2,k+1}, A'_{k+2,k+1}, B_{k+2,k+1}, B'_{k+2,k+1}$ y S. Además,
  - 2^{k-1} de estas membranas contiene el multiconjunto $cod_{3k+2}(\varphi)$ y un subconjunto diferente $\{R_{1,2k+1}, R'_{1,2k+1}, \ldots, R_{k-1,2k+1}, R'_{k-1,2k+1}, T_{k,2k+1}, T'_{k,2k+1}\}$, siendo $R \in \{T, F\}$.
  - 2^{k-1} de estas membranas contiene el multiconjunto $cod'_{3k+2}(\varphi)$ y un subconjunto diferente $\{R_{1,2k+1}, R'_{1,2k+1}, \ldots, R_{k-1,2k+1}, R'_{k-1,2k+1}, F_{k,2k+1}, F'_{k,2k+1}\}$, siendo $R \in \{T, F\}$.

Por otra parte, la configuración $\mathcal{C}_{3(k+1)} = \mathcal{C}_{3k+3}$ se obtiene de la configuración $\mathcal{C}_{3k+2}$ por la aplicación de las reglas:

$[\,\beta_{3k+2} \to \beta_{3k+3}\,]_1, [\,S\,]_2 \to [\,\Gamma_0\,]_2\,[\,\Gamma_1\,]_2, [\,T_{l,2k+1} \to T_{l,2k+2}\,]_2, [\,T'_{l,2k+1} \to T'_{l,2k+2}\,]_2, [\,F_{l,2k+1} \to F_{l,2k+2}\,]_2, [\,F'_{l,2k+1} \to F'_{l,2k+2}\,]_2$, para $1 \leq l \leq k-1, [\,T_{k,2k+1} \to T_{k,2k+2}\,]_2, [\,F'_{k,2k+1} \to F'_{k,2k+2}\,]_2, [\,A_{k+1,k} \to A_{k+1,k+1}\,]_2, [\,A'_{k+1,k} \to A'_{k+1,k+1}\,]_2, [\,B_{k+1,k} \to B_{k+1,k+1}\,]_2, [\,B'_{k+1,k} \to B'_{k+1,k+1}\,]_2, [\,x_{i,j,3k+2} \to x_{i,j,3k+3}\,]_2, [\,\overline{x}_{i,j,3k+2} \to \overline{x}_{i,j,3k+3}\,]_2\ [\,x^*_{i,j,3k+2} \to x^*_{i,j,3k+3}\,]_2, [\,x'_{i,j,3k+2} \to x'_{i,j,3k+3}\,]_2, [\,\overline{x}'_{i,j,3k+2} \to \overline{x}'_{i,j,3k+3}\,]_2\ [\,x'^*_{i,j,3k+2} \to x'^*_{i,j,3k+3}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

Por lo tanto, en la configuración $\mathcal{C}_{3(k+1)} = \mathcal{C}_{3k+3}$ tenemos lo siguiente:

- $\mathcal{C}_{3(k+1)}(1) = \{\alpha, \beta_{3k+3}\}$.
- Hay 2^{k+1} membranas etiquetadas por 2 tales que:
  - 2^k de estas membranas contiene el multiconjunto $cod_{3(k+1)}(\varphi)$, los objetos $A_{k+1,k+1}, B_{k+1,k+1}$ y un subconjunto diferente $\{R_{1,2(k+1)}, \ldots, R_{k,2(k+1)}, T_{k+1,2(k+1)}\}$, siendo $R \in \{T, F\}$.
  - 2^k de estas membranas contiene el multiconjunto $cod'_{3(k+1)}(\varphi)$, los objetos $A'_{k+2,k+2}, B'_{k+2,k+2}$ y un subconjunto diferente $\{R'_{1,2(k+1)}, \ldots, R'_{k,2(k+1)}, F'_{k+1,2(k+1)}\}$, siendo $R \in \{T, F\}$.

La configuración $\mathcal{C}_{3k+4}$ se obtiene de la configuración $\mathcal{C}_{3k+3}$ por la aplicación de las reglas:

$[\,\beta_{3k+3} \to \beta_{3k+4}\,]_1, [\,T_{l,2(k+1)} \to t_{l,2(k+1)}\,]_2, [\,T'_{l,2(k+1)} \to t_{l,2(k+1)}\,]_2, [\,F_{l,2(k+1)} \to f_{l,2(k+1)}\,]_2, [\,F'_{l,2(k+1)} \to f_{l,2(k+1)}\,]_2$, para $1 \leq l \leq k, [\,T_{k+1,2(k+1)} \to t_{k+1,2(k+1)}\,]_2, [\,F'_{k+1,2(k+1)} \to f_{k+1,2(k+1)}\,]_2, [\,A_{k+1,k+1} \to b_{k+1}b'_{k+1}\,]_2, [\,A'_{k+1,k+1} \to b_{k+1}, b'_{k+1}\,]_2, [\,B_{k+1,k+1} \to b_{k+1}, b'_{k+1}\,]_2, [\,B'_{k+1,k+1} \to b_{k+1}, b'_{k+1}\,]_2, [\,x_{i,j,3k+3} \to x_{i,j,3k+4}\,]_2,$

$[\,\overline{x}_{i,j,3k+3} \rightarrow \overline{x}_{i,j,3k+4}\,]_2, [\,x^*_{i,j,3k+3} \rightarrow x^*_{i,j,3k+4}\,]_2, [\,x'_{i,j,3k+3} \rightarrow x'_{i,j,3k+4}\,]_2$, $[\,\overline{x}'_{i,j,3k+3} \rightarrow \overline{x}'_{i,j,3k+4}\,]_2, [\,x'^*_{i,j,3k+3} \rightarrow x'^*_{i,j,3k+3}\,]_4$, para $1 \leq i \leq n, 1 \leq j \leq p$.

Por lo tanto, en la configuración $\mathcal{C}_{3(k+1)+1} = \mathcal{C}_{3k+4}$ tendremos lo siguiente:

- $\mathcal{C}_{3(k+1)+1}(1) = \{\alpha, \beta_{3k+4}\}$.
- Hay 2^{k+1} membranas etiquetadas por 2 tales que contienen los objetos $a_{k+2}, a'_{k+2}, b_{k+2}, b'_{k+2}$. Además,
  - 2^k de estas membranas contiene el multiconjunto $cod_{3(k+1)+1}(\varphi)$ y un subconjunto diferente $\{r_{1,2(k+1)}, \ldots, r_{k,2(k+1)}, t_{k+1,2(k+1)}\}$, siendo $r \in \{t, f\}$.
  - 2^k de estas membranas contiene el multiconjunto $cod'_{3(k+1)+1}(\varphi)$ y un subconjunto diferente $\{r'_{1,2(k+1)}, \ldots, r'_{k,2(k+1)}, f_{k+1,2(k+1)}\}$, siendo $r \in \{t, f\}$.

La configuración $\mathcal{C}_{3k+5}$ se obtiene de la configuración $\mathcal{C}_{3k+4}$ por la aplicación de las reglas: $[\,\beta_{3k+4} \rightarrow \beta_{3k+5}\,]_1, [\,t_{l,2(k+1)} \rightarrow T_{l,2(k+1)+1}T'_{l,2(k+1)+1}\,]_2$, $[\,f_{l,2(k+1)} \rightarrow F_{l,2(k+1)+1}F'_{l,2(k+1)+1}\,]_2$, para $1 \leq l \leq k+1$, $[\,a_{k+1} \rightarrow T_{k+1,2(k+1)-1}A_{k+2,k+1}\,]_2, [\,a'_{k+1} \rightarrow F'_{k+1,2(k+1)-1}A'_{k+2,k+1}\,]_2, [\,b_{k+1} \rightarrow B_{k+2,k+1}S\,]_2, [\,b'_{k+1} \rightarrow B'_{k+2,k+1}\,]_2, [\,x_{i,j,3k+4} \rightarrow x_{i,j,3k+5}\,]_2$, $[\,\overline{x}_{i,j,3k+4} \rightarrow \overline{x}_{i,j,3k+5}\,]_2, [\,x^*_{i,j,3k+4} \rightarrow x^*_{i,j,3k+5}\,]_2, [\,x'_{i,j,3k+4} \rightarrow x'_{i,j,3k+5}\,]_2$, $[\,\overline{x}'_{i,j,3k+4} \rightarrow \overline{x}'_{i,j,3k+5}\,]_2$ y $[\,x'^*_{i,j,3k+4} \rightarrow x'^*_{i,j,3k+5}\,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

Por lo tanto, en la configuración $\mathcal{C}_{3(k+1)+2} = \mathcal{C}_{3k+5}$ tendremos lo siguiente:

- $\mathcal{C}_{3(k+1)+2}(1) = \{\alpha, \beta_{3k+5}\}$.
- Hay 2^{k+1} membranas etiquetadas por 2 tales que contienen los objetos $T_{k+1,2(k+1)-1}, F'_{k+1,2(k+1)-1}, A_{k+2,k+1}, A'_{k+2,k+1}, B_{k+2,k+1}, B'_{k+2,k+1}$ y S. Además,
  - 2^k de estas membranas contiene el multiconjunto $cod_{3(k+1)+2}(\varphi)$ y un subconjunto diferente $\{R_{1,2(k+1)+1}, R'_{1,2(k+1)+1}, \ldots, R_{k,2(k+1)+1}, R'_{k,2(k+1)+1}, T_{k+1,2(k+1)+1}, F'_{k+1,2(k+1)+1}\}$, siendo $R \in \{T, F\}$.
  - 2^k de estas membranas contiene el multiconjunto $cod'_{3(k+1)+2}(\varphi)$ y un subconjunto diferente $\{R_{1,2(k+1)+1}, R'_{1,2(k+1)+1}, \ldots, R_{k,2(k+1)+1}, R'_{k,2(k+1)+1}, T_{k+1,2(k+1)+1}, F'_{k+1,2(k+1)+1}\}$, siendo $R \in \{T, F\}$.

Por tanto, el resultado se verifica para $k+1$. Lo que completa la prueba del apartado (a).

(b) Aplicando el caso (a) para $k = n-2$ resulta que en la configuración $\mathcal{C}_{3(n-2)+2} = \mathcal{C}_{3n-4}$:

- $\mathcal{C}_{3n-4}(1) = \{\alpha, \beta_{3n-4}\}$.
- Hay 2^{n-2} membranas etiquetadas por 2 tales que contienen los objetos $T_{n-1,2(n-1)-1}, F'_{n-1,2(n-1)-1}, A_{n,n-1}, A'_{n,n-1}, B_{n,n-1}, B'_{n,n-1}, S$ y un subconjunto diferente $\{R_{1,2(n-1)-1}, R'_{1,2(n-1)-1}, \ldots, R_{n-3,2(n-1)-1}, R'_{n-3,2(n-1)-1}\}$, siendo $R \in \{T, F\}$. Además,
  - 2^{n-3} de estas membranas contains los objetos $T_{n-2,2(n-1)-1}$, $T'_{n-2,2(n-1)-1}$ y el multiconjunto $cod_{3n-4}(\varphi)$.
  - 2^{n-3} de estas membranas contains los objetos $F_{n-2,2(n-1)-1}$, $F'_{n-2,2(n-1)-1}$ y el multiconjunto $cod'_{3n-4}(\varphi)$.

Por lo tanto, (b) es válido observando que $\mathcal{C}_{3n-3}$ se obtiene de $\mathcal{C}_{3n-4}$ por la aplicación de las reglas

$[\,\beta_{3n-4} \to \beta_{3n-3}\,]_1, [\,S\,]_2 \to [\,\Gamma_0\,]_2\,[\,\Gamma_1\,]_2, [\,T_{l,2(n-2)+1} \to T_{l,2(n-1)}\,]_2$, $[\,T'_{l,2(n-2)+1} \to T'_{l,2(n-1)}\,]_2, [\,F_{l,2(n-2)+1} \to F_{l,2(n-1)}\,]_2, [\,F'_{l,2(n-2)+1} \to F'_{l,2(n-1)}\,]_2$, para $1 \le l \le n-2, [\,T_{n-1,2(n-2)+1} \to T_{n-1,2(n-1)}\,]_2, [\,F'_{n-1,2(n-2)+1} \to F'_{n-1,2(n-1)}\,]_2, [\,A_{n,n-1} \to A_{n.n}\,]_2, [\,A'_{n,n-1} \to A'_{n.n}\,]_2, [\,B_{n,n-1} \to B_{n,n}\,]_2$, $[\,B'_{n,n-1} \to B'_{n,n}\,]_2, [\,x_{i,j,3n-4} \to x_{i,j,3n-3}\,]_2, [\,\overline{x}_{i,j,3n-4} \to \overline{x}_{i,j,3n-3}\,]_2\,[\,x^*_{i,j,3n-4} \to x^*_{i,j,3n-3}\,]_2, [\,x'_{i,j,3n-4} \to x'_{i,j,3n-3}\,]_2, [\,\overline{x}'_{i,j,3n-4} \to \overline{x}'_{i,j,3n-3}\,]_2\,[\,x'^*_{i,j,3n-4} \to x'^*_{i,j,3n-3}\,]_2$, para $1 \le i \le n, 1 \le j \le p$.

(c) El resultado es válido por (b) observando que $\mathcal{C}_{3n-2}$ se obtiene de $\mathcal{C}_{3n-3}$ por la aplicación de las reglas

$[\,\beta_{3n-3} \to \beta_{3n-2}\,]_1, [\,T_{l,2(n-1)} \to t_{l,2(n-1)}\,]_2, [\,T'_{l,2(n-1)} \to t_{l,2(n-1)}\,]_2, [\,F_{l,2(n-1)} \to f_{l,2(n-1)}\,]_2, [\,F'_{l,2(n-1)} \to f_{l,2(n-1)}\,]_2$, para $1 \le l \le n-1, [\,A_{n,n} \to a_n a'_n\,]_2$, $[\,A'_{n,n} \to a_n a'_n\,]_2, [\,B_{n,n} \to b_n\,]_2, [\,B'_{n,n} \to b_n\,]_2, [\,x_{i,j,3n-3} \to x_{i,j,3n-2}\,]_2$, $[\,\overline{x}_{i,j,3n-3} \to \overline{x}_{i,j,3n-2}\,]_2, [\,x^*_{i,j,3n-3} \to x^*_{i,j,3n-2}\,]_2, [\,x'_{i,j,3n-3} \to x'_{i,j,3n-2}\,]_2$, $[\,\overline{x}'_{i,j,3n-3} \to \overline{x}'_{i,j,3n-2}\,]_2\,[\,x'^*_{i,j,3n-3} \to x'^*_{i,j,3n-2}\,]_2$, para $1 \le i \le n, 1 \le j \le p$.

(d) El resultado es válido por (c) observando que $\mathcal{C}_{3n-1}$ se obtiene de $\mathcal{C}_{3n-2}$ por la aplicación de las reglas

$[\,\beta_{3n-2} \to \beta_{3n-1}\,]_1, [\,t_{l,2(n-1)} \to T_{l,2(n-1)+1}T'_{l,2(n-1)+1}\,]_2$, $[\,f_{l,2(n-1)} \to F_{l,2(n-1)+1}F'_{l,2(n-1)+1}\,]_2$, para $1 \le l \le n-1, [\,a_n \to T_{n,2n-1}\,]_2$,

$[\, a'_n \to F'_{n,2n-1} \,]_2, [\, b_n \to S \,]_2, [\, x_{i,j,3n-2} \to x_{i,j,3n-1} \,]_2, [\, \overline{x}_{i,j,3n-2} \to \overline{x}_{i,j,3n-1} \,]_2,$
$[\, x^*_{i,j,3n-2} \to x^*_{i,j,3n-1} \,]_2, [\, x'_{i,j,3n-2} \to x'_{i,j,3n-1} \,]_2, [\, \overline{x}'_{i,j,3n-2} \to \overline{x}'_{i,j,3n-1} \,]_2,$
$[\, x'^*_{i,j,3n-2} \to x'^*_{i,j,3n-1} \,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

(*e*) El resultado es válido por (d) observando que $\mathcal{C}_{3n}$ se obtiene de $\mathcal{C}_{3n-1}$ por la aplicación de las reglas

$[\, \beta_{3n-1} \to \beta_{3n} \,]_1, [\, S \,]_2 \to [\, \Gamma_0 \,]_2 [\, \Gamma_1 \,]_2, [\, T_{l,2n-1} \to T_{l,2n} \,]_2, [\, T'_{l,2n-1} \to T'_{l,2n} \,]_2,$
$[\, F_{l,2n-1} \to F_{l,2n} \,]_2, [\, F'_{l,2n-1} \to F'_{l,2n} \,]_2$, para $1 \leq l \leq n-1, [\, T_{n,2n-1} \to T_{n-1,2n} \,]_2, [\, F'_{n,2n-1} \to F'_{n,2n} \,]_2, [\, x_{i,j,3n-1} \to x_{i,j,3n} \,]_2, [\, \overline{x}_{i,j,3n-1} \to \overline{x}_{i,j,3n} \,]_2,$
$[\, x^*_{i,j,3n-1} \to x^*_{i,j,3n} \,]_2, [\, x'_{i,j,3n-1} \to x'_{i,j,3n} \,]_2, [\, \overline{x}'_{i,j,3n-1} \to \overline{x}'_{i,j,3n} \,]_2 [\, x'^*_{i,j,3n-1} \to x'^*_{i,j,3n} \,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

(*f*) El resultado es válido por (e) observando que $\mathcal{C}_{3n+1}$ se obtiene de $\mathcal{C}_{3n}$ por la aplicación de las reglas

$[\, \beta_{3n} \to \beta_{3n+1} \,]_1, [\, T_{l,2n} \to T^*_{l,1} \,]_2, [\, T'_{l,2n} \to T^*_{l,1} \,]_2, [\, F_{l,2n} \to F^*_{l,1} \,]_2, [\, F'_{l,2n} \to F^*_{l,1} \,]_2$, para $1 \leq l \leq n-1$ $[\, T_{n,2n} \to T^*_{n,1} \,]_2, [\, F'_{n,2n} \to F^*_{n,1} \,]_2, [\, x_{i,j,3n} \to x_{i,j} \,]_2, [\, \overline{x}_{i,j,3n} \to \overline{x}_{i,j} \,]_2, [\, x^*_{i,j,3n} \to x^*_{i,j} \,]_2, [\, x'_{i,j,3n} \to x'_{i,j} \,]_2, [\, \overline{x}'_{i,j,3n} \to \overline{x}'_{i,j} \,]_2$
$[\, x'^*_{i,j,3n} \to x'^*_{i,j} \,]_2$, para $1 \leq i \leq n, 1 \leq j \leq p$.

□

Primera fase de chequeo

En esta fase, verificaremos qué cláusulas son satisfechas por las valoraciones de verdad codificadas en cada membrana etiquetada por 2. Para ello se disparan las reglas 2.3, que serán aplicadas de tal manera que en el j-ésimo paso de computación ($1 \leq j \leq p$) de esta fase, se verifica la cláusula j y en una membrana etiquetada por 2 se generará un objeto c_j si y sólo si la cláusula C_j es satisfecha por la valoración de verdad codificada en dicha membrana.

Proposición 6.7. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(*a*) *Para cada* k $(1 \leq k \leq p-1)$ *en la configuración* $\mathcal{C}_{3n+1+k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{3n+1+k}(1) = \{\alpha, \beta_{3n+1+k}\}$.
- *Hay* 2^n *membranas etiquetadas por* 2 *tales que cada una de ellas contiene:*

- *el multiconjunto* $cod^*(\varphi)^p_{k+1}$ *correspondiente a las cláusulas* $C_{k+1}, \ldots, C_p$*;*
- *un subconjunto diferente* $\{R^*_{1,k+1}, \ldots, R^*_{n,k+1}\}$*, siendo* $R \in \{T, F\}$ *que codifica una valoración de verdad para las variables* $\{x_1, \ldots, x_n\}$ *de* φ*; y*
- *los objetos* $c_{j,k}$ $(1 \leq j \leq k)$ *si y sólo si la cláusula* C_j *es satisfecha por la valoración de verdad codificada en dicha membrana.*

(*b*) $\mathcal{C}_{3n+1+p}(1) = \{\alpha, \beta_{3n+1+p}\}$*, y en la configuración* $\mathcal{C}_{3n+1+p}(2)$ *hay* 2^n *membranas etiquetadas por* 2 *tales que cada una de ellas contiene objetos* c_j *si y sólo si la cláusula* C_j *es satisfecha por la valoración de verdad codificada en dicha membrana. Además, la multiplicidad del objeto* c_j *representa el número de literales que hacen verdadera la cláusula* C_j*.*

Demostración. (*a*) Por inducción sobre k.

- El caso base $k = 1$ es válido por (f) de la anterior proposición, observando que $\mathcal{C}_{(3n+1)+1}$ se obtiene de $\mathcal{C}_{3n+1}$ por la aplicación de la regla $[\,\beta_{3n+1} \to \beta_{3n+2}\,]_1$ y las reglas $[\,T^*_{i,1}x_{i,1} \to T^*_{i,2}c_{1,1}\,]_2, [\,T^*_{i,1}\overline{x}_{i,1} \to T^*_{i,2}\,]_2, [\,T^*_{i,1}x^*_{i,1} \to T^*_{i,2}\,]_2, [\,F^*_{i,1}x_{i,1} \to F^*_{i,2}\,]_2, [\,F^*_{i,1}\overline{x}_{i,1} \to F^*_{i,2}c_{1,1}\,]_2$, $[\,F^*_{i,1}x^*_{i,1} \to F^*_{i,2}\,]_2$, para $1 \leq i \leq n$

  Supongamos entonces que el resultado es válido para $k, 1 \leq k < p-1$. Para probar que el resultado es válido para $k+1$ es suficiente observar que la configuración $\mathcal{C}_{(3n+1)+k+1}$ se obtiene de la configuración $\mathcal{C}_{(3n+1)+k}$ por la aplicación de la regla $[\,\beta_{(3n+1)+k} \to \beta_{(3n+1)+k+1}\,]_1$, las reglas $[\,c_{j,k} \to c_{j,k+1}\,]_2$, para $1 \leq j \leq k$, y las reglas $[\,T^*_{i,k+1}x_{i,k+1} \to T^*_{i,k+2}c_{k+1,k+1}\,]_2, [\,T^*_{i,k+1}\overline{x}_{i,k+1} \to T^*_{i,k+2}\,]_2, [\,T^*_{i,k+1}x^*_{i,k+1} \to T^*_{i,k+2}\,]_2$, $[\,F^*_{i,k+1}x_{i,k+1} \to F^*_{i,k+2}\,]_2, [\,F^*_{i,k+1}\overline{x}_{i,k+1} \to F^*_{i,k+2}c_{k+1,k+1}\,]_2, [\,F^*_{i,k+1}x^*_{i,k+1} \to F^*_{i,k+2}\,]_2$, para $1 \leq i \leq n$.

(*b*) Por (a) deducimos que la configuración $\mathcal{C}_{(3n+1)+p-1} = \mathcal{C}_{3n+p}$ verifica lo siguiente:

- $\mathcal{C}_{3n+p}(1) = \{\alpha, \beta_{3n+p}\}$.
- Hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
  - el multiconjunto $cod^*(\varphi)^p_p$ correspondiente a la cláusula C_p;

- un subconjunto diferente $\{R^*_{1,p}, \ldots, R^*_{n,p}\}$, siendo $R \in \{T, F\}$ que codifica una valoración de verdad para las variables $\{x_1, \ldots, x_n\}$; y
- los objetos $c_{j,p-1}$ $(1 \leq j \leq p-1)$ si y sólo si la cláusula C_j se satisface por la valoración de verdad codificada en dicha membrana.

Por lo tanto, (b) es válido por la anotación anterior observando que la configuración $\mathcal{C}_{(3n+1)+p}$ se obtiene de la configuración $\mathcal{C}_{3n+p}$ aplicando la regla $[\,\beta_{3n+p} \to \beta_{3n+p+1}\,]_1$, las reglas $[\,c_{j,p-1} \to c_j\,]_2$, para$1 \leq j \leq p-1$, y las reglas $[\,T^*_{i,p}x_{i,p} \to c_p\,]_2, [\,T^*_{i,p}\overline{x}_{i,p} \to \#\,]_2, [\,T^*_{i,p}x^*_{i,p} \to \#\,]_2, [\,F^*_{i,p}x_{i,p} \to \#\,]_2, [\,F^*_{i,p}\overline{x}_{i,p} \to c_p\,]_2, [\,F^*_{i,p}x^*_{i,p} \to \#\,]_2$, para $1 \leq i \leq n$.

□

Segunda fase de chequeo

En esta fase, intentamos determinar si hay alguna valoración de verdad codificada en una de las membranas etiquetadas por 2 satisface todas las cláusulas de φ y, por tanto, la fórmula en sí. Para ello, se aplican las reglas de 2.4 de tal manera que el objeto $d_j, 2 \leq j \leq p$ se produce en el caso que todas las cláusulas $C_1, \ldots, C_j$ sean satisfechas (i.e. todos los objetos $c_1, \ldots, c_j$ aparezcan en una membrana). Por lo tanto, la fórmula φ es satisfecha por una valoración de verdad codificada en una membrana etiquetada por 2 si y sólo si aparece un objeto d_p en dicha membrana. Esta fase tarda exactamente $p-1$ pasos de computación.

Proposición 6.8. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(*a*) *Para cada* i $(1 \leq i \leq p-1)$ *en la configuración* $\mathcal{C}_{3n+p+1+i}$ *tendremos lo siguiente:*

- $\mathcal{C}_{3n+1+p+i}(1) = \{\alpha, \beta_{3n+1+p+i}\}$.
- *Hay* 2^n *membranas etiquetadas por* 2 *tales que cada una de ellas contiene un objeto* d_{i+1} *si y sólo si la valoración de verdad codificada en dicha membrana hace verdaderas las cláusulas* $C_1, \ldots, C_{i+1}$.

(*b*) φ *es satisfactible si y sólo si en la configuración* $\mathcal{C}_{3n+1+p+(p-1)} = \mathcal{C}_{3n+2p}$ *hay alguna membrana etiquetada por 2 que contenga al menos un objeto* d_p.

Demostración. A continuación, se probará esta proposición por inducción.

(*a*) Por inducción sobre i.

- El caso base $i = 1$ es válido por (b) de la proposición anterior, observando que $\mathcal{C}_{(3n+1)+p+1}$ se obtiene de $\mathcal{C}_{(3n+1)+p}$ por la aplicación de las reglas $[\,\beta_{(3n+1)+p} \to \beta_{(3n+1)+p+1}\,]_1$ y $[\,c_1c_2 \to d_2\,]_2$.
- Suponiendo que el resultado es válido para $i, 1 \leq i < p-1$. Para probar que sigue siendo válido para $i+1$ es suficiente observar que la configuración $\mathcal{C}_{(3n+1)+p+(i+1)}$ se obtiene de la configuración $\mathcal{C}_{(3n+1)+p+i}$ por la aplicación de las reglas $[\,\beta_{(3n+1)+p+i} \to \beta_{(3n+1)+p+(i+1)}\,]_1$ y $[\,c_{i+1}c_{i+2} \to d_{i+2}\,]_2$.

(*b*) Supongamos que la fórmula φ es satisfactible si y sólo si hay alguna valoración de verdad σ que haga verdadera φ; es decir, que haga verdaderas todas las cláusulas $C_1, \ldots, C_p$. Por (a) deducimos que φ es satisfactible si y sólo si la configuración $\mathcal{C}_{3n+2p}$ hay alguna membrana etiquetada por 2 que contiene algún objeto d_p.

□

Fase de salida

La fase de salida comienza en el paso $(3n + 2p + 1)$, y tarda exactamente 4 pasos de computación, independientemente de si la fórmula φ es o no satisfactible.

- *Respuesta afirmativa*: si la fórmula $\varphi \in I_{\mathtt{SAT}}$ es satisfactible, entonces al menos una de las valoraciones de verdad codificadas en las membranas etiquetadas por 2 satisfarán todas las cláusulas $C_1, \ldots, C_p$. Por lo tanto, una copia del objeto d_p aparecerá en dicha membrana en la configuración $\mathcal{C}_{3n+2p}$. Entonces, por la aplicación de las reglas $[\,\beta_{3n+2p} \to \beta_{3n+2p+1}\,]_1$ y $[\,d_p\,]_2 \to \gamma\,[\;\;]_2$, los objetos γ y $\beta_{3n+2p+1}$ se producirán en la membrana piel. En el siguiente paso de computación, por la aplicación de las reglas $[\,\alpha\gamma \to \gamma'\,]_1$ y $[\,\beta_{3n+2p+1} \to \beta'\,]_1$, tendremos $\mathcal{C}_{3n+2p+2}(1) = \{\gamma', \beta'\}$. A continuación, aplicando la regla $[\,\gamma' \to \gamma''\,]_1$, obtenemos $\mathcal{C}_{3n+2p+3}(1) = \{\gamma'', \beta'\}$ (obsérves que el objeto β' no pude interactuar con α). Finalmente, en el paso $3n + 2p + 4$ mediante la aplicación de la regla $[\,\gamma''\,]_1 \to \mathtt{yes}\,[\;\;]_1$, el objeto `yes` es enviado al entorno y la computación para.

- *Respuesta negativa*: si la fórmula $\varphi \in I_{\mathtt{SAT}}$ no es satisfactible, entonces ninguna de las valoraciones de verdad codificadas en las membranas etiquetadas por 2 hará verdadera la fórmula φ. Por lo tanto, el objeto d_p no aparece en ninguna membrana con etiqueta 2. Entonces, en el paso $3n + 2p + 1$ sólo la regla $[\,\beta_{3n+2p} \rightarrow \beta_{3n+2p+1}\,]_1$ es aplicable a la configuración $\mathcal{C}_{3n+2p}$. Por lo tanto, $\mathcal{C}_{3n+2p+1}(1) = \{\alpha, \beta_{3n+2p+1}\}$. En el siguiente paso, por la aplicación de la regla $[\,\beta_{3n+2p+1} \rightarrow \beta'\,]_1$ obtenemos $\mathcal{C}_{3n+2p+2}(1) = \{\alpha, \beta'\}$. A continuación, el objeto α puede interactuar con el objeto β' por medio de la regla $[\,\alpha\beta' \rightarrow \beta''\,]_1$ produciendo un objeto β'' en la membrana piel en la configuración $\mathcal{C}_{3n+2p+3}$. Finalmente, en el paso $3n + 2p + 4$, mediante la aplicación de la regla $[\,\beta''\,]_1 \rightarrow \mathtt{no}\,[\quad]_1$ el objeco `no` es enviado al entorno y la computación para.

6.2.1.2. Resultado obtenido

Teorema 6.2. $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{SAM}^0(pmc,+c,-d,-n)}$

Demostración. La familia de sistemas P $\mathbf{\Pi} = \{\Pi(t) \mid t \in \mathbb{N}\}$ presentada verifica lo siguiente:

(a) Todos los sistemas de la familia $\mathbf{\Pi}$ son sistemas P reconocedores de $\mathcal{SAM}^0(pmc, +c, -d, -n)$.

(b) La familia $\mathbf{\Pi}$ es polinomialmente uniforme por máquinas de Turing dado que para cada $n, p \in \mathbb{N}$, las reglas de $\Pi(\langle n, p\rangle)$ de la familia son recursivamente definidos por $n, p \in \mathbb{N}$, y la cantidad de recursos que se necesitan para construir un elemento de dicha familia es de orden polinomial con respecto a n y p, como se indica a continuación.

- Tamaño del alfabeto: $8n^2p+n^2+\frac{p^2}{2}+11np+6n+3p+18 \in \Theta(n^2p)$.
- Número inicial de membranas: $2 \in \Theta(1)$.
- Número inicial de objetos en membranas: $4 \in \Theta(1)$.
- Número de reglas: $15n^2p + 7n^2 - 15np + 18n + 3p + 10 \in \Theta(n^2p)$.
- Número máximo de objetos envueltos en una regla: $4 \in \Theta(1)$.

(c) El par (cod, s) de funciones computables en tiempo polinomial definidas cumplen lo siguiente: para cada fórmula $\varphi \in I_{\mathtt{SAT}}$, $s(\varphi)$ es un número natural, $cod(\varphi)$ es un multiconjunto del sistema $\Pi(s(\varphi))$ y para cada $t \in \mathbb{N}$, $s^{-1}(t)$ es un conjunto finito.

(*d*) La familia $\mathbf{\Pi}$ es polinomialmente acotada: de hecho, para cada fórmula φ de SAT, el sistema P determinista $\Pi(s(\varphi)) + cod(\varphi)$ tarda exactamente $n + 2p + 3$ pasos de computación, siendo n el número de variables y p el número de cláusulas de φ.

(*e*) La familia $\mathbf{\Pi}$ es adecuada con respecto a (X, cod, s): para cada fórmula φ, si la computación de $\Pi(s(\varphi)) + cod(\varphi)$ es una computación de aceptación, entonces φ es satisfactible.

(*f*) La familia $\mathbf{\Pi}$ es completa con respecto a (X, cod, s): para cada fórmula φ que sea satisfactible, la computación de $\Pi(s(\varphi)) + cod(\varphi)$ es una computación de aceptación.

Por lo tanto, la familia $\mathbf{\Pi}$ de sistemas P previamente descrita es una solución uniforme y en tiempo polinomial al problema SAT.

□

Corolario 6.2. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{SAM}^0(pmc,+c,-d,-n)}$

Demostración. Es suficiente observar que el problema SAT es un problema **NP**-completo, $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{SAM}^0(pmc,+c-d,-n)}$ y que la clase de complejidad $\mathbf{PMC}_{\mathcal{SAM}^0(pmc,+c,-d,-n)}$ es cerrada bajo reducibilidad en tiempo polinomial y bajo complementario.

□

6.2.2. Nuevas fronteras obtenidas

Se tienen los siguientes resultados:

- $\mathbf{P} = \mathbf{PMC}_{\mathcal{SAM}^0(bmc,+c,+d,+n)}$
- $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{SAM}^0(pmc,+c,-d,-n)}$
- $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{DAM}^0(bmc,+c,-d,-n)}$

Dados los resultados obtenidos aquí, podemos afirmar que la separación de membranas parece una línea de investigación bastante interesante desde el punto de vista de la complejidad computacional. De hecho, varias nuevas fronteras se han obtenido aquí. Respecto a los propios sistemas P con membranas activas con reglas de separación, pasar del uso de *cooperación minimal acotada* a *cooperación minimal primaria* en reglas de evolución de objetos corresponde a pasar de la no eficiencia a la presumible eficiencia.

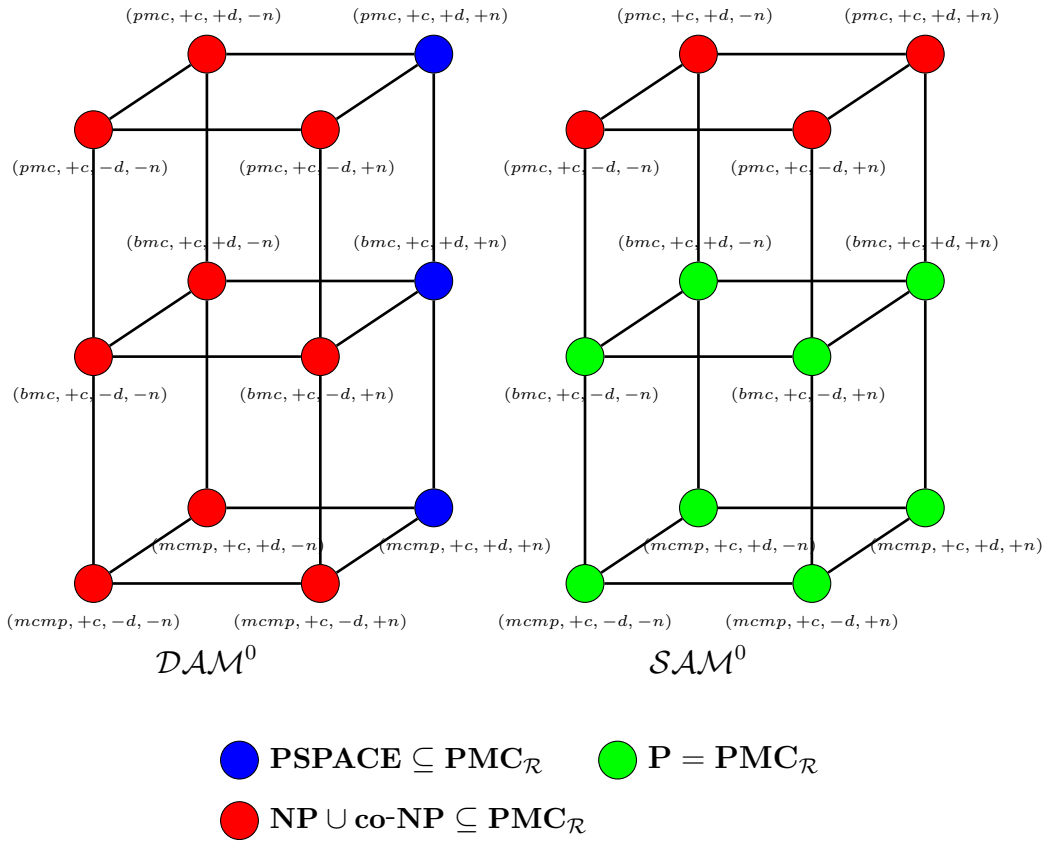


Figura 6.1: Eficiencia computacional en sistemas P con membranas activas sin polarizaciones.

Más aún, otra nueva frontera también es observada. Dado un sistema P con membranas activas y cooperación minimal acotada, el uso de reglas de reglas de separación de membranas limita los problemas resolubles eficientemente a los problemas de la clase **P**, mientras que el uso de reglas de división de membranas permite resolver problemas presuntamente intratables.

En la Figura 6.1 se puede observar el panorama de las clases de complejidad incluidas en los sistemas estudiados en este capítulo.

Capítulo 7

Cooperación minimal en reglas de comunicación

En el transporte de sustancias químicas a través de las membranas celulares, suelen existir elementos químicos que «cooperan» en la realización de esa tarea. En lo que respecta a la comunicación de la célula con el exterior a través de la membrana plasmática, actuando de catalizador o inhibidor del paso de ciertas partículas que se encuentran en las proximidades de la célula en cuestión. En lo que respecta al transporte por el interior de la célula, facilitando el movimiento de las sustancias a través de los diversos orgánulos contenidos en la misma.

El objetivo del presente capítulo es el estudio y obtención de fronteras de la tratabilidad de problemas en el marco de los sistemas P con membranas activas, sin polarizaciones y sin reglas de disolución cuando se permite el uso de reglas de comunicación que sean cooperativas.

7.1. $\mathcal{DAM}^0(-d)$: Cooperación en reglas de comunicación

En esta sección definiremos los citados sistemas P con membranas activas sin polarización con cooperación minimal en reglas de comunicación y división de membranas.

Definición 7.1. *Un sistema P con membranas activas sin polarización con cooperación minimal en reglas de comunicación y división de grado* $q \geq 1$ *es una tupla* $\Pi = (\Gamma, H, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$*, donde* Π *es un sistema P con*

membranas activas sin polarización con división y las reglas de evolución de objetos se definen como sigue:

(b') $u[\ \ ]_h \to [\,c\,]_h$, *para* $h \in H, 1 \leq |u| \leq 2, u \in \Gamma^+, c \in \Gamma$

(c') $[\,u\,]_h \to c[\ \ ]_h$, *para* $h \in H, 1 \leq |u| \leq 2, u \in \Gamma^+, c \in \Gamma$

La semántica de estos sistemas, así como la definición de *configuración* y *computación* son similares a los definidos anteriormente, con la peculiaridad de la cooperación en las reglas de comunicación.

Dependiendo de las reglas de comunicación en las que estén permitidas la cooperación minimal, se utilizarán las siguiente notaciones

- *Cooperación minimal en reglas de comunicación hacia dentro* ($\mathbf{mcmp_{in}}$): mientras que las reglas de comunicación hacia dentro pueden ser cooperativas, las reglas de cooperación hacia fuera están restringidas a las clásicas de los sistemas P con membranas activas sin polarizaciones.

- *Cooperación minimal en reglas de comunicación hacia fuera* ($\mathbf{mcmp_{out}}$): mientras que las reglas de comunicación hacia fuera pueden ser cooperativas, las reglas de cooperación hacia dentro están restringidas a las clásicas de los sistemas P con membranas activas sin polarizaciones.

- *Cooperación minimal en reglas de comunicación hacia dentro y hacia fuera* ($\mathbf{mcmp_{in\text{-}out}}$): tanto las reglas de comunicación hacia dentro como las reglas de comunicación hacia fuera pueden ser cooperativas.

Denotaremos entonces por $\mathcal{DAM}^0(\alpha, \beta, \delta, \gamma)$ la clase de sistemas P con membranas activas y con reglas de división, donde:

- $\alpha = +e$ (respectivamente, $+e_s$) cuando se permita el uso de *reglas de evolución* (respectivamente, *reglas de evolución simple*, del tipo $[\,a \to b\,]_h$, $a, b \in \Gamma$).

- $\beta = mcmp_{in-out}$ (respectivamente, $mcmp_{in}$, $mcmp_{out}$) cuando se permita la cooperación tanto en reglas de comunicación hacia dentro como en reglas de comunicación hacia fuera (respectivamente, sólo en reglas de comunicación hacia dentro, sólo en reglas de comunicación hacia fuera).

- $\delta = \pm d$ cuando se permita (respectivamente, se prohíba) el uso de reglas de disolución.

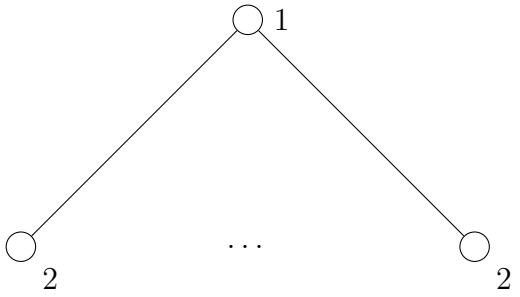


Figura 7.1: Estructura de membranas final en la solución de [65]

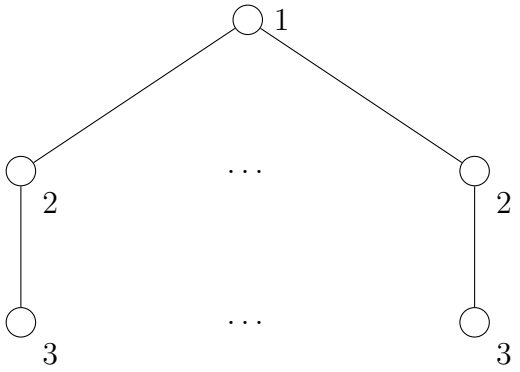


Figura 7.2: Estructura de membranas final en la solución de [64]

- $\gamma = \pm n$ cuando se permita el uso de reglas de división sólo para membranas elementales (respectivamente, para membranas elementales y no elementales).

Para obtener una solución similar a la obtenida en [65], en la que se usaban *cooperación minimal y producción minimal* en reglas de evolución, en este marco, en el que la cooperación se realiza en las reglas de comunicación, necesitamos tener una especie de membrana auxiliar; es decir, donde en la solución anterior había una sola membrana, en este caso tendremos dos que harán las veces de «membrana evolutiva». La diferencia entre estas dos estructuras puede verse reflejada entre las Figuras 7.1 y 7.2. De hecho, se restringirán las reglas de evolución de objetos a $[\, a \to b \,]_h, a, b \in \Gamma$. A continuación, pasaremos a presentar dicha solución.

7.1.1. Una solución eficiente al problema SAT mediante sistemas de $\mathcal{DAM}^0(+e_s, mcmp_{in}, -d, +n)$

A continuación, se describe una familia de sistemas de membranas de $\mathcal{DAM}^0(+e_s, mcmp_{in}, -d, +n)$ que resuelve el problema SAT en tiempo polinomial y de manera uniforme.

Para cada $n, p \in \mathbb{N}$, consideramos el sistema P reconocedor

$$\Pi(\langle n, p\rangle) = (\Gamma, \Sigma, H, \mu, \mathcal{M}_0, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, i_{in}, i_{out})$$

de $\mathcal{DAM}^0(+e_s, mcmp_{in}, -d, +n)$, definida como sigue:

1. Alfabeto de trabajo Γ:

   $\{\mathtt{yes}, \mathtt{no}, \#\} \cup \{a_{i,k} \mid 1 \leq i \leq n, 1 \leq k \leq 2i-1\} \cup$
   $\{\alpha_k \mid 0 \leq k \leq 4np + 2n + 2p + 1\} \cup \{\beta_k \mid 0 \leq k \leq 4np + 2n + 2p + 2\} \cup$
   $\{\gamma_k \mid 0 \leq k \leq 4np + 2n\} \cup$
   $\{t_{i,k}, f_{i,k} \mid 1 \leq i \leq n, 2i-1 \leq k \leq 2n + 2p - 1\} \cup \{T_i, F_i \mid 1 \leq i \leq n\} \cup$
   $\{c_j \mid 1 \leq j \leq p\} \cup \{c_{j,k} \mid 1 \leq j \leq p, 0 \leq k \leq np - 1\} \cup$
   $\{d_j \mid 1 \leq j \leq p\} \cup \{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 1 \leq i \leq n1 \leq j \leq p,$
   $1 \leq k \leq 2n + 2np + n(j-1) + (i-1)\}$

2. Alfabeto de entrada Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,j,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.

3. $H = \{0, 1, 2\}$;

4. $\mu = [\,[\,[\;\;]_2\,]_1\,]_0$; es decir, $\mu = (V, E)$ donde $V = \{0, 1, 2\}$ y $E = \{(0, 1), (1, 2)\}$.

5. $\mathcal{M}_0 = \{\alpha_0, \beta_0\}$, $\mathcal{M}_1 = \{\gamma_0\} \cup \{T^p_i, F^p_i \mid 1 \leq i \leq n\}$, $\mathcal{M}_2 = \{a_{i,1} \mid 1 \leq i \leq n\}$;

6. El conjunto $\mathcal{R}$ tiene las siguientes reglas:

   1.1 Contadores para sincronizar la salida del sistema
   $[\,\alpha_k \to \alpha_{k+1}\,]_0$, para $0 \leq k \leq 4np + 2n + 2p$
   $[\,\beta_k \to \beta_{k+1}\,]_0$, para $0 \leq k \leq 4np + 2n + 2p + 1$
   $[\,\gamma_k \to \gamma_{k+1}\,]_1$, para $0 \leq k \leq 4np + 2n - 1$

   1.2 Reglas para generar 2^n membranas etiquetadas por 1 y 2^n membranas etiquetadas por 2 (estas codificando todas las valoraciones de verdad posibles para n variables de la fórmula φ

$[\, a_{i,2i-1} \,]_2 \to [\, t_{i,i} \,]_2 [\ f_{i,i} \,]_2$, para $1 \leq i \leq n$
$[\, a_{i,j} \to a_{i,j+1} \,]_2$, para $2 \leq i \leq n, 1 \leq j \leq 2i-2$
$[\, [\,]_2 [\,]_2 \,]_1 \to [\, [\,]_2 \,]_1 [\, [\,]_2 \,]_1$
$\left.\begin{array}{l} [\, t_{i,j} \to t_{i,j+1} \,]_2 \\ [\, f_{i,j} \to f_{i,j+1} \,]_2 \end{array}\right\}$, para $1 \leq i \leq n, i \leq j \leq 2n-1$

1.3 Reglas para producir exactamente p copias de cada valoración de verdad dentro de las membranas etiquetada por 2.

$\left.\begin{array}{l} [\, t_{i,2jn} \,]_2 \to t_{i,2jn+1} [\ \ ]_2 \\ [\, f_{i,2jn} \,]_2 \to f_{i,2jn+1} [\ \ ]_2 \end{array}\right\}$, para$1 \leq i \leq n$, $1 \leq j \leq p$

$\left.\begin{array}{l} [\, t_{i,2jn+k} \to t_{i,2jn+k+1} \,]_1 \\ [\, f_{i,2jn+k} \to f_{i,2jn+k+1} \,]_1 \end{array}\right\}$, para $\begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 1 \leq k \leq n-1 \end{array}$

$\left.\begin{array}{l} t_{i,(2j+1)n} F_i [\ \ ]_2 \to [\, t_{i,(2j+1)n+1} \,]_2 \\ f_{i,(2j+1)n} T_i [\ \ ]_2 \to [\, f_{i,(2j+1)n+1} \,]_2 \end{array}\right\}$, para $\begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p-1 \end{array}$

$\left.\begin{array}{l} [\, t_{i,(2j+1)n+k} \to t_{i,(2j+1)n+k+1} \,]_2 \\ [\, f_{i,(2j+1)n+k} \to f_{i,(2j+1)n+k+1} \,]_2 \end{array}\right\}$, para $\begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$

$\left.\begin{array}{l} t_{i,2np+n} F_i [\ \ ]_2 \to [\, \# \,]_2 \\ f_{i,2np+n} T_i [\ \ ]_2 \to [\, \# \,]_2 \end{array}\right\}$, para$1 \leq i \leq n$

1.4 Reglas para preparar la fórmula para validar las cláusulas.

$\left.\begin{array}{l} [\, x_{i,j,k} \to x_{i,j,k+1} \,]_1 \\ [\, \overline{x}_{i,j,k} \to \overline{x}_{i,j,k+1} \,]_1 \\ [\, x^*_{i,j,k} \to x^*_{i,j,k+1} \,]_1 \end{array}\right\}$, para $\begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 0 \leq k \leq 2np + 2n \\ +n(j-1) + (i-1) - 1 \end{array}$

2.1 Reglas para realizar el primer chequeo.

$\left.\begin{array}{l} T_i x_{i,j,2np+2n+n(j-1)+(i-1)} [\ \ ]_2 \to [\, c_{j,0} \,]_2 \\ T_i \overline{x}_{i,j,2np+2n+n(j-1)+(i-1)} [\ \ ]_2 \to [\, \# \,]_2 \\ T_i x^*_{i,j,2np+2n+n(j-1)+(i-1)} [\ \ ]_2 \to [\, \# \,]_2 \\ F_i x_{i,j,2np+2n+n(j-1)+(i-1)} [\ \ ]_2 \to [\, \# \,]_2 \\ F_i \overline{x}_{i,j,2np+2n+n(j-1)+(i-1)} [\ \ ]_2 \to [\, c_{j,0} \,]_2 \\ F_i x^*_{i,j,2np+2n+n(j-1)+(i-1)} [\ \ ]_2 \to [\, \# \,]_2 \end{array}\right\}$, para $\begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$

3.1 Reglas para realizar el segundo chequeo.

$[\, c_{j,k} \to c_{j,k+1} \,]_2$, para$1 \leq j \leq p$, $0 \leq k \leq np-2$
$[\, c_{j,np-1} \,]_2 \to c_j [\ \ ]_2$, para$1 \leq j \leq p$
$\gamma_{4np+2n} c_1 [\ \ ]_1 \to [\, d_1 b \,]_1$
$[\, d_j \,]_2 \to d_j [\ \ ]_2$, para$1 \leq j \leq p$
$d_j c_{j+1} [\ \ ]_2 \to [\, d_{j+1} \,]_2$, para$1 \leq j \leq p-1$

4.1 Reglas para devolver la respuesta correcta.
$[\, d_p \,]_1 \to d_p [\quad]_1$
$\alpha_{4np+2n+2p+1} d_p [\quad]_1 \to [\, \texttt{yes} \,]_1$
$\alpha_{4np+2n+2p+1} \beta_{4np+2n+2p+2} [\quad]_1 \to [\, \texttt{no} \,]_1$
$[\, \texttt{yes} \,]_1 \to \texttt{yes} [\quad]_1$
$[\, \texttt{no} \,]_1 \to \texttt{no} [\quad]_1$
$[\, \texttt{yes} \,]_0 \to \texttt{yes} [\quad]_0$
$[\, \texttt{no} \,]_0 \to \texttt{no} [\quad]_0$

7. La membrana de entrada es la membrana etiquetada por 1 ($i_{in} = 1$) y la región de salida es el entorno ($i_{out} = env$).

Sea $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_p, C_j = l_{j_1} \vee \cdots \vee l_{j_{r_j}}, 1 \leq j \leq p, l_{j_k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$ una instancia del problema SAT. Entonces la instancia φ será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$, en donde $s(\varphi) = \langle n, p \rangle$ y $cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$.

Denotamos por $cod_k(\varphi)$ el conjunto $cod(\varphi)$ cuando los elementos tienen su tercer subíndice igual a k. Denotamos por $cod_k(\varphi)_{k_1}^{k_2}$ al los elementos correspondientes a las cláusulas $C_{k_1}, \ldots, C_{k_2}$ de $cod_k(\varphi)$. De la misma forma, denotamos por $cod'_k(\varphi)_{k_1}^{k_2}$ a los elementos de $cod_k(\varphi)_{k_1}^{k_2}$ con una prima. Por último, denotamos con $cod^*(\varphi)_{k_1}^{k_2}$ a los elementos de $cod(\varphi)_{k_1}^{k_2}$ cuando eliminamos el tercer subíndice.

A continuación, daremos una descripción informal del funcionamiento del sistema $\Pi(s(\varphi)) + cod(\varphi)$.

La solución propuesta sigue un algoritmo de fuerza bruta en el marco de sistemas P reconocedores de membranas activas, con cooperación minimal en reglas de comunicación hacia dentro y reglas de división sólo para membranas elementales, y consiste en las siguientes fases:

- *Fase de generación*: Usando las reglas de división, obtenemos todas las valoraciones de verdad posibles para las variables $\{x_1, \ldots, x_n\}$ asociadas a φ. En concreto, 2^n membranas etiquetadas por 2 y 2^n labelled by 1 are generated. Cada una de las últimas contiene una posible valoración de verdad. Esta fase tarda exactamente $2n + 2np$ pasos de computación, siendo n el número de variables de φ.

- *Primera fase de chequeo*: En esta fase de verifica cuáles son las cláusulas de φ que satisface y cuáles no cada una de las valoraciones de verdad

codificadas en cada membrana etiquetada por 1. Esta fase tarda np pasos, siendo n el número de variables y p el número de cláusulas de φ.

- *Segunda fase de chequeo*: En esta fase se verificará si alguna de las valoraciones de verdad satisface todas las cláusulas de φ, usando las reglas de 3.1. Esta fase tarda exactamente $np + 2p$ pasos de computación.

- *Fase de salida*: El sistema envía al entorno la respuesta adecuada usando las reglas de 4.1, y tarda 4 pasos de computación si la respuesta es afirmativa y 5 pasos en caso que la respuesta sea negativa.

7.1.1.1. Verificación formal

A continuación, se detallará una verificación exhaustiva de la solución.

Fase de generación

Mediante esta fase, todas las valoraciones de verdad para las variables asociadas a la fórmula Booleana φ serán generadas en las membranas etiquetadas por 1, por la aplicación de las reglas de 1.2 y 1.3. En los primeros $2n$ pasos, se generarán 2^n membranas etiquetadas por 2 y 2^n membranas etiquetadas por 1, alternando entre la división de membranas etiquetadas por 2 (en los pasos impares) y la división de membranas etiquetadas por 1 (en los pasos pares).

Proposición 7.1. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a_0) *Para cada* $2k$ *(*$0 \leq k \leq n-1$*) en la configuración* $\mathcal{C}_{2k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2k}(0) = \{\alpha_{2k}, \beta_{2k}\}$
- *Hay* 2^k *membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada* $cod_{2k}(\varphi)$*;*
  - *un objeto* γ_{2k}*; y*
  - p *copias de todos los* T_i *y* F_i*,* $1 \leq i \leq n$.
- *Hay* 2^k *membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *los objetos* $a_{i,2k+1}$*,* $k+1 \leq i \leq n$*; y*

- *un subconjunto diferente* $\{r_{1,j}, \ldots, r_{k,j}\}$, $k+1 \leq j \leq 2k$, *siendo* $r \in \{t, f\}$.

(a_1) *Para cada* $2k+1$ *(*$0 \leq j \leq n-1$*) en la configuración* $\mathcal{C}_{2k+1}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2k+1}(0) = \{\alpha_{2k+1}, \beta_{2k+1}\}$
- *Hay* 2^k *membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada* $cod_{2k+1}(\varphi)$*;*
  - *un objeto* γ_{2k+1}*; y*
  - p *copias de todos los* T_i *y* F_i, $1 \leq i \leq n$.
- *Hay* 2^{k+1} *membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *los objetos* $a_{i,2(k+1)}$, $k+1 \leq i \leq n$*; y*
  - *un subconjunto diferente* $\{r_{1,j}, \ldots, r_{k+1,j}\}$, $k+1 \leq j \leq 2k+1$, *siendo* $r \in \{t, f\}$.

(b) $\mathcal{C}_{2n}(0) = \{\alpha_{2n}, \beta_{2n}\}$*, y en la configuración* $\mathcal{C}_{2n}$ *hay* 2^n *membranas etiquetadas por 1, tales que cada una de ellas contiene el multiconjunto de entrada* $cod_{2n}(\varphi)$*,* p *copias de cada objeto* T_i *y* F_i *(*$1 \leq i \leq n$*) y un objeto* γ_{2n}*; y* 2^n *membranas etiquetadas por 2, tales que cada una de ellas contiene un subconjunto diferente de objetos* $r_{i,2n+1-i}$, $1 \leq i \leq n$.

Demostración. (a) será demostrado por inducción sobre k

- El caso base $k = 0$ es trivial dado que:

  (a_0) en la configuración inicial $\mathcal{C}_0$ tenemos que $\mathcal{C}_0(0) = \{\alpha_0, \beta_0\}$ y hay una sola membrana etiquetada por 1 que contiene el multiconjunto de entrada $cod(\varphi)$, un objeto γ_0 y p copias de T_i y F_i, siendo $1 \leq i \leq n$; y una sola membrana etiquetada por 2 que contiene los objetos $a_{1,1}, \ldots, a_{n,1}$. Por tanto, la configuración $\mathcal{C}_0$ da lugar a la configuración $\mathcal{C}_1$ por la aplicación de las reglas:
  $[\, a_{1,1} \,]_2 \to [\, t_{1,1} \,]_2 \, [\, f_{1,1} \,]_2$
  $[\, a_{i,1} \to a_{i,2} \,]_2$, para $k+1 \leq i \leq n$
  $[\, \alpha_0 \to \alpha_1 \,]_0$
  $[\, \beta_0 \to \beta_1 \,]_0$
  $[\, \gamma_0 \to \gamma_1 \,]_1$

$$\left.\begin{array}{l}[\, x_{i,j,0} \to x_{i,j,1}\,]_1\\ [\, \overline{x}_{i,j,0} \to \overline{x}_{i,j,1}\,]_1\\ [\, x^*_{i,j,0} \to x^*_{i,j,1}\,]_1\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

(a_1) en la configuración $\mathcal{C}_1$ tenemos que $\mathcal{C}_1(0) = \{\alpha_1, \beta_1\}$ y que hay una sola membrana etiquetada por 1 que contiene el multiconjunto de entrada $cod_1(\varphi)$, un objeto γ_1 y p copias de los objetos T_i y F_i, siendo $1 \leq i \leq n$; y dos membranas etiquetadas por 2 que contienen los objetos $a_{2,2}, \ldots, a_{n,2}$, una de ellas con un objeto $t_{1,1}$ y la otra con un objeto $f_{1,1}$. Por tanto, la configuración $\mathcal{C}_1$ da lugar a la configuración $\mathcal{C}_2$ por la aplicación de las reglas:

$[\, t_{1,1} \to t_{1,2}\,]_2$
$[\, f_{1,1} \to f_{1,2}\,]_2$
$[\, [\,]_2 [\,]_2\,]_1 \to [\, [\,]_2\,]_1\, [\, [\,]_2\,]_1$
$[\, a_{i,2} \to a_{i,3}\,]_2$, para $2 \leq i \leq n$
$[\, \alpha_1 \to \alpha_2\,]_0$
$[\, \beta_1 \to \beta_2\,]_0$
$[\, \gamma_1 \to \gamma_2\,]_1$

$$\left.\begin{array}{l}[\, x_{i,j,1} \to x_{i,j,2}\,]_1\\ [\, \overline{x}_{i,j,1} \to \overline{x}_{i,j,2}\,]_1\\ [\, x^*_{i,j,1} \to x^*_{i,j,2}\,]_1\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, $\mathcal{C}_2(0) = \{\alpha_2, \beta_2\}$, y hay dos membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_2(\varphi)$, un objeto γ_2 y p copias de los objetos T_i y F_i, siendo $1 \leq i \leq n$; y dos membranas etiquetadas por 2 que contienen los objetos $a_{2,3}, \ldots, a_{n,3}$, una de ellas con un objeto $t_{1,2}$ y la otra con un objeto $f_{1,2}$. Por lo tanto, el resultado es válido para $k = 1$.

- Suponiendo, por inducción, que el resultado es cierto para k ($0 \leq k \leq n-1$)
  - $\mathcal{C}_{2k}(0) = \{\alpha_{2k}, \beta_{2k}\}$
  - En la configuración $\mathcal{C}_{2k}$ hay 2^k membranas etiquetadas por 1 tales que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2k}(\varphi)$;
    - un objeto γ_{2k}; y
    - p copias de los objetos T_i y F_i, $1 \leq i \leq n$.
  - En la configuración $\mathcal{C}_{2k}$ hay 2^k membranas etiquetadas por 2 tales que cada una de ellas contiene

- los objetos $a_{i,2k+1}$, $k+1 \leq i \leq n$; y
- un subconjunto diferente $\{r_{1,j}, \ldots, r_{k,j}\}$, $k+1 \leq j \leq 2k$, siendo $r \in \{t, f\}$.

Por tanto, la configuración $\mathcal{C}_{2k}$ da lugar a la configuración $\mathcal{C}_{2k+1}$ por la aplicación de las reglas:

$[\, a_{k,2k+1} \,]_2 \to [\, t_{k,k} \,]_2 \, [\, f_{k,k} \,]_2$

$[\, a_{i,2k+1} \to a_{i,2k+2} \,]_2$, para $k+1 \leq i \leq n$

$\left.\begin{array}{l} [\, t_{i,j} \to t_{i,j+1} \,]_2 \\ [\, f_{i,j} \to f_{i,j+1} \,]_2 \end{array}\right\}$ para $1 \leq i \leq k-1, k+1 \leq j \leq 2k$

$[\, \alpha_{2k} \to \alpha_{2k+1} \,]_0$

$[\, \beta_{2k} \to \beta_{2k+1} \,]_0$

$[\, \gamma_{2k} \to \gamma_{2k+1} \,]_1$

$\left.\begin{array}{l} [\, x_{i,j,2k} \to x_{i,j,2k+1} \,]_1 \\ [\, \overline{x}_{i,j,1} \to \overline{x}_{i,j,2k+1} \,]_1 \\ [\, x^*_{i,j,1} \to x^*_{i,j,2k+1} \,]_1 \end{array}\right\}$ para $1 \leq i \leq n, 1 \leq j \leq p$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2k+1}(0) = \{\alpha_{2k+1}, \beta_{2k+1}\}$
- En la configuración $\mathcal{C}_{2k+1}$ hay 2^k membranas etiquetadas por 1 tales que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{2k+1}(\varphi)$;
  - un objeto γ_{2k+1}; y
  - p copias de los objetos T_i y F_i, $1 \leq i \leq n$.
- En la configuración $\mathcal{C}_{2k+1}$ hay 2^{k+1} membranas etiquetadas por 2 tal que cada una de ellas contiene
  - los objetos $a_{i,2(k+1)}$, $k+1 \leq i \leq n$; y
  - un subconjunto diferente $\{r_{1,j}, \ldots, r_{k+1,j}\}$, $k+1 \leq j \leq 2k+1$, siendo $r \in \{t, f\}$.

  Por tanto, la configuración $\mathcal{C}_{2k+1}$ da lugar a la configuración $\mathcal{C}_{2(k+1)}$ por la aplicación de las reglas:

  $\left.\begin{array}{l} [\, t_{i,j} \to t_{i,j+1} \,]_2 \\ [\, f_{i,j} \to f_{i,j+1} \,]_2 \end{array}\right\}$ para $1 \leq i \leq k+1, k+1 \leq j \leq 2k+1$

  $[\, [\,]_2 [\,]_2 \,]_1 \to [\, [\,]_2 \,]_1 \, [\, [\,]_2 \,]_1$

  $[\, a_{i,2(k+1)} \to a_{i,2(k+1)+1} \,]_2$, para $k+1 \leq i \leq n$

  $[\, \alpha_{2k+1} \to \alpha_{2(k+1)} \,]_0$

  $[\, \beta_{2k+1} \to \beta_{2(k+1)} \,]_0$

  $[\, \gamma_{2k+1} \to \gamma_{2(k+1)} \,]_1$

$$\left.\begin{array}{l}[\, x_{i,j,2k+1} \to x_{i,j,2k+2} \,]_1 \\ [\, \overline{x}_{i,j,2k+1} \to \overline{x}_{i,j,2k+2} \,]_1 \\ [\, x^*_{i,j,2k+1} \to x^*_{i,j,2k+2} \,]_1 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2(k+1)}(0) = \{\alpha_{2(k+1)}, \beta_{2(k+1)}\}$
- En la configuración $\mathcal{C}_{2(k+1)}$ hay 2^{k+1} membranas etiquetadas por 1 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{2(k+1)}(\varphi)$;
  - un objeto $\gamma_{2(k+1)}$; y
  - p copias de los objetos T_i y F_i, $1 \leq i \leq n$.
- En la configuración $\mathcal{C}_{2(k+1)}$ hay 2^{k+1} membranas etiquetadas por 2 tal que cada una de ellas contiene
  - los objetos $a_{i,2(k+1)+1}$, $k+1 \leq i \leq n$; y
  - un subconjunto diferente $\{r_{1,j}, \ldots, r_{k+1,j}\}$, $k+1 \leq j \leq 2(k+1)+1$.

  Por lo tanto, el resultado es válido para $k+1$.

- Para probar (b) basta darse cuenta que, por una parte, por (a) la configuración $\mathcal{C}_{2n-1}$ es válido:
  - $\mathcal{C}_{2n-1}(0) = \{\alpha_{2n-1}, \beta_{2n-1}\}$
  - En la configuración $\mathcal{C}_{2n-1}$ hay 2^{n-1} membranas etiquetadas por 1 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2n-1}p(\varphi)$;
    - un objeto γ_{2n-1}; y
    - p copias de los objetos T_i y F_i, $1 \leq i \leq n$.
  - En la configuración $\mathcal{C}_{2n-1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene un subconjunto diferente of objects $r_{i,2n-i}$, $1 \leq i \leq n$.

    Por tanto, la configuración $\mathcal{C}_{2n-1}$ da lugar a la configuración $\mathcal{C}_{2n}$ por la aplicación de las reglas:

    $$\left.\begin{array}{l}[\, t_{i,2n-i} \to t_{i,2n+1-i} \,]_2 \\ [\, f_{i,2n-i} \to f_{i,2n+1-1} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n$$

    $$[\,[\,]_2[\,]_2\,]_1 \to [\,[\,]_2\,]_1\,[\,[\,]_2\,]_1$$
    $$[\, \alpha_{2n-1} \to \alpha_{2n} \,]_0$$
    $$[\, \beta_{2n-1} \to \beta_{2n} \,]_0$$
    $$[\, \gamma_{2n-1} \to \gamma_{2n} \,]_1$$

$$\left.\begin{array}{l}[\, x_{i,j,2n-1} \to x_{i,j,2n} \,]_1 \\ [\, \overline{x}_{i,j,2n-1} \to \overline{x}_{i,j,2n} \,]_1 \\ [\, x^*_{i,j,2n-1} \to x^*_{i,j,2n} \,]_1 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por tanto, tenemos que $\mathcal{C}_{2n}(0) = \{\alpha_{2n}, \beta_{2n}\}$, y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{2n}(\varphi)$, un objeto γ_{2n} y p copias de los objetos T_i y F_i, siendo $1 \leq i \leq n$; y 2^n membranas etiquetadas por 2 que contienen un subconjunto diferente $r_{i,2n+1-i}$, siendo $1 \leq i \leq n$.

□

Cuando la estructura de árbol se crea, empezamos a asignar una valoración de verdad a cada rama. Esto se ejecuta en los siguientes $2np$ pasos. Los últimos n pasos son diferentes de los pasos previos, así que se expondrán en una proposición diferente.

Proposición 7.2. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a_0) *Para cada* k $(1 \leq k \leq n)$ *y* l $(0 \leq l \leq p-1)$ *en la configuración* $\mathcal{C}_{2n+2ln+k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2n+2ln+k}(0) = \{\alpha_{2n+2ln+k}, \beta_{2n+2ln+k}\}$
- *Hay* 2^n *membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada* $cod_{2n+2ln+k}(\varphi)$*;*
  - *un objeto* $\gamma_{2n+2ln+k}$*;*
  - p *copias de cada objeto* T_i *y* F_i*,* $1 \leq i \leq n$ *si la valoración de verdad asociada a la rama contiene su objeto* t_i *o* f_i *correspondiente, y* $p - l$ *copias de lo contrario; y*
  - *los objetos* $r_{i,2n+2ln+k-i+1}$*,* $1 \leq i \leq k$*, siendo* $r \in \{t, f\}$.
- *Hay* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene un subconjunto diferente de objetos* $r_{i,2n+2ln+k-i+1}$*,* $k+1 \leq i \leq n$*, siendo* $r \in \{t, f\}$.

(a_1) *Para cada* k $(1 \leq k \leq n)$ *y* l $(0 \leq l \leq p-2)$ *en la configuración* $\mathcal{C}_{3n+2ln+k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{3n+2ln+k}(0) = \{\alpha_{3n+2ln+k}, \beta_{3n+2ln+k}\}$

- *Hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada $cod_{3n+2ln+k}(\varphi)$;*
  - *un objeto $\gamma_{3n+2ln+k}$;*
  - *p copias de cada objeto T_i y F_i, $1 \leq i \leq n$ si la valoración de verdad asociada a la rama contiene su objeto correspondiente t_i o f_i; de lo contrario, hay $p-l$ objetos si $k+1 \leq i \leq n$, $p-l-1$ en caso contrario; y*
  - *los objetos $r_{i,3n+2ln+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.*
- *Hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,3n+2ln+k-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$.*

(*b*) *$\mathcal{C}_{n+2np}(0) = \{\alpha_{n+2np}, \beta_{n+2np}\}$, y en la configuración $\mathcal{C}_{n+2np}$ hay 2^n membranas etiquetadas por 1, tales que cada una de ellas contiene el multiconjunto de entrada $cod_{n+2np}(\varphi)$, un objeto γ_{n+2np}, p copias de cada objeto T_i y F_i, $1 \leq i \leq n$ si la valoración de verdad asociada a la rama contiene su correspondiente objeto t_i o f_i, y 1 objeto de lo contrario, y objetos $r_{i,n+2np-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad asociada a la rama; y 2^n membranas vacías etiquetadas por 2.*

Demostración. (*a*) será demostrado por inducción sobre l

- El caso base $l = 0$ será demostrado por inducción sobre k

  (a_0) El caso base $k = 1$ es trivial dado que:

  - en la configuración $\mathcal{C}_{2n}$ tenemos que: $\mathcal{C}_{2n}(0) = \{\alpha_{2n}, \beta_{2n}\}$ y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{2n}(\varphi)$, un objeto γ_{2n} y p copias de T_i y F_i, siendo $1 \leq i \leq n$; y 2^n membranas etiquetadas por 2 que contienen un subconjunto diferente de objetos $r_{i,2n-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$, la valoración de verdad correspondiente a la rama. Por tanto, la configuración $\mathcal{C}_{2n}$ da lugar a la configuración $\mathcal{C}_{2n+1}$ por la aplicación de las reglas:

$$\left.\begin{array}{l} [\, t_{i,2n}\,]_2 \to t_{i,2n+1}\,[\quad]_2 \\ [\, f_{i,2n}\,]_2 \to f_{i,2n+1}\,[\quad]_2 \\ [\, t_{i,2n+1-i} \to t_{i,2n+2-i}\,]_2 \\ [\, f_{i,2n+1-i} \to f_{i,2n+2-1}\,]_2 \end{array}\right\} \text{para } 2 \leq i \leq n$$

$[\,\alpha_{2n} \to \alpha_{2n+1}\,]_0$
$[\,\beta_{2n} \to \beta_{2n+1}\,]_0$
$[\,\gamma_{2n} \to \gamma_{2n+1}\,]_1$
$\left.\begin{array}{l}[\,x_{i,j,2n} \to x_{i,j,2n+1}\,]_1\\ [\,\overline{x}_{i,j,2n} \to \overline{x}_{i,j,2n+1}\,]_1\\ [\,x^*_{i,j,2n} \to x^*_{i,j,2n+1}\,]_1\end{array}\right\}$ para $1 \leq i \leq n, 1 \leq j \leq p$

Por lo tanto, $\mathcal{C}_{2n+1}(0) = \{\alpha_{2n+1}, \beta_{2n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{2n+1}(\varphi)$, un objeto γ_{2n+1}, p copias de T_i y F_i, siendo $1 \leq i \leq n$ y un objeto $r_{1,2n+1}$, siendo $r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen un subconjunto diferente de objetos $r_{i,2n-i+2}$, $2 \leq i \leq n$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{2n+k}(0) = \{\alpha_{2n+k}, \beta_{2n+k}\}$
  - En la configuración $\mathcal{C}_{2n+k}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2n+k}(\varphi)$;
    - un objeto γ_{2n+k};
    - p copias de cada objeto T_i y F_i, $1 \leq i \leq n$; y
    - los objetos $r_{i,2n+k-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{2n+k}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,2n+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.

    Por tanto, la configuración $\mathcal{C}_{2n+k}$ da lugar a la configuración $\mathcal{C}_{2n+k+1}$ por la aplicación de las reglas:

    $[\,t_{k+1,2n}\,]_2 \to t_{k+1,2n+1}\,[\quad]_2$
    $[\,f_{k+1,2n}\,]_2 \to f_{k+1,2n+1}\,[\quad]_2$
    $\left.\begin{array}{l}[\,t_{i,2n+k-i+1} \to t_{i,2n+k-i+2}\,]_2\\ [\,f_{i,2n+k-i+1} \to f_{i,2n+k-i+2}\,]_2\end{array}\right\}$ para $k+2 \leq i \leq n$
    $\left.\begin{array}{l}[\,t_{i,2n+k-i+1} \to t_{i,2n+k-i+2}\,]_1\\ [\,f_{i,2n+k-i+1} \to f_{i,2n+k-i+2}\,]_1\end{array}\right\}$ para $1 \leq i \leq k$
    $[\,\alpha_{2n+k} \to \alpha_{2n+k+1}\,]_0$
    $[\,\beta_{2n+k} \to \beta_{2n+k+1}\,]_0$
    $[\,\gamma_{2n+k} \to \gamma_{2n+k+1}\,]_1$
    $\left.\begin{array}{l}[\,x_{i,j,2n+k} \to x_{i,j,2n+k+1}\,]_1\\ [\,\overline{x}_{i,j,2n+k} \to \overline{x}_{i,j,2n+k+1}\,]_1\\ [\,x^*_{i,j,2n+k} \to x^*_{i,j,2n+k+1}\,]_1\end{array}\right\}$ para $1 \leq i \leq n, 1 \leq j \leq p$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2n+k+1}(0) = \{\alpha_{2n+k+1}, \beta_{2n+k+1}\}$
- En la configuración $\mathcal{C}_{2n+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{2n+k+1}(\varphi)$;
  - un objeto γ_{2n+k+1};
  - p copias de T_i y F_i, $1 \leq i \leq n$; y
  - los objetos $r_{i,2n+k-i+2}$, $1 \leq i \leq k+1$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{2n+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,2n+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.

(a_1) El caso base $k = 1$ es trivial dado que:

- en la configuración $\mathcal{C}_{3n}$ tenemos que $\mathcal{C}_{3n}(0) = \{\alpha_{3n}, \beta_{3n}\}$ y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{3n}(\varphi)$, un objeto γ_{3n}, p copias de T_i y F_i, siendo $1 \leq i \leq n$ y un subconjunto diferente de objetos $r_{i,3n+1-i}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad correspondiente a la rama; y 2^n membranas vacías etiquetadas por 2. Por tanto, la configuración $\mathcal{C}_{3n}$ da lugar a la configuración $\mathcal{C}_{3n+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
t_{1,3n} F_1 [\ \ ]_2 \to [\, t_{1,3n+1} \,]_2 \\
f_{1,3n} T_1 [\ \ ]_2 \to [\, f_{1,3n+1} \,]_2 \\
\left.\begin{array}{l} [\, t_{i,3n-i+1} \to t_{i,3n-i+2} \,]_1 \\ [\, f_{i,3n-i+1} \to f_{i,3n-i+2} \,]_1 \end{array}\right\} \text{ para } 2 \leq i \leq n \\
[\, \alpha_{3n} \to \alpha_{3n+1} \,]_0 \\
[\, \beta_{3n} \to \beta_{3n+1} \,]_0 \\
[\, \gamma_{3n} \to \gamma_{3n+1} \,]_1 \\
\left.\begin{array}{l} [\, x_{i,j,3n} \to x_{i,j,3n+1} \,]_1 \\ [\, \overline{x}_{i,j,3n} \to \overline{x}_{i,j,3n+1} \,]_1 \\ [\, x^*_{i,j,3n} \to x^*_{i,j,3n+1} \,]_1 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p
\end{array}
$$

  Por lo tanto, $\mathcal{C}_{3n+1}(0) = \{\alpha_{3n+1}, \beta_{3n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{3n+1}(\varphi)$, un objeto γ_{3n+1}, p copias de T_i y F_i, siendo $2 \leq i \leq n$, y $p-1$ copias de T_1 (respectivamente, F_1) si tenemos el objeto f_1 (respectivamente, t_1) correspondiente en dicha rama, p copias de lo contrario, y un subconjunto diferente de objetos $r_{i,3n-i+2}$, $2 \leq i \leq n$, siendo $r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen un objeto $r_{1,3n+1}$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{3n+k}(0) = \{\alpha_{3n+k}, \beta_{3n+k}\}$
  - En la configuración $\mathcal{C}_{3n+k}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{3n+k}(\varphi)$;
    - un objeto γ_{3n+k};
    - p copias de cada objeto T_i y F_i, si $k+1 \leq i \leq n$ o los objetos t_i o f_i correspondientes están asignados a esa rama, $p-1$ copias de lo contrario; y
    - los objetos $r_{i,3n+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{3n+k}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,3n+k-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$.
    Por tanto, la configuración $\mathcal{C}_{3n+k}$ da lugar a la configuración $\mathcal{C}_{3n+k+1}$ por la aplicación de las reglas:
    $t_{k+1,3n+1}[\ \ ]_{t_{k+1,3n}F_k} \to [$
    $]_{t_{k+1,3n}F_k} f_{k+1,3n} T_k [\ \ ]_2 \to [\, f_{k+1,3n+1}\,]_2$
    $$\left.\begin{array}{l} [\, t_{i,3n+k-i+1} \to t_{i,3n+k-i+2}\,]_1 \\ [\, f_{i,3n+k-i+1} \to f_{i,3n+k-i+2}\,]_1 \end{array}\right\} \text{para } k+2 \leq i \leq n$$
    $$\left.\begin{array}{l} [\, t_{i,3n+k-i+1} \to t_{i,3n+k-i+2}\,]_2 \\ [\, f_{i,3n+k-i+1} \to f_{i,3n+k-i+2}\,]_2 \end{array}\right\} \text{para } 1 \leq i \leq k$$
    $[\, \alpha_{3n+k} \to \alpha_{3n+k+1}\,]_0$
    $[\, \beta_{3n+k} \to \beta_{3n+k+1}\,]_0$
    $[\, \gamma_{3n+k} \to \gamma_{3n+k+1}\,]_1$
    $$\left.\begin{array}{l} [\, x_{i,j,3n+k} \to x_{i,j,3n+k+1}\,]_1 \\ [\, \overline{x}_{i,j,3n+k} \to \overline{x}_{i,j,3n+k+1}\,]_1 \\ [\, x^*_{i,j,3n+k} \to x^*_{i,j,3n+k+1}\,]_1 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$
    Por lo tanto, lo siguiente es válido
  - $\mathcal{C}_{3n+k+1}(0) = \{\alpha_{3n+k+1}, \beta_{3n+k+1}\}$
  - En la configuración $\mathcal{C}_{3n+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{3n+k+1}(\varphi)$;
    - un objeto γ_{3n+k+1};
    - p copias de cada objeto T_i y F_i, si $k+2 \leq i \leq n$ o si el objeto t_i o f_i correspondiente está asignado a dicha rama, $p-1$ copias de lo contrario; y

◇ los objetos $r_{i,3n+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.

∘ En la configuración $\mathcal{C}_{3n+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,3n+k-i+2}$, $1 \leq i \leq k+1$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para l $(0 \leq l \leq p-1)$

(a_0) El caso base $k = 1$ es trivial dado que:

∘ en la configuración $\mathcal{C}_{2n+(l+1)n}$[1] tenemos que: $\mathcal{C}_{2n+(l+1)n}(0) = \{\alpha_{2n+(l+1)n}, \beta_{2n+(l+1)n}\}$ y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{2n+(l+1)n}(\varphi)$, un objeto $\gamma_{2n+(l+1)n}$ y p copias de T_i y F_i, siendo $1 \leq i \leq n$, y $p-l$ copias for T_i (respectivamente, F_i) objetos que pertenecen a una rama con un objeto f_i (respectivamente, t_i); y 2^n membranas etiquetadas por 2 que contienen un subconjunto diferente de objetos $r_{i,2n+(l+1)n-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$, la valoración de verdad correspondiente a la rama. Por tanto, la configuración $\mathcal{C}_{2n+(l+1)n}$ da lugar a la configuración $\mathcal{C}_{2n+(l+1)n+1}$ por la aplicación de las reglas:

$$\left.\begin{array}{l} [\, t_{i,2n+(l+1)n} \,]_2 \to t_{i,2n+(l+1)n+1} [\quad]_2 \\ [\, f_{i,2n+(l+1)n} \,]_2 \to f_{i,2n+(l+1)n+1} [\quad]_2 \\ [\, t_{i,2n+(l+1)n+1-i} \to t_{i,2n+(l+1)n+2-i} \,]_2 \\ [\, f_{i,2n+(l+1)n+1-i} \to f_{i,2n+(l+1)n+2-i} \,]_2 \end{array}\right\} \text{para } 2 \leq i \leq n$$

$$\left.\begin{array}{l} [\, \alpha_{2n+(l+1)n} \to \alpha_{2n+(l+1)n+1} \,]_0 \\ [\, \beta_{2n+(l+1)n} \to \beta_{2n+(l+1)n+1} \,]_0 \\ [\, \gamma_{2n+(l+1)n} \to \gamma_{2n+(l+1)n+1} \,]_1 \\ [\, x_{i,j,2n+(l+1)n} \to x_{i,j,2n+(l+1)n+1} \,]_1 \\ [\, \overline{x}_{i,j,2n+(l+1)n} \to \overline{x}_{i,j,2n+(l+1)n+1} \,]_1 \\ [\, x^*_{i,j,2n+(l+1)n} \to x^*_{i,j,2n+(l+1)n+1} \,]_1 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, $\mathcal{C}_{2n+(l+1)n+1}(0) = \{\alpha_{2n+(l+1)n+1}, \beta_{2n+(l+1)n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{2n+(l+1)n+1}(\varphi)$, un objeto $\gamma_{2n+(l+1)n+1}$, p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha

[1] Obsérvese que $(l+1)n = ln + n$, y se ha demostrado en el primer paso de inducción que es correcto.

rama, y $p - l$ copias de F_i (respectivamente, T_i) y un objeto $r_{1,2n+(l+1)n+1}$, siendo $r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen un subconjunto diferente de objetos $r_{i,2n+(l+1)n-i+2}$, $2 \leq i \leq n$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{2n+(l+1)n+k}(0) = \{\alpha_{2n+(l+1)n+k}, \beta_{2n+(l+1)n+k}\}$
  - En la configuración $\mathcal{C}_{2n+(l+1)n+k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2n+(l+1)n+k}(\varphi)$;
    - un objeto $\gamma_{2n+(l+1)n+k}$;
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p - l$ copias de F_i (respectivamente, T_i); y
    - los objetos $r_{i,2n+(l+1)n+k-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{2n+(l+1)n+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,2n+(l+1)n+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.
    Por tanto, la configuración $\mathcal{C}_{2n+k}$ da lugar a la configuración $\mathcal{C}_{2n+(l+1)n+k+1}$ por la aplicación de las reglas:
    $[\, t_{k+1,2n+(l+1)n}\,]_2 \to t_{k+1,2n+(l+1)n+1}\,[\quad]_2$
    $[\, f_{k+1,2n+(l+1)n}\,]_2 \to f_{k+1,2n+(l+1)n+1}\,[\quad]_2$
    $\left.\begin{array}{l}[\, t_{i,2n+(l+1)n+k-i+1} \to t_{i,2n+k-i+2}\,]_2 \\ [\, f_{i,2n+(l+1)n+k-i+1} \to f_{i,2n+k-i+2}\,]_2\end{array}\right\}$ para $k+2 \leq i \leq n$
    $\left.\begin{array}{l}[\, t_{i,2n+(l+1)n+k-i+1} \to t_{i,2n+(l+1)n+k-i+2}\,]_1 \\ [\, f_{i,2n+(l+1)n+k-i+1} \to f_{i,2n+(l+1)n+k-i+2}\,]_1\end{array}\right\}$ para $1 \leq i \leq k$
    $[\, \alpha_{2n+(l+1)n+k} \to \alpha_{2n+(l+1)n+k+1}\,]_0$
    $[\, \beta_{2n+(l+1)n+k} \to \beta_{2n+(l+1)n+k+1}\,]_0$
    $[\, \gamma_{2n+(l+1)n+k} \to \gamma_{2n+(l+1)n+k+1}\,]_1$
    $\left.\begin{array}{l}[\, x_{i,j,2n+(l+1)n+k} \to x_{i,j,2n+(l+1)n+k+1}\,]_1 \\ [\, \overline{x}_{i,j,2n+(l+1)n+k} \to \overline{x}_{i,j,2n+(l+1)n+k+1}\,]_1 \\ [\, x^*_{i,j,2n+(l+1)n+k} \to x^*_{i,j,2n+(l+1)n+k+1}\,]_1\end{array}\right\}$ para $1 \leq i \leq n, 1 \leq j \leq p$
    Por lo tanto, lo siguiente es válido
  - $\mathcal{C}_{2n+(l+1)n+k+1}(0) = \{\alpha_{2n+(l+1)n+k+1}, \beta_{2n+(l+1)n+k+1}\}$

- En la configuración $\mathcal{C}_{2n+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{2n+(l+1)n+k+1}(\varphi)$;
  - un objeto $\gamma_{2n+(l+1)n+k+1}$;
  - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p - l$ copias de F_i (respectivamente, T_i); y
  - los objetos $r_{i,2n+(l+1)n+k-i+2}$, $1 \leq i \leq k+1$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{2n+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,2n+(l+1)n+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.

(a_1) El caso base $k = 1$ es trivial dado que:

- en la configuración $\mathcal{C}_{3n+(l+1)n}$ tenemos que $\mathcal{C}_{3n+(l+1)n}(0) = \{\alpha_{3n+(l+1)n}, \beta_{3n+(l+1)n}\}$ y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{3n+(l+1)n}(\varphi)$, un objeto $\gamma_{3n+(l+1)n}$, p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p-l$ copias de F_i (respectivamente, T_i) y un subconjunto diferente de objetos $r_{i,3n+(l+1)n-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad correspondiente a la rama; y 2^n membranas vacías etiquetadas por 2. Por tanto, la configuración $\mathcal{C}_{3n+(l+1)n}$ da lugar a la configuración $\mathcal{C}_{3n+(l+1)n+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
t_{1,3n+(l+1)n} F_1 [\ \ ]_2 \to [\, t_{1,3n+(l+1)n+1} \,]_2 \\
f_{1,3n+(l+1)n} T_1 [\ \ ]_2 \to [\, f_{1,3n+(l+1)n+1} \,]_2 \\
\left.\begin{array}{l}
[\, t_{i,3n+(l+1)n-i+1} \to t_{i,3n+(l+1)n-i+2} \,]_1 \\
[\, f_{i,3n+(l+1)n-i+1} \to f_{i,3n+(l+1)n-i+2} \,]_1
\end{array}\right\} \text{para } 2 \leq i \leq n \\
[\, \alpha_{3n+(l+1)n} \to \alpha_{3n+(l+1)n+1} \,]_0 \\
[\, \beta_{3n+(l+1)n} \to \beta_{3n+(l+1)n+1} \,]_0 \\
[\, \gamma_{3n+(l+1)n} \to \gamma_{3n+(l+1)n+1} \,]_1 \\
\left.\begin{array}{l}
[\, x_{i,j,3n+(l+1)n} \to x_{i,j,3n+(l+1)n+1} \,]_1 \\
[\, \overline{x}_{i,j,3n+(l+1)n} \to \overline{x}_{i,j,3n+(l+1)n+1} \,]_1 \\
[\, x^*_{i,j,3n+(l+1)n} \to x^*_{i,j,3n+(l+1)n+1} \,]_1
\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p
\end{array}
$$

Por lo tanto, $\mathcal{C}_{3n+(l+1)n+1}(0) = \{\alpha_{3n+(l+1)n+1}, \beta_{3n+(l+1)n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{3n+(l+1)n+1}(\varphi)$, un objeto $\gamma_{3n+(l+1)n+1}$, p

copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p - l$ copias de F_i (respectivamente, T_i) si $k + 1 \leq i \leq n$, $p - l - 1$ de lo contrario, y un subconjunto diferente de objetos $r_{i,3n+(l+1)n-i+2}$, $2 \leq i \leq n$, siendo $r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen un objeto $r_{1,3n+(l+1)n+1}$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{3n+(l+1)n+k}(0) = \{\alpha_{3n+(l+1)n+k}, \beta_{3n+(l+1)n+k}\}$
  - En la configuración $\mathcal{C}_{3n+(l+1)n+k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{3n+(l+1)n+k}(\varphi)$;
    - un objeto $\gamma_{3n+(l+1)n+k}$;
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p - l$ copias de F_i (respectivamente, T_i) si $k + 1 \leq i \leq n$, $p - l - 1$ de lo contrario; y
    - los objetos $r_{i,3n+k-i+1}$, $k + 1 \leq i \leq n$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{3n+(l+1)n+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,3n+(l+1)n-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$. Por tanto, la configuración $\mathcal{C}_{3n+(l+1)n+k}$ da lugar a la configuración $\mathcal{C}_{3n+(l+1)n+k+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
t_{k+1,3n+(l+1)n} F_k\, [\ \ ]_2 \to [\, t_{k+1,3n+(l+1)n+1}\,]_2 \\
f_{k+1,3n+(l+1)n} T_k\, [\ \ ]_2 \to [\, f_{k+1,3n+(l+1)n+1}\,]_2 \\
\left.\begin{array}{l}
[\, t_{i,3n+(l+1)n+k-i+1} \to t_{i,3n+(l+1)n+k-i+2}\,]_1 \\
[\, f_{i,3n+(l+1)n+k-i+1} \to f_{i,3n+(l+1)n+k-i+2}\,]_1
\end{array}\right\} \text{para } k+2 \leq i \leq n \\
\left.\begin{array}{l}
[\, t_{i,3n+(l+1)n+k-i+1} \to t_{i,3n+(l+1)n+k-i+2}\,]_2 \\
[\, f_{i,3n+(l+1)n+k-i+1} \to f_{i,3n+(l+1)n+k-i+2}\,]_2
\end{array}\right\} \text{para } 1 \leq i \leq k \\
[\, \alpha_{3n+(l+1)n+k} \to \alpha_{3n+(l+1)n+k+1}\,]_0 \\
[\, \beta_{3n+(l+1)n+k} \to \beta_{3n+(l+1)n+k+1}\,]_0 \\
[\, \gamma_{3n+(l+1)n+k} \to \gamma_{3n+(l+1)n+k+1}\,]_1 \\
\left.\begin{array}{l}
[\, x_{i,j,3n+(l+1)n+k} \to x_{i,j,3n+(l+1)n+k+1}\,]_1 \\
[\, \overline{x}_{i,j,3n+(l+1)n+k} \to \overline{x}_{i,j,3n+(l+1)n+k+1}\,]_1 \\
[\, x^*_{i,j,3n+(l+1)n+k} \to x^*_{i,j,3n+(l+1)n+k+1}\,]_1
\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p
\end{array}
$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{3n+(l+1)n+k+1}(0) = \{\alpha_{3n+(l+1)n+k+1}, \beta_{3n+(l+1)n+k+1}\}$
- En la configuración $\mathcal{C}_{3n+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{3n+(l+1)n+k+1}(\varphi)$;
  - un objeto $\gamma_{3n+(l+1)n+k+1}$;
  - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p - l$ copias de F_i (respectivamente, T_i) si $k + 2 \leq i \leq n$, $p - l - 1$ de lo contrario; y
  - los objetos $r_{i,3n+(l+1)n+k-i+2}$, $k + 2 \leq i \leq n$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{3n+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene un subconjunto diferente de objetos $r_{i,3n+(l+1)n+k-i+2}$, $1 \leq i \leq k + 1$, siendo $r \in \{t, f\}$.

- Para probar (b) basta con darse cuenta de que, por una parte, de (a), en la configuración $\mathcal{C}_{n+2np-1}$[2] se tiene que:

  - $\mathcal{C}_{n+2np-1}(0) = \{\alpha_{n+2np-1}, \beta_{n+2np-1}\}$
  - En la configuración $\mathcal{C}_{n+2np-1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{n+2np-1}(\varphi)$;
    - un objeto $\gamma_{n+2np-1}$;
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de lo contrario; y
    - los objetos $r_{i,n+2np-i}$, $1 \leq i \leq n - 1$
  - En la configuración $\mathcal{C}_{n+2np-1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene un objeto $r_{n,2np}$, siendo $r \in \{t, f\}$.

    Por tanto, la configuración $\mathcal{C}_{n+2np-1}$ da lugar a la configuración $\mathcal{C}_{n+2np}$ por la aplicación de las reglas:

    $[\, t_{n,2np} \,]_2 \to t_{n,2np+1} [\quad]_2$
    $[\, f_{n,2np} \,]_2 \to f_{n,2np+1} [\quad]_2$

[2] Obsérvese que $n + 2np - 1 = 2n + 2n(p - 1) + (n - 1)$.

$$\left.\begin{array}{l}[\, t_{i,n+2np-i} \to t_{i,n+2np-i+1} \,]_1 \\ [\, f_{i,n+2np-i} \to f_{i,n+2np-i} \,]_1\end{array}\right\} \text{ para } 1 \leq i \leq n-1$$

$$[\, \alpha_{n+2np-1} \to \alpha_{n+2np} \,]_0$$
$$[\, \beta_{n+2np-1} \to \beta_{n+2np} \,]_0$$
$$[\, \gamma_{n+2np-1} \to \gamma_{n+2np} \,]_1$$

$$\left.\begin{array}{l}[\, x_{i,j,n+2np-1} \to x_{i,j,n+2np} \,]_1 \\ [\, \overline{x}_{i,j,n+2np-1} \to \overline{x}_{i,j,n+2np} \,]_1 \\ [\, x^*_{i,j,n+2np-1} \to x^*_{i,j,n+2np} \,]_1\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, tenemos que $\mathcal{C}_{n+2np}(0) = \{\alpha_{n+2np}, \beta_{n+2np}\}$, y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{n+2np}(\varphi)$, un objeto γ_{n+2np}, p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de lo contrario y un subconjunto diferente de objetos $r_{i,n+2np-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad asociada a la rama; y 2^n membranas vacías etiquetadas por 2.

□

Proposición 7.3. *Sea $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ una computación del sistema $\Pi(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$.*

(*a*) *Para cada k ($1 \leq k \leq n-1$) en la configuración $\mathcal{C}_{n+2np+k}$ tenemos lo siguiente:*

- $\mathcal{C}_{n+2np+k}(0) = \{\alpha_{n+2np+k}, \beta_{n+2np+k}\}$
- *Hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada $cod_{n+2np+k}(\varphi)$;*
  - *un objeto $\gamma_{n+2np+k}$;*
  - *p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia of F_i (respectivamente, T_i) si $k+1 \leq i \leq n$; y*
  - *los objetos $r_{i,n+2np+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.*
- *hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene k objetos #*

(*b*) *$\mathcal{C}_{2n+2np}(0) = \{\alpha_{2n+2np}, \beta_{2n+2np}\}$, y en la configuración $\mathcal{C}_{2n+2np}$ hay 2^n membranas etiquetadas por 1, tales que cada una de ellas contiene el*

multiconjunto de entrada $cod_{2n+2np}(\varphi)$, un objeto γ_{2n+2np}, p copias de cada objeto T_i y F_i, $1 \leq i \leq n$ si la valoración de verdad asociada a la rama contiene su correspondiente objeto t_i o f_i; y 2^n membranas etiquetadas por 2, tales que cada una de ellas contiene n objetos $\#$.

Demostración. (a) será demostrado por inducción sobre k

- el caso base $k = 1$ es trivial dado que:
  - en la configuración $\mathcal{C}_{n+2np}$ tenemos que $\mathcal{C}_{n+2np}(0) = \{\alpha_{n+2np}, \beta_{n+2np}\}$ y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{n+2np}(\varphi)$, un objeto γ_{n+2np} p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de lo contrario y un subconjunto diferente de objetos $r_{i,n+2np-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad asociada a la rama; y 2^n membranas vacías etiquetadas por 2. Por tanto, la configuración $\mathcal{C}_{n+2np}$ da lugar a la configuración $\mathcal{C}_{n+2np+1}$ por la aplicación de las reglas:
    $$t_{1,n+2np} F_1 [\ \ ]_2 \to [\,\#\,]_2$$
    $$f_{1,n+2np} T_1 [\ \ ]_2 \to [\,\#\,]_2$$
    $$\left.\begin{array}{l} [\, t_{i,n+2np-i+1} \to t_{i,n+2np-i+2}\,]_1 \\ [\, f_{i,n+2np-i+1} \to f_{i,n+2np-i+2}\,]_1 \end{array}\right\} \text{ para } 2 \leq i \leq n$$
    $$[\, \alpha_{n+2np} \to \alpha_{n+2np+1}\,]_0$$
    $$[\, \beta_{n+2np} \to \beta_{n+2np+1}\,]_0$$
    $$[\, \gamma_{n+2np} \to \gamma_{n+2np+1}\,]_1$$
    $$\left.\begin{array}{l} [\, x_{i,j,n+2np} \to x_{i,j,n+2np+1}\,]_1 \\ [\, \overline{x}_{i,j,n+2np} \to \overline{x}_{i,j,n+2np+1}\,]_1 \\ [\, x^*_{i,j,n+2np} \to x^*_{i,j,n+2np+1}\,]_1 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$
    Por lo tanto, $\mathcal{C}_{n+2np+1}(0) = \{\alpha_{n+2np+1}, \beta_{n+2np+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{n+2np+1}(\varphi)$, un objeto $\gamma_{n+2np+1}$, p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia of F_i (respectivamente, T_i) si $k+2 \leq i \leq n$ y objetos $r_{i,n+2np-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen un objeto $\#$.
- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n-1$)

- $\mathcal{C}_{n+2np+k}(0) = \{\alpha_{n+2np+k}, \beta_{n+2np+k}\}$
- En la configuración $\mathcal{C}_{n+2np+k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{n+2np+k}(\varphi)$;
  - un objeto $\gamma_{n+2np+k}$;
  - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia of F_i (respectivamente, T_i) si $k+1 \leq i \leq n$; y
  - los objetos $r_{i,n+2np+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{n+2np+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene k objetos #.

  Por tanto, la configuración $\mathcal{C}_{n+2np+k}$ da lugar a la configuración $\mathcal{C}_{n+2np+k+1}$ por la aplicación de las reglas:

  $$\begin{array}{l} t_{k+1,n+2np}F_1\,[\quad]_2 \to [\,\#\,]_2 \\ f_{k+1,n+2np}T_1\,[\quad]_2 \to [\,\#\,]_2 \\ \left.\begin{array}{l}[\,t_{i,n+2np+k-i+1} \to t_{i,n+2np+k-i+2}\,]_1 \\ [\,f_{i,n+2np+k-i+1} \to f_{i,n+2np+k-i+2}\,]_1\end{array}\right\} \text{para } 2 \leq i \leq n \\ [\,\alpha_{n+2np+k} \to \alpha_{n+2np+k+1}\,]_0 \\ [\,\beta_{n+2np+k} \to \beta_{n+2np+k+1}\,]_0 \\ [\,\gamma_{n+2np+k} \to \gamma_{n+2np+k+1}\,]_1 \\ \left.\begin{array}{l}[\,x_{i,j,n+2np+k} \to x_{i,j,n+2np+k+1}\,]_1 \\ [\,\overline{x}_{i,j,n+2np+k} \to \overline{x}_{i,j,n+2np+k+1}\,]_1 \\ [\,x^*_{i,j,n+2np+k} \to x^*_{i,j,n+2np+k+1}\,]_1\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p \end{array}$$

  Por lo tanto, lo siguiente es válido
- $\mathcal{C}_{n+2np+k+1}(0) = \{\alpha_{n+2np+k+1}, \beta_{n+2np+k+1}\}$
- En la configuración $\mathcal{C}_{n+2np+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{n+2np+k+1}(\varphi)$;
  - un objeto $\gamma_{n+2np+k+1}$;
  - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia of F_i (respectivamente, T_i) si $k+2 \leq i \leq n$; y
  - los objetos $r_{i,n+2np+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{n+2np+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene $k+1$ objetos #.

- Para probar (b) basta darse cuenta que, por una parte, por (a) la configuración $\mathcal{C}_{2n+2np-1}$[3] es válido:
  - $\mathcal{C}_{2n+2np-1}(0) = \{\alpha_{2n+2np-1}, \beta_{2n+2np-1}\}$
  - En la configuración $\mathcal{C}_{2n+2np-1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2n+2np-1}(\varphi)$;
    - un objeto $\gamma_{2n+2np-1}$;
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de F_n (respectivamente, T_n); y
    - un objeto $r_{n,n+2np}$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{2n+2np-1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene $n-1$ objetos $\#$.

    Por tanto, la configuración $\mathcal{C}_{2n+2np-1}$ da lugar a la configuración $\mathcal{C}_{2n+2np}$ por la aplicación de las reglas:

    $$t_{n,n+2np} F_1 [\ \ ]_2 \to [\,\#\,]_2$$
    $$f_{n,n+2np} T_1 [\ \ ]_2 \to [\,\#\,]_2$$
    $$[\,\alpha_{2n+2np-1} \to \alpha_{2n+2np}\,]_0$$
    $$[\,\beta_{2n+2np-1} \to \beta_{2n+2np}\,]_0$$
    $$[\,\gamma_{2n+2np-1} \to \gamma_{2n+2np}\,]_1$$
    $$\left.\begin{array}{l} [\,x_{i,j,2n+2np-1} \to x_{i,j,2n+2np}\,]_1 \\ [\,\overline{x}_{i,j,2n+2np-1} \to \overline{x}_{i,j,2n+2np}\,]_1 \\ [\,x^*_{i,j,2n+2np-1} \to x^*_{i,j,2n+2np}\,]_1 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

    Por lo tanto, lo siguiente es válido
  - $\mathcal{C}_{2n+2np}(0) = \{\alpha_{2n+2np}, \beta_{2n+2np}\}$
  - En la configuración $\mathcal{C}_{2n+2np}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2n+2np}(\varphi)$;
    - un objeto γ_{2n+2np}; y
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama.
  - En la configuración $\mathcal{C}_{2n+2np}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene n objetos $\#$.

□

[3] Obsérvese que $2n + 2np - 1 = n + 2np + (n-1)$.

Primera fase de chequeo

En esta fase, tratamos de determinar qué cláusulas son satisfechas por la valoración de verdad codificada en cada rama. Para ello, las reglas de 2.1 serán aplicadas de tal manera que en el m-ésimo paso, siendo $m = ln + k$ $(1 \leq k \leq n, 0 \leq l \leq p-1)$, la cláusula C_{l+1} será evaluada con la k-ésima variable de la fórmula φ. Esta fase tarda np pasos.

Proposición 7.4. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(*a*) *Para cada* k $(1 \leq k \leq n)$ *y* l $(0 \leq l \leq p-1)$ *en la configuración* $\mathcal{C}_{2n+2np+ln+k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2n+2np+ln+k}(0) = \{\alpha_{2n+2np+ln+k}, \beta_{2n+2np+ln+k}\}$
- *Hay* 2^n *membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *los* $(n-k)$*-ésimos últimos elementos de* $cod_{2n+2np+ln+k}(\varphi)_{l+1}^{l+1}$;
  - *el multiconjunto de entrada* $cod_{2n+2np+ln+k}(\varphi)_{l+2}^{p}$;
  - *un objeto* $\gamma_{2n+2np+ln+k}$; *y*
  - $p-l$ *copias de objetos* T_i *o* F_i, $k+1 \leq i \leq n$, $p-l-1$ *copias de lo contrario, correspondiendo a la valoración de verdad asignada a la rama.*
- *Hay* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - m *objetos* $c_{j,t}$ $(1 \leq j \leq l+1,\ 0 \leq t \leq ln+k-1)$*; es decir, las cláusulas que han sido satisfechas por alguna variable; y*
  - $n + ln + k - m$ *objetos* #.

(*b*) $\mathcal{C}_{2n+3np}(0) = \{\alpha_{2n+3np}, \beta_{2n+3np}\}$, *y en la configuración* $\mathcal{C}_{2n+3np}$ *hay* 2^n *membranas etiquetadas por 1, tales que cada una de ellas contiene un objeto* γ_{2n+3np}; *y* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene* m *objetos* $c_{j,t}$ $(1 \leq j \leq p,\ 0 \leq t \leq np-1)$*; es decir, las cláusulas satisfechas por alguna variable y* $n + np - m$ *objetos* #.

Demostración. (*a*) será demostrado por inducción sobre l

- El caso base $l = 0$ será demostrado por inducción sobre k
  - El caso base $k = 1$ es trivial dado que:

- en la configuración $\mathcal{C}_{2n+2np}$ tenemos que: $\mathcal{C}_{2n+2np}(0) = \{\alpha_{2n+2np}, \beta_{2n+2np}\}$ y hay 2^n membranas etiquetadas por 1, tales que cada una de ellas contiene el multiconjunto de entrada $cod_{2n+2np}(\varphi)$, un objeto γ_{2n+2np} y p copias de objetos T_i y F_i, $1 \leq i \leq n$, que representan la valoración de verdad correspondiente a dicha rama; y 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene n objetos #. Por tanto, la configuración $\mathcal{C}_{2n+2np}$ da lugar a la configuración $\mathcal{C}_{2n+2np+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
T_1 x_{1,1,2n+2np}\,[\quad]_2 \to [\,c_{1,0}\,]_2 \\
T_1 \overline{x}_{1,1,2n+2np}\,[\quad]_2 \to [\,\#\,]_2 \\
T_1 x^*_{1,1,2n+2np}\,[\quad]_2 \to [\,\#\,]_2 \quad {}^{4} \\
F_1 x_{1,1,2n+2np}\,[\quad]_2 \to [\,\#\,]_2 \\
F_1 \overline{x}_{1,1,2n+2np}\,[\quad]_2 \to [\,c_{1,0}\,]_2 \\
F_1 x^*_{1,1,2n+2np}\,[\quad]_2 \to [\,\#\,]_2 \\
[\,\alpha_{2n+2np} \to \alpha_{2n+2np+1}\,]_0 \\
[\,\beta_{2n+2np} \to \beta_{2n+2np+1}\,]_0 \\
[\,\gamma_{2n+2np} \to \gamma_{2n+2np+1}\,]_1 \\
\left.\begin{array}{l}
[\,x_{i,j,2n+2np} \to x_{i,j,2n+2np+1}\,]_1 \\
[\,\overline{x}_{i,j,2n+2np} \to \overline{x}_{i,j,2n+2np+1}\,]_1 \\
[\,x^*_{i,j,2n+2np} \to x^*_{i,j,2n+2np+1}\,]_1
\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p
\end{array}
$$

  Por lo tanto, $\mathcal{C}_{2n+2np+1}(0) = \{\alpha_{2n+2np+1}, \beta_{2n+2np+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen los últimos $n-1$ elementos de $cod_{2n+2np+1}(\varphi)^1_1$, el multiconjunto de entrada $cod_{2n+2np+1}(\varphi)^p_2$, p copias de T_i o F_i, siendo $2 \leq i \leq n$, y $p-1$ copias de T_1 o F_1; y 2^n membranas etiquetadas por 2 que contienen n objetos # y un objeto $c_{1,0}$ si la valoración de verdad correspondiente hace verdadera la cláusula 1 con la variable 1, otro objeto # de lo contrario.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{2n+2np+k}(0) = \{\alpha_{2n+2np+k}, \beta_{2n+2np+k}\}$
  - En la configuración $\mathcal{C}_{2n+2np+k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - los $(n-k)$-ésimos últimos elementos de $cod_{2n+2np+k}(\varphi)^1_1$;
    - el multiconjunto de entrada $cod_{2n+2np+k}(\varphi)^p_2$;
    - un objeto $\gamma_{2n+2np+k}$; y

[4] Si $k = 1, l = 0$, entonces $i = 1, j = 1$, así que $2np + 2n + n(j-1) + (i-1) = 2n + 2np$.

- $\diamond$ p copias de objetos T_i o F_i, $k+1 \leq i \leq n$, $p-1$ copias si $1 \leq i \leq k$, correspondiendo a la valoración de verdad asignada a la rama.

- En la configuración $\mathcal{C}_{2n+2np+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
  - $\diamond$ m objetos $c_{1,t}$ $(0 \leq t \leq k-1)$; es decir, el número de variables que hace verdadera la fórmula φ con la valoración de verdad correspondiente; y
  - $\diamond$ $n+k-m$ objetos $\#$.

  Por tanto, la configuración $\mathcal{C}_{2n+2np+k}$ da lugar a la configuración $\mathcal{C}_{2n+2np+k+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
T_k x_{k+1,1,2n+2np+k}\,[\quad]_2 \to [\,c_{1,0}\,]_2 \\
T_k \overline{x}_{k+1,1,2n+2np+k}\,[\quad]_2 \to [\,\#\,]_2 \\
T_k x^*_{k+1,1,2n+2np+k}\,[\quad]_2 \to [\,\#\,]_2 \quad {}^{5} \\
F_k x_{k+1,1,2n+2np+k}\,[\quad]_2 \to [\,\#\,]_2 \\
F_k \overline{x}_{k+1,1,2n+2np+k}\,[\quad]_2 \to [\,c_{1,0}\,]_2 \\
F_k x^*_{k+1,1,2n+2np+k}\,[\quad]_2 \to [\,\#\,]_2 \\
[\,\alpha_{2n+2np+k} \to \alpha_{2n+2np+k+1}\,]_0 \\
[\,\beta_{2n+2np+k} \to \beta_{2n+2np+k+1}\,]_0 \\
[\,\gamma_{2n+2np+k} \to \gamma_{2n+2np+k+1}\,]_1 \\
\left.\begin{array}{l}
[\,x_{i,j,2n+2np+k} \to x_{i,j,2n+2np+k+1}\,]_1 \\
[\,\overline{x}_{i,j,2n+2np+k} \to \overline{x}_{i,j,2n+2np+k+1}\,]_1 \\
[\,x^*_{i,j,2n+2np+k} \to x^*_{i,j,2n+2np+k+1}\,]_1
\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p \\
[\,c_{1,t} \to c_{1,t+1}\,]_2 \text{ para } 0 \leq t \leq k-1
\end{array}
$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2n+2np+k+1} = \{\alpha_{2n+2np+k+1}, \beta_{2n+2np+k+1}\}$
- En la configuración $\mathcal{C}_{2n+2np+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
  - $\diamond$ los $(n-k+1)$-ésimos últimos elementos de $cod_{2n+2np+k+1}(\varphi)^1_1$;
  - $\diamond$ el multiconjunto de entrada $cod_{2n+2np+k+1}(\varphi)^p_2$,
  - $\diamond$ un objeto $\gamma_{2n+2np+k+1}$; y
  - $\diamond$ p copias de objetos T_i o F_i, $k+2 \leq i \leq n$, $p-1$ copias si $1 \leq i \leq k+1$, correspondiendo a la valoración de verdad asignada a la rama.
- En la configuración $\mathcal{C}_{2n+2np+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene

[5] Si $l = 0$, entonces $i = k+1, j = 1$, así que $2np + 2n + n(j-1) + (i-1) = 2n + 2np + k$.

◇ m objetos $c_{1,t}$ $(0 \leq t \leq k)$; es decir, el número de variables que hace verdadera la cláusula C_1 con la valoración de verdad correspondiente; y

◇ $n + k + 1 - m$ objetos #.

- Suponiendo, por inducción, que el resultado es cierto para l $(0 \leq l \leq p-1)$

  - El caso base $k = 1$ es trivial dado que:

    ○ en la configuración $\mathcal{C}_{2n+2np+(l+1)n}$ tenemos que: $\mathcal{C}_{2n+2np+(l+1)n}(0) = \{\alpha_{2n+2np+(l+1)n}, \beta_{2n+2np+(l+1)n}\}$ y hay 2^n membranas etiquetadas por 1 que contienen el multiconjunto de entrada $cod_{2n+2np+(l+1)n}(\varphi)^p_{l+1}$, un objeto $\gamma_{2n+2np+(l+1)n}$ y $p-l$ copias de objetos T_i o F_i, $1 \leq i \leq n$; y 2^n membranas etiquetadas por 2 que contienen m objetos $c_{j,t}$ $(1 \leq j \leq l, 0 \leq t \leq ln-1)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_l con la valoración de verdad correspondiente y $n+(l+1)n-m$ objetos #. Por tanto, la configuración $\mathcal{C}_{2n+2np+(l+1)n}$ da lugar a la configuración $\mathcal{C}_{2n+2np+(l+1)n+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
T_1 x_{1,1,2n+2np+(l+1)n}\,[\ \ ]_2 \to [\,c_{l+1,0}\,]_2 \\
T_1 \overline{x}_{1,1,2n+2np+(l+1)n}\,[\ \ ]_2 \to [\,\#\,]_2 \\
T_1 x^*_{1,1,2n+2np+(l+1)n}\,[\ \ ]_2 \to [\,\#\,]_2 \\
F_1 x_{1,1,2n+2np+(l+1)n}\,[\ \ ]_2 \to [\,\#\,]_2 \\
F_1 \overline{x}_{1,1,2n+2np+(l+1)n}\,[\ \ ]_2 \to [\,c_{l+1,0}\,]_2 \\
F_1 x^*_{1,1,2n+2np+(l+1)n}\,[\ \ ]_2 \to [\,\#\,]_2 \\
[\,\alpha_{2n+2np+(l+1)n} \to \alpha_{2n+2np+(l+1)n+1}\,]_0 \\
[\,\beta_{2n+2np+(l+1)n} \to \beta_{2n+2np+(l+1)n+1}\,]_0 \\
[\,\gamma_{2n+2np+(l+1)n} \to \gamma_{2n+2np+(l+1)n+1}\,]_1 \\
\left.\begin{array}{l}
[\,x_{i,j,2n+2np+(l+1)n} \to x_{i,j,2n+2np+(l+1)n+1}\,]_1 \\
[\,\overline{x}_{i,j,2n+2np+(l+1)n} \to \overline{x}_{i,j,2n+2np+(l+1)n+1}\,]_1 \\
[\,x^*_{i,j,2n+2np+(l+1)n} \to x^*_{i,j,2n+2np+(l+1)n+1}\,]_1
\end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n \\ 1 \leq j \leq p \end{array} \\
[\,c_{j,t} \to c_{j,t+1}\,]_2 \text{ para } 1 \leq j \leq l+1, 0 \leq t \leq ln-1
\end{array}
$$

Por lo tanto, $\mathcal{C}_{2n+2np+(l+1)n+1}(0) = \{\alpha_{2n+2np+(l+1)n+1}, \beta_{2n+2np+(l+1)n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen los últimos $n-1$ elementos de $cod_{2n+2np+(l+1)n+1}(\varphi)^{l+1}_{l+1}$, el multiconjunto de entrada $cod_{2n+2np+(l+1)n+1}(\varphi)^p_{l+2}$, $p-l$ copias de T_i o F_i, siendo $2 \leq i \leq n$, y $p-l-1$ copias de T_1 o F_1; y 2^n membranas etiquetadas

por 2 que contienen m objetos $c_{j,t}$ ($1 \leq j \leq ln$, $0 \leq t \leq ln$); es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_{l+1} con la valoración de verdad correspondiente y $n + (l+1)n + 1 - m$ objetos #.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{2n+2np+(l+1)n+k}(0) = \{\alpha_{2n+2np+(l+1)n+k}, \beta_{2n+2np+(l+1)n+k}\}$
  - En la configuración $\mathcal{C}_{2n+2np+(l+1)n+k}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
    - los $(n-k)$-ésimos últimos elementos de $cod_{2n+2np+(l+1)n+k}(\varphi)_{l+1}^{l+1}$;
    - el multiconjunto de entrada $cod_{2n+2np+(l+1)n+k}(\varphi)_{l+2}^{p}$,
    - un objeto $\gamma_{2n+2np+(l+1)n+k}$; y
    - $p - l$ copias de objetos T_i o F_i, $k+1 \leq i \leq n$, $p - l - 1$ copias si $1 \leq i \leq k$, correspondiendo a la valoración de verdad asignada a la rama.
  - En la configuración $\mathcal{C}_{2n+2np+(l+1)n+k}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
    - m objetos $c_{j,t}$ ($1 \leq j \leq l+1$, $0 \leq t \leq ln + k - 1$); es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_{l+1} con la valoración de verdad correspondiente; y
    - $n + (l+1)n + k + 1 - m$ objetos #.

  Por lo tanto, la configuración $\mathcal{C}_{2n+2np+(l+1)n+k}$ da lugar a la configuración $\mathcal{C}_{2n+2np+(l+1)n+k+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
T_k x_{1,1,2n+2np+(l+1)n+k}\,[\ \ ]_2 \to [\,c_{l+1,0}\,]_2 \\
T_k \overline{x}_{1,1,2n+2np+(l+1)n+k}\,[\ \ ]_2 \to [\,\#\,]_2 \\
T_k x^*_{1,1,2n+2np+(l+1)n+k}\,[\ \ ]_2 \to [\,\#\,]_2 \\
F_k x_{1,1,2n+2np+(l+1)n+k}\,[\ \ ]_2 \to [\,\#\,]_2 \\
F_k \overline{x}_{1,1,2n+2np+(l+1)n+k}\,[\ \ ]_2 \to [\,c_{l+1,0}\,]_2 \\
F_k x^*_{1,1,2n+2np+(l+1)n+k}\,[\ \ ]_2 \to [\,\#\,]_2 \\
[\,\alpha_{2n+2np+(l+1)n+k} \to \alpha_{2n+2np+(l+1)n+k+1}\,]_0 \\
[\,\beta_{2n+2np+(l+1)n+k} \to \beta_{2n+2np+(l+1)n+k+1}\,]_0 \\
[\,\gamma_{2n+2np+(l+1)n+k} \to \gamma_{2n+2np+(l+1)n+k+1}\,]_1 \\
\left.\begin{array}{l}
[\,x_{i,j,2n+2np+(l+1)n+k} \to x_{i,j,2n+2np+(l+1)n+k+1}\,]_1 \\
[\,\overline{x}_{i,j,2n+2np+(l+1)n+k} \to \overline{x}_{i,j,2n+2np+(l+1)n+k+1}\,]_1 \\
[\,x^*_{i,j,2n+2np+(l+1)n+k} \to x^*_{i,j,2n+2np+(l+1)n+k+1}\,]_1
\end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n \\ 1 \leq j \leq p \end{array}
\end{array}
$$

$[\, c_{j,t} \to c_{j,t+1} \,]_2$ para $1 \leq j \leq l+1, 0 \leq t \leq ln+k-1$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2n+2np+(l+1)n+k+1}(0) = \{\alpha_{2n+2np+(l+1)n+k+1}, \beta_{2n+2np+(l+1)n+k+1}\}$
- En la configuración $\mathcal{C}_{2n+2np+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
  - los $(n-(k+1))$-ésimos últimos elementos de $cod_{2n+2np+(l+1)n+k+1}(\varphi)_{l+1}^{l+1}$,
  - el multiconjunto de entrada $cod_{2n+2np+(l+1)n+k+1}(\varphi)_{l+1}^{p}$,
  - un objeto $\gamma_{2n+2np+(l+1)n+k+1}$;
  - $p-l$ copias de objetos T_i o F_i, $k+2 \leq i \leq n$, $p-l-1$ copias si $1 \leq i \leq k+1$, correspondiendo a la valoración de verdad asignada a la rama.
- En la configuración $\mathcal{C}_{2n+2np+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
  - m objetos $c_{j,t}$ $(1 \leq j \leq l+1,\ 0 \leq t \leq ln+k)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_{l+1} con la valoración de verdad correspondiente; y
  - $n+(l+1)n+k+1-m$ objetos $\#$.

- Para probar (b) basta darse cuenta que, por una parte, por (a) la configuración $\mathcal{C}_{2n+3np-1}$ [6] es válido:

  - $\mathcal{C}_{2n+3np-1}(0) = \{\alpha_{2n+3np-1}, \beta_{2n+3np-1}\}$
  - En la configuración $\mathcal{C}_{2n+3np-1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - el último elemento de $cod_{2n+3np-1}(\varphi)_p^p$;
    - un objeto $\gamma_{2n+3np-1}$; y
    - un objeto T_n o F_n correspondiendo a la valoración de verdad asignada a la rama.
  - En la configuración $\mathcal{C}_{2n+3np-1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - m objetos $c_{j,t}$ $(1 \leq j \leq p,\ 0 \leq t \leq np-2)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_p con la valoración de verdad correspondiente; y

[6] Obsérvese que $2n+3np-1 = 2n+3n(p-1)+(n-1)$.

- $n + np - 1 - m$ objetos $\#$.

Por tanto, la configuración $\mathcal{C}_{2n+3np-1}$ da lugar a la configuración $\mathcal{C}_{2n+3np}$ por la aplicación de las reglas:

$T_n x_{n,p,2n+3np-1} [\ \ ]_2 \to [\, c_{p,0} \,]_2$
$T_n \overline{x}_{n,p,2n+3np-1} [\ \ ]_2 \to [\, \# \,]_2$
$T_n x^*_{n,p,2n+3np-1} [\ \ ]_2 \to [\, \# \,]_2$
$F_n x_{n,p,2n+3np-1} [\ \ ]_2 \to [\, \# \,]_2$
$F_n \overline{x}_{n,p,2n+3np-1} [\ \ ]_2 \to [\, c_{p,0} \,]_2$
$F_n x^*_{n,p,2n+3np-1} [\ \ ]_2 \to [\, \# \,]_2$

$[\, \alpha_{2n+2np+(l+1)n+k} \to \alpha_{2n+2np+(l+1)n+k+1} \,]_0$
$[\, \beta_{2n+2np+(l+1)n+k} \to \beta_{2n+2np+(l+1)n+k+1} \,]_0$
$[\, \gamma_{2n+2np+(l+1)n+k} \to \gamma_{2n+2np+(l+1)n+k+1} \,]_1$

$$\left.\begin{array}{l} [\, x_{i,j,2n+2np+(l+1)n+k} \to x_{i,j,2n+2np+(l+1)n+k+1} \,]_1 \\ [\, \overline{x}_{i,j,2n+2np+(l+1)n+k} \to \overline{x}_{i,j,2n+2np+(l+1)n+k+1} \,]_1 \\ [\, x^*_{i,j,2n+2np+(l+1)n+k} \to x^*_{i,j,2n+2np+(l+1)n+k+1} \,]_1 \end{array}\right\} \text{para } \begin{array}{c} 1 \leq i \leq n \\ 1 \leq j \leq p \end{array}$$

$[\, c_{j,t} \to c_{j,t+1} \,]_2$ para $1 \leq j \leq l+1, 0 \leq t \leq np-2$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2n+3np}(0) = \{\alpha_{2n+3np}, \beta_{2n+3np}\}$
- En la configuración $\mathcal{C}_{2n+3np}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene un objeto γ_{2n+3np}.
- En la configuración $\mathcal{C}_{2n+3np}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
  - m objetos $c_{j,t}$ $(1 \leq j \leq p,\ 0 \leq t \leq np-1)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_p con la valoración de verdad correspondiente; y
  - $n + np - m$ objetos $\#$.

□

Segunda fase de chequeo

En esta fase, que empieza en la configuración $\mathcal{C}_{2n+3np}$, se podrá ver si hay alguna valoración de verdad que haga verdadera la fórmula φ, usando las reglas de 3.1. Dividiremos esta fase en dos partes. La primera estará destinada a enviar «hacia fuera» todos los objetos c_j, para $1 \leq j \leq p$ para prepararlos para la siguiente parte.

Sea $k = ln + i$ $(0 \leq l \leq p-1, 1 \leq i \leq n)$, así podremos referirnos a cada cláusula $(l+1)$ cuando estemos viendo la verificación. Sea $m = \sum_{j=1}^{p} m_j$, siendo m_j el número de objetos $c_{j,k}$ en cada membrana etiquetada por 2 en la configuración $\mathcal{C}_{2n+3np}$. En esta parte, no podemos asegurar cuantos objetos $c_{l+1,k}$ están presentes en cada membrana cuando $i \neq 0$ [7], así que al no poder estar seguros, diremos que habrá $\widetilde{m}_j$ objetos en una membrana etiquetada por 2 (recordemos que $\widetilde{m}_j$ es siempre menor que o igual a m_j). Ignoraremos los objetos # dado que no tienen efecto a lo largo de la computación.

Proposición 7.5. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a) *Para cada* k $(1 \leq k \leq np-1)$ *en la configuración* $\mathcal{C}_{2n+3np+k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2n+3np+k}(0) = \{\alpha_{2n+3np+k}, \beta_{2n+3np+k}\}$
- *Hay* 2^n *membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *un objeto* $\gamma_{2n+3np+k}$*; y*
  - m_j *objetos* c_j *para* $1 \leq j \leq l$ *y* $m_{l+1} - \widetilde{m}_{l+1}$ *objetos* c_{l+1}
- *Hay* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene* $\widetilde{m}_{l+1}$ *objetos* $c_{l+1,t}$ $((p-1)n+1 \leq t \leq np-1)$ *y* m_j *objetos* $c_{j,t}$ $(l+2 \leq j \leq p, ln+i \leq t \leq np-1)$

(b) $\mathcal{C}_{2n+4np}(0) = \{\alpha_{2n+4np}, \beta_{2n+4np}\}$*, hay* 2^n *membranas etiquetadas por 1, tales que cada una de ellas contiene* m *objetos* c_j $(1 \leq j \leq p)$ *y un objeto* γ_{2n+4np}*; y* 2^n *membranas vacías etiquetadas por 2.*

Demostración. (a) será demostrado por inducción sobre k

- El caso base $k = 1$ es trivial dado que: en la configuración $\mathcal{C}_{2n+3np}$ tenemos que: $\mathcal{C}_{2n+3np}(0) = \{\alpha_{2n+3np}, \beta_{2n+3np}\}$ y hay 2^n membranas etiquetadas por 1 que contienen un objeto γ_{2n+3np}; y 2^n membranas etiquetadas por 2 que contienen m objetos $c_{j,t}$ $(1 \leq j \leq k, 0 \leq t \leq np-1)$. Por tanto, la configuración $\mathcal{C}_{2n+3np}$ da lugar a la configuración $\mathcal{C}_{2n+3np+1}$ por la aplicación de las reglas:

[7] Esto es debido a que los objetos $c_{j,k}$ no tienen por qué haber sido creados consecutivamente.

$[\,\alpha_{2n+3np} \to \alpha_{2n+3np+1}\,]_0$
$[\,\beta_{2n+3np} \to \beta_{2n+3np+1}\,]_0$
$[\,\gamma_{2n+3np} \to \gamma_{2n+3np+1}\,]_1$
$[\,c_{j,t} \to c_{j,t+1}\,]_2$, para$1 \leq j \leq p$, $0 \leq k \leq np-2$
$[\,c_{1,np-1}\,]_2 \to c_1\,[\quad]_2$

Por lo tanto, $\mathcal{C}_{2n+3np+1}(0) = \{\alpha_{2n+3np+1}, \beta_{2n+3np+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen un objeto $\gamma_{2n+3np+1}$ y $m_1 - \widetilde{m}_1$ objetos c_1 [8]; y 2^n membranas etiquetadas por 2 que contienen $\widetilde{m}_1$ objetos c_1 y m_j objetos c_j $(2 \leq j \leq p)$. Por lo tanto, el resultado es válido para $k = 1$.

- Suponiendo, por inducción, que el resultado es cierto para k $(1 \leq k \leq np-1)$
  - $\mathcal{C}_{2n+3np+k}(0) = \{\alpha_{2n+3np+k}, \beta_{2n+3np+k}\}$
  - En la configuración $\mathcal{C}_{2n+3np+k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - un objeto $\gamma_{2n+3np+k}$; y
    - m_j objetos c_j para $1 \leq j \leq l$ y $m_{l+1} - \widetilde{m}_{l+1}$ objetos c_{l+1}.
  - En la configuración $\mathcal{C}_{2n+3np+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene $\widetilde{m}_{l+1}$ objetos $c_{l+1,t}$ $((p-1)n+1 \leq t \leq np-1)$ y m_j objetos $c_{j,t}$ $(l+2 \leq j \leq p, ln+i \leq t \leq np-1)$.
    Por tanto, la configuración $\mathcal{C}_{2n+3np+k}$ da lugar a la configuración $\mathcal{C}_{2n+3np+k}$ por la aplicación de las reglas:
    $[\,\alpha_{2n+3np+k} \to \alpha_{2n+3np+k+1}\,]_0$
    $[\,\beta_{2n+3np+k} \to \beta_{2n+3np+k+1}\,]_0$
    $[\,\gamma_{2n+3np+k} \to \gamma_{2n+3np+k+1}\,]_1$
    $[\,c_{j,t} \to c_{j,t+1}\,]_2$, para$l+1 \leq j \leq p$, $0 \leq k \leq np-2$
    $[\,c_{l+1,np-1}\,]_2 \to c_1\,[\quad]_2$
    Por lo tanto, lo siguiente es válido
  - $\mathcal{C}_{2n+3np+k+1}(0) = \{\alpha_{2n+3np+k+1}, \beta_{2n+3np+k+1}\}$
  - En la configuración $\mathcal{C}_{2n+3np+k+1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
    - un objeto $\gamma_{2n+3np+k+1}$; y

[8] Es decir, si la valoración de verdad de la variable 1 hace verdadera la cláusula 1, entonces un objeto $c_{1,0}$ fue creado en el $(2n+2np+1)$-ésimo paso, y será enviado a la membrana etiquetada por 1 correspondiente. Así que, $m_1 - \widetilde{m}_1$ puede ser 0 o 1 en este paso.

- m_j objetos c_j para $1 \leq j \leq l$ y $m_{l+1} - \widetilde{m}_{l+1}$ objetos c_{l+1}.

- En la configuración $\mathcal{C}_{2n+3np+k+1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene $\widetilde{m}_{l+1}$ objetos $c_{l+1,t+1}$ $((p-1)n+1 \leq t \leq np-1)$ y m_j objetos $c_{j,t+1}$ $(l+2 \leq j \leq p, ln+i \leq t \leq np-1)$.
  Por lo tanto, el resultado es válido para $k+1$.

- Para probar (b) basta darse cuenta que, por una parte, por (a) la configuración $\mathcal{C}_{2n+4np-1}$ es válido:

  - $\mathcal{C}_{2n+4np-1}(0) = \{\alpha_{2n+4np-1}, \beta_{2n+4np-1}\}$
  - En la configuración $\mathcal{C}_{2n+4np-1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
    - un objeto $\gamma_{2n+4np-1}$; y
    - m_j objetos c_j para $1 \leq j \leq p-1$ y $m_p - \widetilde{m}_p$ [9] objetos c_p.
  - En la configuración $\mathcal{C}_{2n+4np-1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene $\widetilde{m}_p$ objetos $c_{p,np}$.
    Por tanto, la configuración $\mathcal{C}_{2n+4np-1}$ da lugar a la configuración $\mathcal{C}_{2n+4np}$ por la aplicación de las reglas:
    $[\,\alpha_{2n+4np-1} \to \alpha_{2n+4np}\,]_0$
    $[\,\beta_{2n+4np-1} \to \beta_{2n+4np}\,]_0$
    $[\,\gamma_{2n+4np-1} \to \gamma_{2n+4np}\,]_1$
    $[\,c_{p,np}\,]_2 \to c_p\,[\quad]_2$
    Por lo tanto, tenemos que $\mathcal{C}_{2n+4np}(0) = \{\alpha_{2n+4np}, \beta_{2n+4np}\}$, y hay 2^n membranas etiquetadas por 1 que contienen un objeto γ_{2n+4np} y m objetos c_j $(1 \leq j \leq p)$; y hay 2^n membranas vacías etiquetadas por 2.

□

Cuando los objetos c_j se mueven a las membranas etiquetadas por 1, podemos empezar a chequear si todas las cláusulas de la fórmula φ son satisfechas por alguna valoración de verdad. Como estamos usando los objetos c_j para denotar si la cláusula C_j es satisfecha por alguna variable, es posible que algún objeto c_j puede faltar; es decir, que para algún $j, 1 \leq j \leq p$, c_j no aparece en ninguna membrana etiquetada por 1 en $\mathcal{C}_{2n+4np}$. Sea $\widetilde{j}$ el índice j [10] de dicha cláusula. Se hará en $2p$ pasos de computación.

[9] En este caso, $\widetilde{m}_p$ puede tomar dos valores: 0 o 1.

[10] Si $\widehat{j}$ no está definido, suponemos que es igual a $p+1$.

Proposición 7.6. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a_0) *Para cada* $2k+1$ *(*$0 \leq k \leq p-1$*) en la configuración* $\mathcal{C}_{2n+4np+2k+1}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2n+4np+2k+1}(0) = \{\alpha_{2n+4np+2k+1}, \beta_{2n+4np+2k+1}\}$
- *Hay* 2^n *membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *un objeto* γ_{2n+4np} *o* $d_{\widetilde{j-1}}$ *(respectivamente, un objeto* d_k*) si la valoración de verdad correspondiente no hace verdadera (respectivamente, hace verdadera) la cláusula* C_1 *o* C_j *(*$2 \leq j \leq p$*) (respectivamente, las* k *primeras cláusulas); y*
  - $m_j - 1$ *objetos* c_j *para* $1 \leq j \leq min(\widetilde{j}, k+1)$ *y* m_j *objetos* c_j *para* $min(\widetilde{j}, k+2) \leq j \leq p$.
- *Hay* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene un objeto* d_{k+1} *si y sólo si la valoración de verdad asociada a dicha rama hace verdaderas las* $k+1$ *primeras cláusulas.*

(a_1) *Para cada* $2k$ *(*$1 \leq k \leq p-1$*) en la configuración* $\mathcal{C}_{2n+4np+2k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2n+4np+2k}(0) = \{\alpha_{2n+4np+2k}, \beta_{2n+4np+2k}\}$
- *Hay* 2^n *membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *un objeto* γ_{2n+4np} *o* $d_{\widetilde{j-1}}$ *si la valoración de verdad correspondiente no hace verdadera la cláusula* C_1 *o la cláusula* C_j *(*$2 \leq j \leq p$*); y*
  - $m_j - 1$ *objetos* c_j *para* $1 \leq j \leq min(\widetilde{j}, k)$ *y* m_j *objetos* c_j *para* $min(\widetilde{j}, k+1) \leq j \leq p$.
- *hay* 2^n *membranas vacías etiquetadas por 2.*

(b) $\mathcal{C}_{2n+4np+2p}(0) = \{\alpha_{2n+4np+2p}, \beta_{2n+4np+2p}\}$, *y en la configuración* $\mathcal{C}_{2n+4np+2p}$ *hay* 2^n *membranas etiquetadas por 1, tales que cada una de ellas contiene un objeto* d_p *si y sólo si la valoración de verdad correspondiente la fórmula* φ *(*$d_{\widetilde{j-1}}$ *de lo contrario),* $m_j - 1$ *objetos* c_j *para* $1 \leq j \leq min(\widetilde{j}, p+1)$ *y* m_j *objetos* c_j *para* $min(\widetilde{j}, p+1) \leq j \leq p$*; y* 2^n *membranas vacías etiquetadas por 2.*

Demostración. (a) será demostrado por inducción sobre k

- El caso base $k = 1$ es trivial dado que:

  (a_0) en la configuración $\mathcal{C}_{2n+4np}$ tenemos que: $\mathcal{C}_{2n+4np}(0) = \{\alpha_{2n+4np}, \beta_{2n+4np}\}$ y hay 2^n membranas etiquetadas por 1 que contienen un objeto γ_{2n+4np} y m objetos c_j $(1 \leq j \leq p)$; y hay 2^n membranas vacías etiquetadas por 2. Por tanto, la configuración $\mathcal{C}_{2n+4np}$ da lugar a la configuración $\mathcal{C}_{2n+4np+1}$ por la aplicación de las reglas:

  $$[\,\alpha_{2n+4np} \to \alpha_{2n+4np+1}\,]_0$$
  $$[\,\beta_{2n+4np} \to \beta_{2n+4np+1}\,]_0$$
  $$\gamma_{4np+2n} c_1\,[\quad]_2 \to [\,d_1\,]_2$$

  (a_1) en la configuración $\mathcal{C}_{2n+4np+1}(0) = \{\alpha_{2n+4np+1}, \beta_{2n+4np+1}\}$ y hay 2^n membranas etiquetadas por 1 que contienen un objeto γ_{2n+4np} si y sólo si no había objetos c_1 en la configuración $\mathcal{C}_{2n+4np}$, $m_1 - 1$ (respectivamente, m_1) objetos c_1 si había algún objeto c_j en dicha membrana en la configuración anterior (respectivamente, m_1) y m_j objetos c_j para $2 \leq j \leq p$; y 2^n membranas etiquetadas por 2 que contienen un objeto d_1 si y sólo si había al menos un objeto c_1 en dicha membrana etiquetada por 1 en la configuración $\mathcal{C}_{2n+4np}$. Así, la configuración $\mathcal{C}_{2n+4np+1}$ da lugar a la configuración $\mathcal{C}_{2n+4np+2}$ por la aplicación de las reglas:

  $$[\,\alpha_{2n+4np+1} \to \alpha_{2n+4np+2}\,]_0$$
  $$[\,\beta_{2n+4np+1} \to \beta_{2n+4np+2}\,]_0$$
  $$[\,d_1\,]_2 \to d_1\,[\quad]_2$$

  Por lo tanto, $\mathcal{C}_{2n+4np+2}(0) = \{\alpha_{2n+4np+2}, \beta_{2n+4np+2}\}$, y hay 2^n membranas etiquetadas por 1 que contienen un objeto d_1 (respectivamente, γ_{2n+4np}) si la valoración de verdad correspondiente hace verdadera (respectivamente, no hace verdadera) la cláusula C_1, $m_1 - 1$ (respectivamente, m_1) objetos c_1 y m_j objetos c_j para $1 \leq j \leq p$; y hay 2^n membranas vacías etiquetadas por 2. Por lo tanto, el resultado es válido para $k = 1$.

- Suponiendo, por inducción, que el resultado es cierto para k $(0 \leq k \leq p - 1)$

  - $\mathcal{C}_{2n+4np+2k}(0) = \{\alpha_{2n+4np+2k}, \beta_{2n+4np+2k}\}$
  - En la configuración $\mathcal{C}_{2n+4np+2k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene

  - un objeto γ_{2n+4np} o $d_{\widetilde{j-1}}$ (respectivamente, un objeto d_k) si la valoración de verdad correspondiente no hace verdadera (respectivamente, hace verdadera) la cláusula C_1 o C_j $(2 \leq j \leq p)$ (respectivamente, las k primeras cláusulas); y
  - $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, k+1)$ y m_j objetos c_j para $min(\widetilde{j}, k+2) \leq j \leq p$.

- En la configuración $\mathcal{C}_{2n+4np+2k}$ hay 2^n membranas vacías etiquetadas por 2.

  Por tanto, la configuración $\mathcal{C}_{2n+4np+2k}$ da lugar a la configuración $\mathcal{C}_{2n+4np+2k+1}$ por la aplicación de las reglas:

  $[\, \alpha_{2n+4np+2k} \to \alpha_{2n+4np+2k+1} \,]_0$
  $[\, \beta_{2n+4np+2k} \to \beta_{2n+4np+2k+1} \,]_0$
  $d_k c_{k+1} [\quad]_2 \to [\, d_{k+1} \,]_2$

  Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2n+4np+2k+1}(0) = \{\alpha_{2n+4np+2k+1}, \beta_{2n+4np+2k+1}\}$
- En la configuración $\mathcal{C}_{2n+4np+2k+1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene

  - un objeto γ_{2n+4np} o $d_{\widetilde{j-1}}$ si la valoración de verdad correspondiente no hace verdadera la cláusula C_1 o la cláusula C_j $(2 \leq j \leq p)$; y
  - $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, k)$ y m_j objetos c_j para $min(\widetilde{j}, k+1) \leq j \leq p$.

- En la configuración $\mathcal{C}_{2n+4np+2k+1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene un objeto d_{k+1} si y sólo si la valoración de verdad correspondiente hace verdaderas las $k+1$ primeras cláusulas.

  Por tanto, la configuración $\mathcal{C}_{2n+4np+2k+1}$ da lugar a la configuración $\mathcal{C}_{2n+4np+2k+2}$ por la aplicación de las reglas:

  $[\, \alpha_{2n+4np+2k+1} \to \alpha_{2n+4np+2k+2} \,]_0$
  $[\, \beta_{2n+4np+2k+1} \to \beta_{2n+4np+2k+2} \,]_0$
  $[\, d_{k+1} \,]_2 \to d_{k+1} [\quad]_2$

  Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2n+4np+2k+2}(0) = \{\alpha_{2n+4np+2k+2}, \beta_{2n+4np+2k+2}\}$
- En la configuración $\mathcal{C}_{2n+4np+2k+2}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene

- un objeto γ_{2n+4np} o $d_{\widetilde{j-1}}$ (respectivamente, un objeto d_{k+1}) si la valoración de verdad correspondiente no hace verdadera (respectivamente, hace verdadera) la cláusula C_1 o C_j $(2 \leq j \leq p)$ (respectivamente, las $k+1$ primeras cláusulas); y
- $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, k+2)$ y m_j objetos c_j para $min(\widetilde{j}, k+3) \leq j \leq p$.

- En la configuración $\mathcal{C}_{2n+4np+2k+2}$ hay 2^n membranas vacías etiquetadas por 2.

  Por lo tanto, el resultado es válido para $k+1$.

- Para probar (b) basta con observar que, por una parte, de (a), la configuración $\mathcal{C}_{2n+4np+2p-1}$ tiene que:

  - $\mathcal{C}_{2n+4np+2p-1}(0) = \{\alpha_{2n+4np+2p-1}, \beta_{2n+4np+2p-1}\}$
  - En la configuración $\mathcal{C}_{2n+4np+2p-1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
    - un objeto γ_{2n+4np} o $d_{\widetilde{j-1}}$ si la valoración de verdad correspondiente no hace verdadera la cláusula C_1 o la cláusula C_j $(2 \leq j \leq p)$; y
    - $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, p)$ y m_j objetos c_j para $min(\widetilde{j}, p+1) \leq j \leq p$.
  - En la configuración $\mathcal{C}_{2n+4np+2p-1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene un objeto d_p si y sólo si la valoración de verdad correspondiente la fórmula φ.

    Por tanto, la configuración $\mathcal{C}_{2n+4np+2p-1}$ da lugar a la configuración $\mathcal{C}_{2n+4np+2p}$ por la aplicación de las reglas:

    $[\,\alpha_{2n+4np+2p-1} \to \alpha_{2n+4np+2p}\,]_0$
    $[\,\beta_{2n+4np+2p-1} \to \beta_{2n+4np+2p}\,]_0$
    $[\,d_p\,]_2 \to d_p\,[\quad]_2$

    Por tanto, tenemos que $\mathcal{C}_{2n+4np+2p}(0) = \{\alpha_{2n+4np+2p}, \beta_{2n+4np+2p}\}$, y hay 2^n membranas etiquetadas por 1 que contienen un objeto γ_{2n+4np} o $d_{\widetilde{j-1}}$ (respectivamente, un objeto d_p) si la valoración de verdad correspondiente no hace verdadera (respectivamente, hace verdadera) la cláusula C_1 o C_j $(2 \leq j \leq p)$ (respectivamente, la fórmula φ), $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, p+1)$ y m_j objetos c_j para $min(\widetilde{j}, p+1) \leq j \leq p$; y hay 2^n membranas vacías etiquetadas por 2.

□

Fase de salida

La fase de salida comienza en el paso $2n+4np+2p$, y tarda 4 pasos en caso de devolver una respuesta afirmativa y 5 pasos en caso de tener una respuesta negativa. Las reglas de 4.1 son las disparadas en esta fase.

- *Respuesta afirmativa*: En este caso, en la configuración $\mathcal{C}_{2n+4np+2p}$, en alguna membrana etiquetada por 1 habrá un objeto d_p. Por la aplicación de la regla $[\, d_p\,]_1 \to d_p\,[\quad]_1$ (al mismo tiempo que se disparan $[\, \alpha_{2n+4np+2p} \to \alpha_{2n+4np+2p+1}\,]_0$ y $[\, \beta_{2n+4np+2p} \to \beta_{2n+4np+2p+1}\,]_0$), el objeto d_p es enviado a la membrana etiquetada por 0. En el siguiente paso, por la aplicación de las reglas $\alpha_{4np+2n+2p+1} d_p\,[\quad]_1 \to [\,\texttt{yes}\,]_1$ y $[\, \beta_{2n+4np+2p+1} \to \beta_{2n+4np+2p+2}\,]_0$, un objeto `yes` es producido en alguna membrana etiquetada por 1 (sólo en una de las membranas, elegida de manera no determinista). En el siguiente paso, el objeto `yes` será enviado de nuevo a la membrana etiquetada por 0 en la configuración $\mathcal{C}_{2n+4np+2p+3}$ por la aplicación de la regla $[\,\texttt{yes}\,]_1 \to \texttt{yes}\,[\quad]_1$. Obsérvese que el objeto $\beta_{2n+4np+2p+2}$ no puede interactuar con ningún objeto α. Finalmente, en el paso $2n + 4np + 2p + 4$-ésimo, el objeto `yes` es enviado al entorno mediante la regla $[\,\texttt{yes}\,]_0 \to \texttt{yes}\,[\quad]_0$ y la computación para.

- *Respuesta negativa*: En este caso, en la configuración $\mathcal{C}_{2n+4np+2p}$, no habrá ninguna membrana etiquetada por 1 tal que contenga un objeto d_p, así que las reglas ejecutadas serán las reglas $[\, \alpha_{2n+4np+2p} \to \alpha_{2n+4np+2p+1}\,]_0$ y $[\, \beta_{2n+4np+2p} \to \beta_{2n+4np+2p+1}\,]_0$. La regla $[\, \beta_{2n+4np+2p+1} \to \beta_{2n+4np+2p+2}\,]_0$ es ejecutada en el siguiente paso de computación. por lo tanto, en la membrana piel en la configuración $\mathcal{C}_{2n+4np+2p+2}$ tenemos una copia del objeto $\alpha_{2n+4np+2p+1}$ y una copia del objeto $\beta_{2n+4np+2p+2}$. Por la aplicación de la regla $\alpha_{4np+2n+2p+1}\beta_{4np+2n+2p+2}\,[\quad]_1 \to [\,\texttt{no}\,]_1$, un objeto `no` es producido en una membrana etiquetada por 1 (elegida de manera no determinista). En el siguiente paso, el objeto `no` se enviará al entorno por la aplicación de la regla $[\,\texttt{no}\,]_1 \to \texttt{no}\,[\quad]_1$ que, finalmente, en el paso $2n+4np+2p+5$ será enviado al entorno mediante la regla $[\,\texttt{no}\,]_0 \to \texttt{no}\,[\quad]_0$, y la computación para.

7.1.1.2. Resultado obtenido

Teorema 7.1. $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{DAM}^0(+e_s, mcmp_{in}, -d, +n)}$.

Demostración. La familia $\mathbf{\Pi}$ de sistemas P descrita verifica lo siguiente:

(a) La familia $\mathbf{\Pi}$ es polinomialmente uniforme por máquinas de Turing, dado que para cada $n, p \in \mathbb{N}$, las reglas de $\Pi(\langle n, p\rangle)$ de la familia son recursivas recursivamente de $n, p \in \mathbb{N}$, y la cantidad de recursos necesarios para construir un elemento de dicha familia es de orden polinomial con respecto a n y p, como se muestra a continuación:

- Tamaño del alfabeto: $\frac{15n^2p^2}{2}+6n^2p+3n^2+2np^2+\frac{35np}{2}+8n+7p+9 \in \Theta(n^2p^2)$.
- Número inicial de membranas: $3 \in \Theta(1)$.
- Número inicial de objetos en membranas: $3np + n + 3 \in \Theta(np)$.
- Número de reglas: $\frac{15n^2p^2}{2}+8n^2p+4n^2+\frac{41np}{2}+5n+5p+11 \in \Theta(n^2p^2)$.
- Número máximo de objetos envueltos en cualquier regla: $3 \in \Theta(1)$.

(b) La familia $\mathbf{\Pi}$ está polinomialmente acotada con respecto a $(\texttt{SAT}, cod, s)$: de hecho, para cada instancia φ de SAT, cualquier computación del sistema $\mathbf{\Pi}(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$ tarda, a lo sumo, $2n + 4np + 2p + 5$ pasos de computación.

(c) La familia $\mathbf{\Pi}$ es adecuada con respecto a $(\texttt{SAT}, cod, s)$: de hecho, para cada instancia φ de SAT, si la computación de $\mathbf{\Pi}(s(\varphi)) + cod(\varphi)$ es una computación de aceptación, entonces φ es satisfactible.

(d) La familia $\mathbf{\Pi}$ es completa con respecto a $(\texttt{SAT}, cod, s)$: de hecho, para cada instancia φ de SAT tal que φ sea satisfactible, cualquier computación de $\mathbf{\Pi}(s(\varphi)) + cod(\varphi)$ será una computación de aceptación.

Por lo tanto, la familia $\mathbf{\Pi}$ de sistemas P descrita anteriormente resuelve el problema SAT en tiempo polinomial de manera uniforme.

□

Corolario 7.1. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{DAM}^0(+e_s,mcmp_{in},-d,+n)}$.

Demostración. Basta notar que el problema SAT es un problema **NP**-completo, $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{DAM}^0(+e_s,mcmp_{in},-d,+n)}$, y la clase de complejidad $\mathbf{PMC}_{\mathcal{DAM}^0(+e_s,mcmp_{in},-d,+n)}$ es cerrada bajo reducibilidad en tiempo polinomial y bajo complementario.

□

7.1.2. Una solución eficiente al problema SAT mediante sistemas de $\mathcal{DAM}^0(+e_s, mcmp_{out}, -d, +n)$

A continuación, se describe una familia de sistemas de membranas de $\mathcal{DAM}^0(+e_s, mcmp_{in}, -d, +n)$ que resuelve el problema SAT en tiempo polinomial y de manera uniforme.

Para cada $n, p \in \mathbb{N}$, consideramos el sistema P reconocedor

$$\Pi(\langle n, p\rangle) = (\Gamma, \Sigma, H, \mu, \mathcal{M}_0, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, i_{in}, i_{out})$$

de $\mathcal{DAM}^0(+e_s, mcmp_{out}, -d, +n)$, definido como sigue:

1. Alfabeto de trabajo Γ:

   $\Sigma \cup \{\texttt{yes}, \texttt{no}, \#\} \cup \{a_{i,k} \mid 1 \leq i \leq n, 1 \leq k \leq 2i-1\} \cup$
   $\{\alpha_k \mid 0 \leq k \leq 4np+n+2p\} \cup \{\beta_k \mid 0 \leq k \leq 4np+n+2p+1\} \cup$
   $\{\gamma_k \mid 0 \leq k \leq 4np+n\} \cup$
   $\{t_{i,k}, f_{i,k} \mid 1 \leq i \leq n, 2i-1 \leq k \leq 2n+2p-1\} \cup$
   $\{T_i, F_i \mid 1 \leq i \leq n\} \cup \{c_j \mid 1 \leq j \leq p\} \cup$
   $\{c_{j,k} \mid 1 \leq j \leq p, 0 \leq k \leq np-1\} \cup \{d_j \mid 1 \leq j \leq p\} \cup$
   $\{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 1 \leq i \leq n, 1 \leq j \leq p,$
   $1 \leq k \leq n + 2np + n(j-1) + (i-1)\}$

2. Alfabeto de entrada Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,j,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$;

3. $H = \{0, 1, 2\}$;

4. $\mu = [\,[\,[\;\;]_2\,]_1\,]_0$; es decir, $\mu = (V, E)$ donde $V = \{0, 1, 2\}$ y $E = \{(0,1)(1,2)\}$.

5. $\mathcal{M}_0 = \{\alpha_0, \beta_0\}$, $\mathcal{M}_1 = \emptyset$, $\mathcal{M}_2 = \{\gamma_0\} \cup \{a_{i,1}, T^p_i, F^p_i \mid 1 \leq i \leq n\}$.

6. El conjunto $\mathcal{R}$ tiene las siguientes reglas:

   1.1 Contadores para sincronizar la salida del sistema.

   $[\,\alpha_k \to \alpha_{k+1}\,]_0$, para $0 \leq k \leq 4np+n+2p-1$

   $[\,\beta_k \to \beta_{k+1}\,]_0$, para $0 \leq k \leq 4np+n+2p$

   $[\,\gamma_k \to \gamma_{k+1}\,]_2$, para $0 \leq k \leq 4np+n-1$

   1.2 Reglas para generar 2^n membranas etiquetadas por 1 y 2^n membranas etiquetadas por 2 (estas codificando todas las valoraciones de verdad posibles para n variables de la fórmula φ

$[\, a_{i,2i-1} \,]_2 \to [\, t_{i,i} \,]_2 [\, f_{i,i} \,]_2$, para $1 \leq i \leq n$

$[\, a_{i,j} \to a_{i,j+1} \,]_2$, para $2 \leq i \leq n, 1 \leq j \leq 2i-2$

$[\, [\,]_2 [\,]_2 \,]_1 \to [\, [\,]_2 \,]_1 [\, [\,]_2 \,]_1$

$\left.\begin{array}{l} [\, t_{i,j} \to t_{i,j+1} \,]_2 \\ [\, f_{i,j} \to f_{i,j+1} \,]_2 \end{array}\right\}$, para $1 \leq i \leq n, i \leq j \leq 2n-1$

1.3 Reglas para producir exactamente p copias de cada valoración de verdad dentro de las membranas etiquetada por 2.

$\left.\begin{array}{l} [\, t_{i,2jn} F_i \,]_2 \to t_{i,2jn+1} [\quad]_2 \\ [\, f_{i,2jn} T_i \,]_2 \to f_{i,2jn+1} [\quad]_2 \\ t_{i,(2j+1)n} [\quad]_2 \to [\, t_{i,(2j+1)n+1} \,]_2 \\ f_{i,(2j+1)n} [\quad]_2 \to [\, f_{i,(2j+1)n+1} \,]_2 \end{array}\right\}$, para$1 \leq i \leq n$, $1 \leq j \leq p-1$

$\left.\begin{array}{l} [\, t_{i,2np} F_i \,]_2 \to \# [\quad]_2 \\ [\, f_{i,2np} T_i \,]_2 \to \# [\quad]_2 \end{array}\right\}$, para$1 \leq i \leq n$

$\left.\begin{array}{l} [\, t_{i,(2j+1)n+k} \to t_{i,(2j+1)n+k+1} \,]_2 \\ [\, f_{i,(2j+1)n+k} \to f_{i,(2j+1)n+k+1} \,]_2 \\ [\, t_{i,2jn+k} \to t_{i,2jn+k+1} \,]_1 \\ [\, f_{i,2jn+k} \to f_{i,2jn+k+1} \,]_1 \end{array}\right\}$, para $\begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p-1, \\ 1 \leq k \leq n-1 \end{array}$

1.4 Reglas para preparar la fórmula para validar las cláusulas.

$\left.\begin{array}{l} [\, x_{i,j,k} \to x_{i,j,k+1} \,]_2 \\ [\, \overline{x}_{i,j,k} \to \overline{x}_{i,j,k+1} \,]_2 \\ [\, x^*_{i,j,k} \to x^*_{i,j,k+1} \,]_2 \end{array}\right\}$, para $\begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 0 \leq k \leq 2np + n + n(j-1) + (i-1) - 1 \end{array}$

2.1 Reglas para realizar el primer chequeo.

$\left.\begin{array}{l} [\, T_i x_{i,j,2np+n+n(j-1)+(i-1)} \,]_2 \to c_{j,0} [\quad]_2 \\ [\, T_i \overline{x}_{i,j,2np+n+n(j-1)+(i-1)} \,]_2 \to \# [\quad]_2 \\ [\, T_i x^*_{i,j,2np+n+n(j-1)+(i-1)} \,]_2 \to \# [\quad]_2 \\ [\, F_i x_{i,j,2np+n+n(j-1)+(i-1)} \,]_2 \to \# [\quad]_2 \\ [\, F_i \overline{x}_{i,j,2np+n+n(j-1)+(i-1)} \,]_2 \to c_{j,0} [\quad]_2 \\ [\, F_i x^*_{i,j,2np+n+n(j-1)+(i-1)} \,]_2 \to \# [\quad]_2 \end{array}\right\}$, para $\begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$

3.1 Reglas para realizar el segundo chequeo.

$[\, c_{j,k} \to c_{j,k+1} \,]_1$, para$1 \leq j \leq p$, $0 \leq k \leq np-2$

$c_{j,np-1} [\quad]_2 \to [\, c_j \,]_2$, para$1 \leq j \leq p$

$[\, \gamma_{4np+n} c_1 \,]_2 \to d_1 [\quad]_2$

$\left.\begin{array}{l} [\, d_j c_{j+1} \,]_2 \to d_{j+1} [\quad]_2 \\ d_j [\quad]_2 \to [\, d_j \,]_2 \end{array}\right\}$, para$1 \leq j \leq p-1$

4.1 Reglas para devolver la respuesta correcta.

$[\, d_p \,]_1 \to d_p [\quad]_1$

$$[\,\alpha_{4np+n+2p} d_p\,]_0 \rightarrow \texttt{yes}\,[\quad]_0$$
$$[\,\alpha_{4np+n+2p}\beta_{4np+n+2p+1}\,]_0 \rightarrow \texttt{no}\,[\quad]_0$$

7. La membrana de entrada es la membrana etiquetada por 2 ($i_{in} = 2$) y la región de salida es el entorno ($i_{out} = env$).

Sea $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_p, C_j = l_{j_1} \vee \cdots \vee l_{j_{r_j}}, 1 \leq j \leq p, l_{j_k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$ una instancia del problema SAT. Entonces la instancia φ será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$, en donde $s(\varphi) = \langle n, p\rangle$ y $cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$.

Denotamos por $cod_k(\varphi)$ el conjunto $cod(\varphi)$ cuando los elementos tienen su tercer subíndice igual a k. Denotamos por $cod_k(\varphi)^{k_2}_{k_1}$ al los elementos correspondientes a las cláusulas $C_{k_1}, \ldots, C_{k_2}$ de $cod_k(\varphi)$. De la misma forma, denotamos por $cod'_k(\varphi)^{k_2}_{k_1}$ a los elementos de $cod_k(\varphi)^{k_2}_{k_1}$ con una prima. Por último, denotamos con $cod^*(\varphi)^{k_2}_{k_1}$ a los elementos de $cod(\varphi)^{k_2}_{k_1}$ cuando eliminamos el tercer subíndice.

A continuación, daremos una descripción informal del funcionamiento del sistema $\Pi(s(\varphi)) + cod(\varphi)$.

La solución propuesta sigue un algoritmo de fuerza bruta en el marco de sistemas P reconocedores de membranas activas, con cooperación minimal en reglas de comunicación hacia fuera y reglas de división sólo para membranas elementales, y consiste en las siguientes fases:

- *Fase de generación*: usando las reglas de división, obtenemos todas las valoraciones de verdad posibles para las variables $\{x_1, \ldots, x_n\}$ asociadas a φ. En concreto, 2^n membranas etiquetadas por 2 y 2^n labelled by 1 are generated. Cada una de las primeras contiene una posible valoración de verdad. Esta fase tarda exactamente $n + 2np$ pasos de computación, siendo n el número de variables de φ.

- *Primera fase de chequeo*: en esta fase de verifica cuáles son las cláusulas de φ que satisface y cuáles no cada una de las valoraciones de verdad codificadas en cada membrana etiquetada por 2. Esta fase tarda np pasos, siendo n el número de variables y p el número de cláusulas de φ.

- *Segunda fase de chequeo*: en esta fase se verificará si alguna de las valoraciones de verdad satisface todas las cláusulas de φ, usando las reglas de 3.1. Esta fase tarda exactamente $np + 2p$ pasos de computación.

- *Fase de salida*: el sistema envía al entorno la respuesta adecuada usando las reglas de 4.1, y tarda 2 pasos de computación si la respuesta es afirmativa y 3 pasos en caso que la respuesta sea negativa.

7.1.2.1. Verificación formal

A continuación, se detallará una verificación exhaustiva de la solución.

Fase de generación

Mediante esta fase, todas las valoraciones de verdad para las variables asociadas a la fórmula Booleana φ serán generadas en las membranas etiquetadas por 2, por la aplicación de las reglas de 1.2 y 1.3. En los primeros $2n$ pasos, se generarán 2^n membranas etiquetadas por 1 y 2^n membranas etiquetadas por 2, alternando entre la división de membranas etiquetadas por 2 (en los pasos impares) y la división de membranas etiquetadas por 1 (en los pasos pares).

Proposición 7.7. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a_0) *Para cada* $2k$ *(*$0 \leq k \leq n-1$*) en la configuración* $\mathcal{C}_{2k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2k}(0) = \{\alpha_{2k}, \beta_{2k}\}$
- *hay* 2^k *membranas vacías etiquetadas por 1.*
- *Hay* 2^k *membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada* $cod_{2k}(\varphi)$*;*
  - *un objeto* γ_{2k}*; y*
  - *p copias de todos los* T_i *y* F_i*,* $1 \leq i \leq n$*.*
  - *los objetos* $a_{i,2k+1}$*,* $k+1 \leq i \leq n$*; y*
  - *un subconjunto diferente* $\{r_{1,j}, \ldots, r_{k,j}\}$*,* $k+1 \leq j \leq 2k$*, siendo* $r \in \{t, f\}$*.*

(a_1) *Para cada* $2k+1$ *(*$0 \leq j \leq n-1$*) en la configuración* $\mathcal{C}_{2k+1}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2k+1}(0) = \{\alpha_{2k+1}, \beta_{2k+1}\}$
- *hay* 2^k *membranas vacías etiquetadas por 1.*

- *Hay* 2^{k+1} *membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada* $cod_{2k+1}(\varphi)$*;*
  - *un objeto* γ_{2k+1}*; y*
  - p *copias de todos los* T_i *y* F_i*,* $1 \leq i \leq n$*.*
  - *los objetos* $a_{i,2(k+1)}$*,* $k+1 \leq i \leq n$*; y*
  - *un subconjunto diferente* $\{r_{1,j}, \ldots, r_{k+1,j}\}$*,* $k+1 \leq j \leq 2k+1$*, siendo* $r \in \{t, f\}$*.*

(*b*) $\mathcal{C}_{2n}(0) = \{\alpha_{2n}, \beta_{2n}\}$*, y en la configuración* $\mathcal{C}_{2n}$ *hay* 2^n *membranas vacías etiquetadas por 1; y* 2^n *membranas etiquetadas por 2, tales que cada una de ellas contiene el multiconjunto de entrada* $cod_{2n}(\varphi)$*,* p *copias de cada objeto* T_i *y* F_i *(*$1 \leq i \leq n$*), un objeto* γ_{2n} *y un subconjunto diferente de objetos* $r_{i,2n+1-i}$*,* $1 \leq i \leq n$*.*

Demostración. (a) será demostrado por inducción sobre k

- El caso base $k = 0$ es trivial dado que:

  (a_0) en la configuración inicial $\mathcal{C}_0$ tenemos que: $\mathcal{C}_0(0) = \{\alpha_0, \beta_0\}$ y hay una sola membrana etiquetada por 1 vacía; y una sola membrana etiquetada por 2 que contiene el multiconjunto de entrada $cod(\varphi)$, un objeto γ_0, p copias de T_i y F_i, siendo $1 \leq i \leq n$, the objetos $a_{1,1}, \ldots, a_{n,1}$. Por tanto, la configuración $\mathcal{C}_0$ da lugar a la configuración $\mathcal{C}_1$ por la aplicación de las reglas:

  $$[\, a_{1,1} \,]_2 \to [\, t_{1,1} \,]_2 [\, f_{1,1} \,]_2$$
  $$[\, a_{i,1} \to a_{i,2} \,]_2 \text{ para } k+1 \leq i \leq n$$
  $$[\, \alpha_0 \to \alpha_1 \,]_0$$
  $$[\, \beta_0 \to \beta_1 \,]_0$$
  $$[\, \gamma_0 \to \gamma_1 \,]_2$$
  $$\left.\begin{array}{l} [\, x_{i,j,0} \to x_{i,j,1} \,]_2 \\ [\, \overline{x}_{i,j,0} \to \overline{x}_{i,j,1} \,]_2 \\ [\, x^*_{i,j,0} \to x^*_{i,j,1} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

  (a_1) en la configuración $\mathcal{C}_1$ tenemos que $\mathcal{C}_1(0) = \{\alpha_1, \beta_1\}$ y hay una sola membrana etiquetada por 1 vacía; y dos membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_1(\varphi)$, un objeto γ_1, p copias de T_i y F_i, siendo $1 \leq i \leq n$, the objetos $a_{2,2}, \ldots, a_{n,2}$, una de ellas con un objeto $t_{1,1}$ y la otra con un objeto $f_{1,1}$. Así, la

configuración $\mathcal{C}_1$ da lugar a la configuración $\mathcal{C}_2$ por la aplicación de las reglas:

$[\, t_{1,1} \to t_{1,2} \,]_2$
$[\, f_{1,1} \to f_{1,2} \,]_2$
$[\, [\,]_2 [\,]_2 \,]_1 \to [\, [\,]_2 \,]_1 \, [\, [\,]_2 \,]_1$
$[\, a_{i,2} \to a_{i,3} \,]_2$ para $2 \leq i \leq n$
$[\, \alpha_1 \to \alpha_2 \,]_0$
$[\, \beta_1 \to \beta_2 \,]_0$
$[\, \gamma_1 \to \gamma_2 \,]_2$
$\left.\begin{array}{l} [\, x_{i,j,1} \to x_{i,j,2} \,]_2 \\ [\, \overline{x}_{i,j,1} \to \overline{x}_{i,j,2} \,]_2 \\ [\, x^*_{i,j,1} \to x^*_{i,j,2} \,]_2 \end{array}\right\}$ para $1 \leq i \leq n, 1 \leq j \leq p$

Por lo tanto, $\mathcal{C}_2(0) = \{\alpha_2, \beta_2\}$, y there exist two membranas vacías etiquetadas por 1; y dos membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_2(\varphi)$, un objeto γ_2, p copias de T_i y F_i, siendo $1 \leq i \leq n$, los objetos $a_{2,3}, \ldots, a_{n,3}$ y una con un objeto $t_{1,2}$ y la otra con un objeto $f_{1,2}$. Por lo tanto, el resultado es válido para $k = 1$.

- Suponiendo, por inducción, que el resultado es cierto para k ($0 \leq k \leq n-1$)
  - $\mathcal{C}_{2k}(0) = \{\alpha_{2k}, \beta_{2k}\}$
  - En la configuración $\mathcal{C}_{2k}$ hay 2^k membranas vacías etiquetadas por 1.
  - En la configuración $\mathcal{C}_{2k}$ hay 2^k membranas etiquetadas por 2 tales que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2k}(\varphi)$;
    - un objeto γ_{2k};
    - p copias de los objetos T_i y F_i, $1 \leq i \leq n$.
    - los objetos $a_{i,2k+1}$, $k+1 \leq i \leq n$; y
    - un subconjunto diferente $\{r_{1,j}, \ldots, r_{k,j}\}$, $k+1 \leq j \leq 2k$, siendo $r \in \{t, f\}$.

  Por tanto, la configuración $\mathcal{C}_{2k}$ da lugar a la configuración $\mathcal{C}_{2k+1}$ por la aplicación de las reglas:

  $[\, a_{k,2k+1} \,]_2 \to [\, t_{k,k} \,]_2 \, [\, f_{k,k} \,]_2$
  $[\, a_{i,2k+1} \to a_{i,2k+2} \,]_2$ para $k+1 \leq i \leq n$

$$\left.\begin{array}{l}[\, t_{i,j} \to t_{i,j+1} \,]_2 \\ [\, f_{i,j} \to f_{i,j+1} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq k-1, k+1 \leq j \leq 2k$$
$$[\, \alpha_{2k} \to \alpha_{2k+1} \,]_0$$
$$[\, \beta_{2k} \to \beta_{2k+1} \,]_0$$
$$[\, \gamma_{2k} \to \gamma_{2k+1} \,]_2$$
$$\left.\begin{array}{l}[\, x_{i,j,2k} \to x_{i,j,2k+1} \,]_2 \\ [\, \overline{x}_{i,j,1} \to \overline{x}_{i,j,2k+1} \,]_2 \\ [\, x^*_{i,j,1} \to x^*_{i,j,2k+1} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2k+1}(0) = \{\alpha_{2k+1}, \beta_{2k+1}\}$
- En la configuración $\mathcal{C}_{2k+1}$ hay 2^k membranas vacías etiquetadas por 1.
- En la configuración $\mathcal{C}_{2k+1}$ hay 2^{k+1} membranas etiquetadas por 2 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{2k+1}(\varphi)$;
  - un objeto γ_{2k+1};
  - p copias de los objetos T_i y F_i, $1 \leq i \leq n$.
  - los objetos $a_{i,2(k+1)}$, $k+1 \leq i \leq n$; y
  - un subconjunto diferente $\{r_{1,j}, \ldots, r_{k+1,j}\}$, $k+1 \leq j \leq 2k+1$, siendo $r \in \{t, f\}$.

  Por tanto, la configuración $\mathcal{C}_{2k+1}$ da lugar a la configuración $\mathcal{C}_{2(k+1)}$ por la aplicación de las reglas:

$$\left.\begin{array}{l}[\, t_{i,j} \to t_{i,j+1} \,]_2 \\ [\, f_{i,j} \to f_{i,j+1} \,]_2 \end{array}\right\} \text{ para } 1 \leq i \leq k+1, k+1 \leq j \leq 2k+1$$
$$[\, [\,]_2 [\,]_2 \,]_1 \to [\, [\,]_2 \,]_1 [\, [\,]_2 \,]_1$$
$$[\, a_{i,2(k+1)} \to a_{i,2(k+1)+1} \,]_2 \text{ para } k+1 \leq i \leq n$$
$$[\, \alpha_{2k+1} \to \alpha_{2(k+1)} \,]_0$$
$$[\, \beta_{2k+1} \to \beta_{2(k+1)} \,]_0$$
$$[\, \gamma_{2k+1} \to \gamma_{2(k+1)} \,]_2$$
$$\left.\begin{array}{l}[\, x_{i,j,2k+1} \to x_{i,j,2k+2} \,]_2 \\ [\, \overline{x}_{i,j,2k+1} \to \overline{x}_{i,j,2k+2} \,]_2 \\ [\, x^*_{i,j,2k+1} \to x^*_{i,j,2k+2} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2(k+1)}(0) = \{\alpha_{2(k+1)}, \beta_{2(k+1)}\}$
- En la configuración $\mathcal{C}_{2(k+1)}$ hay 2^{k+1} membranas vacías etiquetadas por 1.

- En la configuración $\mathcal{C}_{2(k+1)}$ hay 2^{k+1} membranas etiquetadas por 2 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{2(k+1)}(\varphi)$;
  - un objeto $\gamma_{2(k+1)}$;
  - p copias de los objetos T_i y F_i, $1 \leq i \leq n$.
  - los objetos $a_{i,2(k+1)+1}$, $k+1 \leq i \leq n$; y
  - un subconjunto diferente $\{r_{1,j}, \ldots, r_{k+1,j}\}$, $k+1 \leq j \leq 2(k+1)+1$.

  Por lo tanto, el resultado es válido para $k+1$.

- Para probar (b) basta darse cuenta que, por una parte, por (a) la configuración $\mathcal{C}_{2n-1}$ es válido:
  - $\mathcal{C}_{2n-1}(0) = \{\alpha_{2n-1}, \beta_{2n-1}\}$
  - En la configuración $\mathcal{C}_{2n-1}$ hay 2^{n-1} membranas vacías etiquetadas por 1.
  - En la configuración $\mathcal{C}_{2n-1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2n-1}p(\varphi)$;
    - un objeto γ_{2n-1};
    - p copias de T_i y F_i, $1 \leq i \leq n$; y
    - un subconjunto diferente de objetos $r_{i,2n-i}$, $1 \leq i \leq n$.

    Por tanto, la configuración $\mathcal{C}_{2n-1}$ da lugar a la configuración $\mathcal{C}_{2n}$ por la aplicación de las reglas:
    $$\left.\begin{array}{l}[\, t_{i,2n-i} \to t_{i,2n+1-i}\,]_2 \\ [\, f_{i,2n-i} \to f_{i,2n+1-1}\,]_2\end{array}\right\} \text{ para } 1 \leq i \leq n$$
    $$[\,[\,]_2[\,]_2\,]_1 \to [\,[\,]_2\,]_1\,[\,[\,]_2\,]_1$$
    $$[\,\alpha_{2n-1} \to \alpha_{2n}\,]_0$$
    $$[\,\beta_{2n-1} \to \beta_{2n}\,]_0$$
    $$[\,\gamma_{2n-1} \to \gamma_{2n}\,]_2$$
    $$\left.\begin{array}{l}[\, x_{i,j,2n-1} \to x_{i,j,2n}\,]_2 \\ [\, \overline{x}_{i,j,2n-1} \to \overline{x}_{i,j,2n}\,]_2 \\ [\, x^*_{i,j,2n-1} \to x^*_{i,j,2n}\,]_2\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$
    Por tanto, tenemos que $\mathcal{C}_{2n}(0) = \{\alpha_{2n}, \beta_{2n}\}$, y hay 2^n membranas vacías etiquetadas por 1; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{2n}(\varphi)$, un objeto γ_{2n}, p

copias de T_i y F_i, siendo $1 \leq i \leq n$ y un subconjunto diferente de objetos $r_{i,2n+1-i}$, siendo $1 \leq i \leq n$.

□

Cuando la estructura de árbol se crea, empezamos a asignar una valoración de verdad a cada rama. Esto se ejecuta en los siguientes $2np - n$ pasos. Los últimos n pasos son diferentes de los pasos previos, así que se expondrán en una proposición diferente.

Proposición 7.8. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a_0) *para each* k $(1 \leq k \leq n)$ *y* l $(0 \leq l \leq p-1)$ *en la configuración* $\mathcal{C}_{2n+2ln+k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2n+2ln+k}(0) = \{\alpha_{2n+2ln+k}, \beta_{2n+2ln+k}\}$
- *Hay* 2^n *membranas etiquetadas por 1 tales que cada una de ellas contiene un subconjunto diferente de objetos* $r_{i,2n+2ln+k-i+1}$, $1 \leq i \leq k$, *siendo* $r \in \{t, f\}$.
- *Hay* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada* $cod_{2n+2ln+k}(\varphi)$;
  - *un objeto* $\gamma_{2n+2ln+k}$;
  - p *copias de cada objeto* T_i *y* F_i, $1 \leq i \leq n$ *si la valoración de verdad asociada a la rama contiene su correspondiente objeto* t_i *o* f_i; *de lo contrario, hay* $p-l$ *copias si* $k+1 \leq i \leq n$, $p-l-1$ *de lo contrario; y*
  - *los objetos* $r_{i,2n+2ln+k-i+1}$, $k+1 \leq i \leq n$, *siendo* $r \in \{t, f\}$.

(a_1) *Para cada* k $(1 \leq k \leq n)$ *y* l $(0 \leq l \leq p-1)$ *en la configuración* $\mathcal{C}_{3n+2ln+k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{2n+2ln+k}(0) = \{\alpha_{2n+2ln+k}, \beta_{2n+2ln+k}\}$
- *Hay* 2^n *membranas etiquetadas por 1 tales que cada una de ellas contiene un subconjunto diferente de objetos* $r_{i,3n+ln+k-i+1}$, $k+1 \leq i \leq n$, *siendo* $r \in \{t, f\}$.
- *Hay* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene*

- *el multiconjunto de entrada* $cod_{3n+2ln+k}(\varphi)$;
- *un objeto* $\gamma_{3n+2ln+k}$;
- p *copias de cada objeto* T_i *y* F_i, $1 \leq i \leq n$ *si la valoración de verdad asociada a la rama contiene su objeto* t_i *o* f_i *correspondiente, y* $p - l$ *copias de lo contrario; y*
- *los objetos* $r_{i,3n+2ln+k-i+1}$, $1 \leq i \leq k$, *siendo* $r \in \{t, f\}$.

(*b*) $\mathcal{C}_{2np}(0) = \{\alpha_{2np}, \beta_{2np}\}$, *y en la configuración* $\mathcal{C}_{2np}$ *hay* 2^n *membranas vacías etiquetadas por 1; y* 2^n *membranas etiquetadas por 2 tales que cada una de ellas contiene el multiconjunto de entrada* $cod_{2np}(\varphi)$, *un objeto* γ_{2np}, p *copias de cada objeto* T_i *y* F_i, $1 \leq i \leq n$ *si la valoración de verdad asociada a la rama contiene su correspondiente objeto* t_i *o* f_i, *y 1 objeto de lo contrario y objetos* $r_{i,2np-i+1}$, $1 \leq i \leq n$, *siendo* $r \in \{t, f\}$; *es decir, la valoración de verdad asociada a la rama.*

Demostración. (a) será demostrado por inducción sobre l

- El caso base $l = 0$ será demostrado por inducción sobre k

  (a_0) El caso base $k = 1$ es trivial dado que:

  - en la configuración $\mathcal{C}_{2n}$ tenemos que: $\mathcal{C}_{2n}(0) = \{\alpha_{2n}, \beta_{2n}\}$ y hay 2^n membranas vacías etiquetadas por 1; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{2n}(\varphi)$, un objeto γ_{2n}, p copias de T_i y F_i, siendo $1 \leq i \leq n$ y un subconjunto diferente de objetos $r_{i,2n-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$, la valoración de verdad correspondiente a la rama. Por tanto, la configuración $\mathcal{C}_{2n}$ da lugar a la configuración $\mathcal{C}_{2n+1}$ por la aplicación de las reglas:
    $$\left.\begin{array}{l} [\, t_{i,2n} F_i \,]_2 \to t_{i,2n+1} [\quad]_2 \\ [\, f_{i,2n} T_i \,]_2 \to f_{i,2n+1} [\quad]_2 \\ [\, t_{i,2n+1-i} \to t_{i,2n+2-i} \,]_2 \\ [\, f_{i,2n+1-i} \to f_{i,2n+2-1} \,]_2 \end{array}\right\} \text{para } 2 \leq i \leq n$$
    $$[\, \alpha_{2n} \to \alpha_{2n+1} \,]_0$$
    $$[\, \beta_{2n} \to \beta_{2n+1} \,]_0$$
    $$[\, \gamma_{2n} \to \gamma_{2n+1} \,]_2$$
    $$\left.\begin{array}{l} [\, x_{i,j,2n} \to x_{i,j,2n+1} \,]_2 \\ [\, \overline{x}_{i,j,2n} \to \overline{x}_{i,j,2n+1} \,]_2 \\ [\, x^*_{i,j,2n} \to x^*_{i,j,2n+1} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$
    Por lo tanto, $\mathcal{C}_{2n+1}(0) = \{\alpha_{2n+1}, \beta_{2n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen un objeto $r_{1,2n+1}$, siendo

$r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{2n+1}(\varphi)$, un objeto γ_{2n+1}, p copias de T_i y F_i, siendo $2 \leq i \leq n$, y $p-1$ copias de T_1 (respectivamente, F_i) si objeto $f_{1,2n}$ (respectivamente, $t_{1,2n}$) estaba en dicha membrana etiquetada por 2 en la configuración $\mathcal{C}_{2n}$, y p copias de F_1 (respectivamente, T_1), y un subconjunto diferente de objetos $r_{i,2n-i+2}$, $2 \leq i \leq n$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{2n+k}(0) = \{\alpha_{2n+k}, \beta_{2n+k}\}$
  - En la configuración $\mathcal{C}_{2n+k}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene objetos $r_{i,2n+k-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{2n+k}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2n+k}(\varphi)$;
    - un objeto γ_{2n+k};
    - p copias de cada objeto T_i y F_i, $1 \leq i \leq n$ si la valoración de verdad asociada a la rama contiene su correspondiente objeto t_i o f_i, $p-1$ objetos T_i y F_i ($1 \leq i \leq k$) de lo contrario; y
    - un subconjunto diferente de objetos $r_{i,2n+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.

  Por tanto, la configuración $\mathcal{C}_{2n+k}$ da lugar a la configuración $\mathcal{C}_{2n+k+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
[\, t_{k+1,2n} F_{k+1} \,]_2 \to t_{k+1,2n+1} [\ \ ]_2 \\
[\, f_{k+1,2n} T_{k+1} \,]_2 \to f_{k+1,2n+1} [\ \ ]_2 \\
\left.\begin{array}{l} [\, t_{i,2n+k-i+1} \to t_{i,2n+k-i+2} \,]_2 \\ [\, f_{i,2n+k-i+1} \to f_{i,2n+k-i+2} \,]_2 \end{array}\right\} \text{para } k+2 \leq i \leq n \\
\left.\begin{array}{l} [\, t_{i,2n+k-i+1} \to t_{i,2n+k-i+2} \,]_1 \\ [\, f_{i,2n+k-i+1} \to f_{i,2n+k-i+2} \,]_1 \end{array}\right\} \text{para } 1 \leq i \leq k \\
[\, \alpha_{2n+k} \to \alpha_{2n+k+1} \,]_0 \\
[\, \beta_{2n+k} \to \beta_{2n+k+1} \,]_0 \\
[\, \gamma_{2n+k} \to \gamma_{2n+k+1} \,]_2 \\
\left.\begin{array}{l} [\, x_{i,j,2n+k} \to x_{i,j,2n+k+1} \,]_2 \\ [\, \overline{x}_{i,j,2n+k} \to \overline{x}_{i,j,2n+k+1} \,]_2 \\ [\, x^*_{i,j,2n+k} \to x^*_{i,j,2n+k+1} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p
\end{array}
$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2n+k+1}(0) = \{\alpha_{2n+k+1}, \beta_{2n+k+1}\}$
- En la configuración $\mathcal{C}_{2n+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene objetos $r_{i,2n+k-i+2}$, $1 \leq i \leq k+1$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{2n+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{2n+k+1}(\varphi)$;
  - un objeto γ_{2n+k+1};
  - p copias de cada objeto T_i y F_i, $1 \leq i \leq n$ si la valoración de verdad asociada a la rama contiene su correspondiente objeto t_i o f_i, $p-1$ objetos T_i y F_i $(1 \leq i \leq k+1)$ de lo contrario; y
  - un subconjunto diferente de objetos $r_{i,2n+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.

(a_1) El caso base $k = 1$ es trivial dado que:

- en la configuración $\mathcal{C}_{3n}$ tenemos que $\mathcal{C}_{3n}(0) = \{\alpha_{3n}, \beta_{3n}\}$ y hay 2^n membranas etiquetadas por 1 que contienen y un subconjunto diferente de objetos $r_{i,3n+1-i}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad correspondiente a la rama; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{3n}(\varphi)$, un objeto γ_{3n}, p copias de T_i y F_i, siendo $1 \leq i \leq n$ si la valoración de verdad asociada a la rama contiene su correspondiente objeto objeto t_i o f_i, $p-1$s de lo contrario. Por tanto, la configuración $\mathcal{C}_{3n}$ da lugar a la configuración $\mathcal{C}_{3n+1}$ por la aplicación de las reglas:

$$
\left.
\begin{array}{l}
t_{1,3n}\,[\ \ ]_2 \to [\,t_{1,3n+1}\,]_2 \\
f_{1,3n}\,[\ \ ]_2 \to [\,f_{1,3n+1}\,]_2 \\
[\,t_{i,3n-i+1} \to t_{i,3n-i+2}\,]_1 \\
[\,f_{i,3n-i+1} \to f_{i,3n-i+2}\,]_1
\end{array}
\right\} \text{para } 2 \leq i \leq n
$$

$$
\begin{array}{l}
[\,\alpha_{3n} \to \alpha_{3n+1}\,]_0 \\
[\,\beta_{3n} \to \beta_{3n+1}\,]_0 \\
[\,\gamma_{3n} \to \gamma_{3n+1}\,]_2
\end{array}
$$

$$
\left.
\begin{array}{l}
[\,x_{i,j,3n} \to x_{i,j,3n+1}\,]_2 \\
[\,\overline{x}_{i,j,3n} \to \overline{x}_{i,j,3n+1}\,]_2 \\
[\,x^*_{i,j,3n} \to x^*_{i,j,3n+1}\,]_2
\end{array}
\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p
$$

Por lo tanto, $\mathcal{C}_{3n+1}(0) = \{\alpha_{3n+1}, \beta_{3n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen un subconjunto diferente de

objetos $r_{i,3n-i+2}$, $2 \leq i \leq n$, siendo $r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{3n+1}(\varphi)$, un objeto γ_{3n+1}, p copias de T_i y F_i, siendo $1 \leq i \leq n$ si la valoración de verdad asociada a la rama tiene el correspondiente objeto t_i o f_i, $p-1$ objetos de lo contrario y un objeto $r_{1,3n+1}$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{3n+k}(0) = \{\alpha_{3n+k}, \beta_{3n+k}\}$
  - En la configuración $\mathcal{C}_{3n+k}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene objetos $r_{i,3n+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{3n+k}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{3n+k}(\varphi)$;
    - un objeto γ_{3n+k};
    - p copias de cada objeto T_i y F_i para $1 \leq i \leq n$ o si el objeto t_i o f_i correspondiente está asignado a esa rama, $p-l$ copias de lo contrario; y
    - un subconjunto diferente de objetos $r_{i,3n+k-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$.

    Por tanto, la configuración $\mathcal{C}_{3n+k}$ da lugar a la configuración $\mathcal{C}_{3n+k+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
t_{k+1,3n}\,[\ \ ]_2 \to [\,t_{k+1,3n+1}\,]_2 \\
f_{k+1,3n}\,[\ \ ]_2 \to [\,f_{k+1,3n+1}\,]_2 \\
\left.\begin{array}{l} [\,t_{i,3n+k-i+1} \to t_{i,3n+k-i+2}\,]_1 \\ [\,f_{i,3n+k-i+1} \to f_{i,3n+k-i+2}\,]_1 \end{array}\right\} \text{para } k+2 \leq i \leq n \\
\left.\begin{array}{l} [\,t_{i,3n+k-i+1} \to t_{i,3n+k-i+2}\,]_2 \\ [\,f_{i,3n+k-i+1} \to f_{i,3n+k-i+2}\,]_2 \end{array}\right\} \text{para } 1 \leq i \leq k \\
[\,\alpha_{3n+k} \to \alpha_{3n+k+1}\,]_0 \\
[\,\beta_{3n+k} \to \beta_{3n+k+1}\,]_0 \\
[\,\gamma_{3n+k} \to \gamma_{3n+k+1}\,]_2 \\
\left.\begin{array}{l} [\,x_{i,j,3n+k} \to x_{i,j,3n+k+1}\,]_2 \\ [\,\overline{x}_{i,j,3n+k} \to \overline{x}_{i,j,3n+k+1}\,]_2 \\ [\,x^*_{i,j,3n+k} \to x^*_{i,j,3n+k+1}\,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p
\end{array}
$$

    Por lo tanto, lo siguiente es válido
  - $\mathcal{C}_{3n+k+1}(0) = \{\alpha_{3n+k+1}, \beta_{3n+k+1}\}$

- En la configuración $\mathcal{C}_{3n+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene objetos $r_{i,3n+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{3n+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{3n+k+1}(\varphi)$;
  - un objeto γ_{3n+k+1};
  - p copias de cada objeto T_i y F_i para $1 \leq i \leq n$ o the corresponding t_i o f_i is assigned to that branch, $p-l$ copias de lo contrario; y
  - un subconjunto diferente de objetos $r_{i,3n+k-i+2}$, $1 \leq i \leq k+1$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para l $(0 \leq l \leq p-1)$

  (a_0) El caso base $k = 1$ es trivial dado que:

  - en la configuración $\mathcal{C}_{2n+(l+1)n}$[11] tenemos que: $\mathcal{C}_{2n+(l+1)n}(0) = \{\alpha_{2n+(l+1)n}, \beta_{2n+(l+1)n}\}$ y hay 2^n membranas vacías etiquetadas por 1; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{2n+(l+1)n}(\varphi)$, un objeto $\gamma_{2n+(l+1)n}$, p copias de T_i y F_i, siendo $1 \leq i \leq n$, y $p-l$ copias para T_i (respectivamente, F_i) objetos que pertenecen a una rama con un objeto f_i (respectivamente, t_i) y un subconjunto diferente de objetos $r_{i,2n+(l+1)n-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$, la valoración de verdad correspondiente a la rama. Así, la configuración $\mathcal{C}_{2n+(l+1)n}$ da lugar a la configuración $\mathcal{C}_{2n+(l+1)n+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
[\, t_{i,2n+(l+1)n} F_i \,]_2 \to t_{i,2n+(l+1)n+1} [\ \ ]_2 \\
[\, f_{i,2n+(l+1)n} T_i \,]_2 \to f_{i,2n+(l+1)n+1} [\ \ ]_2 \\
\left.\begin{array}{l}
[\, t_{i,2n+(l+1)n+1-i} \to t_{i,2n+(l+1)n+2-i} \,]_2 \\
[\, f_{i,2n+(l+1)n+1-i} \to f_{i,2n+(l+1)n+2-i} \,]_2
\end{array}\right\} \text{para } 2 \leq i \leq n \\
[\, \alpha_{2n+(l+1)n} \to \alpha_{2n+(l+1)n+1} \,]_0 \\
[\, \beta_{2n+(l+1)n} \to \beta_{2n+(l+1)n+1} \,]_0 \\
[\, \gamma_{2n+(l+1)n} \to \gamma_{2n+(l+1)n+1} \,]_2
\end{array}
$$

[11] Obsérvese que $(l+1)n = ln + n$, y se ha demostrado en el primer paso de inducción que es correcto.

$$\left.\begin{array}{l}[\, x_{i,j,2n+(l+1)n} \to x_{i,j,2n+(l+1)n+1}\,]_2\\ [\, \overline{x}_{i,j,2n+(l+1)n} \to \overline{x}_{i,j,2n+(l+1)n+1}\,]_2\\ [\, x^*_{i,j,2n+(l+1)n} \to x^*_{i,j,2n+(l+1)n+1}\,]_2\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, $\mathcal{C}_{2n+(l+1)n+1}(0) = \{\alpha_{2n+(l+1)n+1}, \beta_{2n+(l+1)n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen y un objeto $r_{1,2n+(l+1)n+1}$, siendo $r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{2n+(l+1)n+1}(\varphi)$, un objeto $\gamma_{2n+(l+1)n+1}$, p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, de lo contrario $p - l$ copias de F_i (respectivamente, T_i) si $2 \leq i \leq n$, $p - l - 1$ de lo contrario y un subconjunto diferente de objetos $r_{i,2n+(l+1)n-i+2}$, $2 \leq i \leq n$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{2n+(l+1)n+k}(0) = \{\alpha_{2n+(l+1)n+k}, \beta_{2n+(l+1)n+k}\}$
  - En la configuración $\mathcal{C}_{2n+(l+1)n+k}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene objetos $r_{i,2n+(l+1)n+k-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{2n+(l+1)n+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2n+(l+1)n+k}(\varphi)$;
    - un objeto $\gamma_{2n+(l+1)n+k}$;
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, de lo contrario $p - l$ copias de F_i (respectivamente, T_i) si $k + 1 \leq i \leq n$, $p - l - 1$ de lo contrario; y
    - un subconjunto diferente de objetos $r_{i,2n+(l+1)n+k-i+1}$, $k + 1 \leq i \leq n$, siendo $r \in \{t, f\}$.

  Por tanto, la configuración $\mathcal{C}_{2n+k}$ da lugar a la configuración $\mathcal{C}_{2n+(l+1)n+k+1}$ por la aplicación de las reglas:

$$[\, t_{k+1,2n+(l+1)n} F_{k+1}\,]_2 \to t_{k+1,2n+(l+1)n+1}\,[\quad]_2$$
$$[\, f_{k+1,2n+(l+1)n} T_{k+1}\,]_2 \to f_{k+1,2n+(l+1)n+1}\,[\quad]_2$$
$$\left.\begin{array}{l}[\, t_{i,2n+(l+1)n+k-i+1} \to t_{i,2n+k-i+2}\,]_2\\ [\, f_{i,2n+(l+1)n+k-i+1} \to f_{i,2n+k-i+2}\,]_2\end{array}\right\} \text{para } k+2 \leq i \leq n$$
$$\left.\begin{array}{l}[\, t_{i,2n+(l+1)n+k-i+1} \to t_{i,2n+(l+1)n+k-i+2}\,]_1\\ [\, f_{i,2n+(l+1)n+k-i+1} \to f_{i,2n+(l+1)n+k-i+2}\,]_1\end{array}\right\} \text{para } 1 \leq i \leq k$$

$$\left.\begin{array}{l}
[\,\alpha_{2n+(l+1)n+k} \to \alpha_{2n+(l+1)n+k+1}\,]_0 \\
[\,\beta_{2n+(l+1)n+k} \to \beta_{2n+(l+1)n+k+1}\,]_0 \\
[\,\gamma_{2n+(l+1)n+k} \to \gamma_{2n+(l+1)n+k+1}\,]_2 \\
[\,x_{i,j,2n+(l+1)n+k} \to x_{i,j,2n+(l+1)n+k+1}\,]_2 \\
[\,\overline{x}_{i,j,2n+(l+1)n+k} \to \overline{x}_{i,j,2n+(l+1)n+k+1}\,]_2 \\
[\,x^*_{i,j,2n+(l+1)n+k} \to x^*_{i,j,2n+(l+1)n+k+1}\,]_2
\end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{2n+(l+1)n+k+1}(0) = \{\alpha_{2n+(l+1)n+k+1}, \beta_{2n+(l+1)n+k+1}\}$
- En la configuración $\mathcal{C}_{2n+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene objetos $r_{i,2n+(l+1)n+k-i+2}$, $1 \leq i \leq k+1$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{2n+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{2n+(l+1)n+k+1}(\varphi)$;
  - un objeto $\gamma_{2n+(l+1)n+k+1}$;
  - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, de lo contrario $p - l$ copias de F_i (respectivamente, T_i) si $k+2 \leq i \leq n$, $p - l - 1$ de lo contrario; y
  - un subconjunto diferente de objetos $r_{i,2n+(l+1)n+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.

(a_1) El caso base $k = 1$ es trivial dado que:

- en la configuración $\mathcal{C}_{3n+(l+1)n}$ tenemos que $\mathcal{C}_{3n+(l+1)n}(0) = \{\alpha_{3n+(l+1)n}, \beta_{3n+(l+1)n}\}$ y hay 2^n membranas etiquetadas por 1 que contienen un subconjunto diferente de objetos $r_{i,3n+(l+1)n-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad correspondiente a la rama; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{3n+(l+1)n}(\varphi)$, un objeto $\gamma_{3n+(l+1)n}$ y p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p-l$ copias de F_i (respectivamente, T_i). Por tanto, la configuración $\mathcal{C}_{3n+(l+1)n}$ da lugar a la configuración $\mathcal{C}_{3n+(l+1)n+1}$ por la aplicación de las reglas:

$$\left.\begin{array}{l}
t_{1,3n+(l+1)n}\,[\quad]_2 \to [\,t_{1,3n+(l+1)n+1}\,]_2 \\
f_{1,3n+(l+1)n}\,[\quad]_2 \to [\,f_{1,3n+(l+1)n+1}\,]_2 \\
[\,t_{i,3n+(l+1)n-i+1} \to t_{i,3n+(l+1)n-i+2}\,]_1 \\
[\,f_{i,3n+(l+1)n-i+1} \to f_{i,3n+(l+1)n-i+2}\,]_1
\end{array}\right\} \text{para } 2 \leq i \leq n$$

$$\left.\begin{array}{l}
[\,\alpha_{3n+(l+1)n} \to \alpha_{3n+(l+1)n+1}\,]_0 \\
[\,\beta_{3n+(l+1)n} \to \beta_{3n+(l+1)n+1}\,]_0 \\
[\,\gamma_{3n+(l+1)n} \to \gamma_{3n+(l+1)n+1}\,]_2 \\
[\,x_{i,j,3n+(l+1)n} \to x_{i,j,3n+(l+1)n+1}\,]_2 \\
[\,\overline{x}_{i,j,3n+(l+1)n} \to \overline{x}_{i,j,3n+(l+1)n+1}\,]_2 \\
[\,x^*_{i,j,3n+(l+1)n} \to x^*_{i,j,3n+(l+1)n+1}\,]_2
\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, $\mathcal{C}_{3n+(l+1)n+1}(0) = \{\alpha_{3n+(l+1)n+1}, \beta_{3n+(l+1)n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen un subconjunto diferente de objetos $r_{i,3n+(l+1)n-i+2}$, $2 \leq i \leq n$, siendo $r \in \{t, f\}$; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{3n+(l+1)n+1}(\varphi)$, un objeto $\gamma_{3n+(l+1)n+1}$, p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p-l$ copias de F_i (respectivamente, T_i) y un objeto $r_{1,3n+(l+1)n+1}$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n$)
  - $\mathcal{C}_{3n+(l+1)n+k}(0) = \{\alpha_{3n+(l+1)n+k}, \beta_{3n+(l+1)n+k}\}$
  - En la configuración $\mathcal{C}_{3n+(l+1)n+k}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene objetos $r_{i,3n+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{3n+(l+1)n+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{3n+(l+1)n+k}(\varphi)$;
    - un objeto $\gamma_{3n+(l+1)n+k}$;
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p-l$ copias de F_i (respectivamente, T_i)
    - un subconjunto diferente de objetos $r_{i,3n+(l+1)n-i+1}$, $1 \leq i \leq k$, siendo $r \in \{t, f\}$.

  Por tanto, la configuración $\mathcal{C}_{3n+(l+1)n+k}$ da lugar a la configuración $\mathcal{C}_{3n+(l+1)n+k+1}$ por la aplicación de las reglas:

$$\begin{array}{l}
t_{k+1,3n+(l+1)n}\,[\quad]_2 \to [\,t_{k+1,3n+(l+1)n+1}\,]_2 \\
f_{k+1,3n+(l+1)n}\,[\quad]_2 \to [\,f_{k+1,3n+(l+1)n+1}\,]_2 \\
\left.\begin{array}{l}
[\,t_{i,3n+(l+1)n+k-i+1} \to t_{i,3n+(l+1)n+k-i+2}\,]_1 \\
[\,f_{i,3n+(l+1)n+k-i+1} \to f_{i,3n+(l+1)n+k-i+2}\,]_1
\end{array}\right\} \text{para } k+2 \leq i \leq n \\
\left.\begin{array}{l}
[\,t_{i,3n+(l+1)n+k-i+1} \to t_{i,3n+(l+1)n+k-i+2}\,]_2 \\
[\,f_{i,3n+(l+1)n+k-i+1} \to f_{i,3n+(l+1)n+k-i+2}\,]_2
\end{array}\right\} \text{para } 1 \leq i \leq k
\end{array}$$

$$\left.\begin{array}{l}[\,\alpha_{3n+(l+1)n+k} \to \alpha_{3n+(l+1)n+k+1}\,]_0\\ [\,\beta_{3n+(l+1)n+k} \to \beta_{3n+(l+1)n+k+1}\,]_0\\ [\,\gamma_{3n+(l+1)n+k} \to \gamma_{3n+(l+1)n+k+1}\,]_2\\ [\,x_{i,j,3n+(l+1)n+k} \to x_{i,j,3n+(l+1)n+k+1}\,]_2\\ [\,\overline{x}_{i,j,3n+(l+1)n+k} \to \overline{x}_{i,j,3n+(l+1)n+k+1}\,]_2\\ [\,x^*_{i,j,3n+(l+1)n+k} \to x^*_{i,j,3n+(l+1)n+k+1}\,]_2\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{3n+(l+1)n+k+1}(0) = \{\alpha_{3n+(l+1)n+k+1}, \beta_{3n+(l+1)n+k+1}\}$
- En la configuración $\mathcal{C}_{3n+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene objetos $r_{i,3n+(l+1)n+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.
- En la configuración $\mathcal{C}_{3n+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
  - el multiconjunto de entrada $cod_{3n+(l+1)n+k+1}(\varphi)$;
  - un objeto $\gamma_{3n+(l+1)n+k+1}$;
  - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y $p - l$ copias de F_i (respectivamente, T_i)
  - un subconjunto diferente de objetos $r_{i,3n+(l+1)n+k-i+2}$, $1 \leq i \leq k+1$, siendo $r \in \{t, f\}$.

- Para probar (b) basta con darse cuenta de que, por una parte, de (a), en la configuración $\mathcal{C}_{2np-1}$[12] se cumple:

  - $\mathcal{C}_{2np-1}(0) = \{\alpha_{2np-1}, \beta_{2np-1}\}$
  - En la configuración $\mathcal{C}_{2np-1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene un objeto $r_{n,2np}$, siendo $r \in \{t, f\}$.
  - En la configuración $\mathcal{C}_{n+2np-1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2np-1}(\varphi)$;
    - un objeto γ_{2np-1};
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de lo contrario; y
    - un subconjunto diferente de objetos $r_{i,2np-i}$, $1 \leq i \leq n-1$.

[12] Obsérvese que $2np - 1 = n + 2n(p-1) + (n-1)$

Por tanto, la configuración $\mathcal{C}_{n+2np-1}$ da lugar a la configuración $\mathcal{C}_{n+2np}$ por la aplicación de las reglas:

$$
\begin{array}{l}
t_{n,2np}[\ \ ]_2 \to [\, t_{n,2np+1}\,]_2 \\
f_{n,2np}[\ \ ]_2 \to [\, f_{n,2np+1}\,]_2 \\
\left.\begin{array}{l}
[\, t_{i,n+2np-i} \to t_{i,n+2np-i+1}\,]_2 \\
[\, f_{i,n+2np-i} \to f_{i,n+2np-i}\,]_2
\end{array}\right\} \text{para } 1 \leq i \leq n-1 \\
[\, \alpha_{n+2np-1} \to \alpha_{n+2np}\,]_0 \\
[\, \beta_{n+2np-1} \to \beta_{n+2np}\,]_0 \\
[\, \gamma_{n+2np-1} \to \gamma_{n+2np}\,]_2 \\
\left.\begin{array}{l}
[\, x_{i,j,n+2np-1} \to x_{i,j,n+2np}\,]_2 \\
[\, \overline{x}_{i,j,n+2np-1} \to \overline{x}_{i,j,n+2np}\,]_2 \\
[\, x^*_{i,j,n+2np-1} \to x^*_{i,j,n+2np}\,]_2
\end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p
\end{array}
$$

Por tanto, tenemos que $\mathcal{C}_{2np}(0) = \{\alpha_{2np}, \beta_{2np}\}$, y hay 2^n membranas vacías etiquetadas por 1; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{2np}(\varphi)$, un objeto γ_{2np}, p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de lo contrario y un subconjunto diferente de objetos $r_{i,2np-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad asociada a la rama.

□

Proposición 7.9. *Sea $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ una computación del sistema $\Pi(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$.*

(*a*) *Para cada k ($1 \leq k \leq n-1$) en la configuración $\mathcal{C}_{2np+k}$ tenemos lo siguiente:*

- $\mathcal{C}_{2np+k}(0) = \{\alpha_{2np+k}, \beta_{2np+k}\}$
- *Hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene k objetos #.*
- *hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *el multiconjunto de entrada $cod_{2np+k}(\varphi)$;*
  - *un objeto γ_{2np+k};*
  - *p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de F_i (respectivamente, T_i) si $k+1 \leq i \leq n$; y*

- *los objetos* $r_{i,2np+k-i+1}$, $k+1 \leq i \leq n$.

(*b*) $\mathcal{C}_{n+2np}(0) = \{\alpha_{n+2np}, \beta_{n+2np}\}$, *y en la configuración* $\mathcal{C}_{n+2np}$ *hay* 2^n *membranas etiquetadas por 1, tales que cada una de ellas contiene n objetos #; y* 2^n *membranas etiquetadas por 2, tales que cada una de ellas contiene el multiconjunto de entrada* $cod_{n+2np}(\varphi)$, *un objeto* γ_{n+2np}, p *copias de cada objeto* T_i *y* F_i, $1 \leq i \leq n$ *si la valoración de verdad asociada a la rama contiene su correspondiente objeto* t_i *o* f_i.

Demostración. (*a*) será demostrado por inducción sobre k

- el caso base $k = 1$ es trivial dado que:

  - en la configuración $\mathcal{C}_{2np}$ tenemos que $\mathcal{C}_{2np}(0) = \{\alpha_{2np}, \beta_{2np}\}$ y hay 2^n membranas vacías etiquetadas por 1; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{2np}(\varphi)$, un objeto γ_{2np} p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de lo contrario y un subconjunto diferente de objetos $r_{i,2np-i+1}$, $1 \leq i \leq n$, siendo $r \in \{t, f\}$; es decir, la valoración de verdad asociada a la rama. Por tanto, la configuración $\mathcal{C}_{2np}$ da lugar a la configuración $\mathcal{C}_{2np+1}$ por la aplicación de las reglas:
    $$[\, t_{1,2np} F_1 \,]_2 \to \#\, [\quad]_2$$
    $$[\, f_{1,2np} T_1 \,]_2 \to \#\, [\quad]_2$$
    $$\left.\begin{array}{l} [\, t_{i,2np-i+1} \to t_{i,2np-i+2} \,]_2 \\ [\, f_{i,2np-i+1} \to f_{i,2np-i+2} \,]_2 \end{array}\right\} \text{ para } 2 \leq i \leq n$$
    $$[\, \alpha_{2np} \to \alpha_{2np+1} \,]_0$$
    $$[\, \beta_{2np} \to \beta_{2np+1} \,]_0$$
    $$[\, \gamma_{2np} \to \gamma_{2np+1} \,]_2$$
    $$\left.\begin{array}{l} [\, x_{i,j,2np} \to x_{i,j,2np+1} \,]_2 \\ [\, \overline{x}_{i,j,2np} \to \overline{x}_{i,j,2np+1} \,]_2 \\ [\, x^*_{i,j,2np} \to x^*_{i,j,2np+1} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$
    Por lo tanto, $\mathcal{C}_{2np+1}(0) = \{\alpha_{2np+1}, \beta_{2np+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen un objeto #; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{2np+1}(\varphi)$, un objeto γ_{2np+1}, p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de F_i (respectivamente, T_i) si $k+2 \leq i \leq n$ y objetos $r_{i,2np-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.

- Suponiendo, por inducción, que el resultado es cierto para k ($1 \leq k \leq n-1$)

  - $\mathcal{C}_{2np+k}(0) = \{\alpha_{2np+k}, \beta_{2np+k}\}$
  - En la configuración $\mathcal{C}_{2np+k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene k objetos #.
  - En la configuración $\mathcal{C}_{2np+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2np+k}(\varphi)$;
    - un objeto γ_{2np+k};
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de F_i (respectivamente, T_i) si $k+1 \leq i \leq n$; y
    - los objetos $r_{i,2np+k-i+1}$, $k+1 \leq i \leq n$, siendo $r \in \{t, f\}$.

    Por tanto, la configuración $\mathcal{C}_{2np+k}$ da lugar a la configuración $\mathcal{C}_{2np+k+1}$ por la aplicación de las reglas:

    $$[\, t_{k+1,2np} F_1 \,]_2 \to \#\, [\quad]_2$$
    $$[\, f_{k+1,2np} T_1 \,]_2 \to \#\, [\quad]_2$$
    $$\left.\begin{array}{l} [\, t_{i,2np+k-i+1} \to t_{i,2np+k-i+2} \,]_2 \\ [\, f_{i,2np+k-i+1} \to f_{i,2np+k-i+2} \,]_2 \end{array}\right\} \text{ para } 2 \leq i \leq n$$
    $$[\, \alpha_{2np+k} \to \alpha_{2np+k+1} \,]_0$$
    $$[\, \beta_{2np+k} \to \beta_{2np+k+1} \,]_0$$
    $$[\, \gamma_{2np+k} \to \gamma_{2np+k+1} \,]_2$$
    $$\left.\begin{array}{l} [\, x_{i,j,2np+k} \to x_{i,j,2np+k+1} \,]_2 \\ [\, \overline{x}_{i,j,2np+k} \to \overline{x}_{i,j,2np+k+1} \,]_2 \\ [\, x^*_{i,j,2np+k} \to x^*_{i,j,2np+k+1} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

    Por lo tanto, lo siguiente es válido

  - $\mathcal{C}_{2np+k+1}(0) = \{\alpha_{2np+k+1}, \beta_{2np+k+1}\}$
  - En la configuración $\mathcal{C}_{2np+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene $k+1$ objetos #.
  - En la configuración $\mathcal{C}_{2np+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{2np+k+1}(\varphi)$;
    - un objeto $\gamma_{2np+k+1}$;

- p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de F_i (respectivamente, T_i) si $k+2 \leq i \leq n$; y
- los objetos $r_{i,2np+k-i+2}$, $k+2 \leq i \leq n$, siendo $r \in \{t, f\}$.

- Para probar (b) basta darse cuenta que, por una parte, por (a) la configuración $\mathcal{C}_{n+2np-1}$[13] es válido:

  - $\mathcal{C}_{n+2np-1}(0) = \{\alpha_{n+2np-1}, \beta_{n+2np-1}\}$
  - En la configuración $\mathcal{C}_{n+2np-1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene $n-1$ objetos $\#$.
  - En la configuración $\mathcal{C}_{n+2np-1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - el multiconjunto de entrada $cod_{n+2np-1}(\varphi)$;
    - un objeto $\gamma_{n+2np-1}$;
    - p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama, y 1 copia de F_n (respectivamente, T_n); y
    - un objeto $r_{n,2np}$, siendo $r \in \{t, f\}$.

    Por tanto, la configuración $\mathcal{C}_{n+2np-1}$ da lugar a la configuración $\mathcal{C}_{n+2np}$ por la aplicación de las reglas:

    $$[\, t_{n,2np} F_1 \,]_2 \to \#\, [\quad]_2$$
    $$[\, f_{n,2np} T_1 \,]_2 \to \#\, [\quad]_2$$
    $$[\, \alpha_{n+2np-1} \to \alpha_{n+2np} \,]_0$$
    $$[\, \beta_{n+2np-1} \to \beta_{n+2np} \,]_0$$
    $$[\, \gamma_{n+2np-1} \to \gamma_{n+2np} \,]_2$$
    $$\left.\begin{array}{l} [\, x_{i,j,n+2np-1} \to x_{i,j,n+2np} \,]_2 \\ [\, \overline{x}_{i,j,n+2np-1} \to \overline{x}_{i,j,n+2np} \,]_2 \\ [\, x^*_{i,j,n+2np-1} \to x^*_{i,j,n+2np} \,]_2 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

    Por lo tanto, lo siguiente es válido

  - $\mathcal{C}_{n+2np}(0) = \{\alpha_{n+2np}, \beta_{n+2np}\}$
  - En la configuración $\mathcal{C}_{n+2np}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene n objetos $\#$.
  - En la configuración $\mathcal{C}_{n+2np}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene

[13] Obsérvese que $n + 2np - 1 = 2np + (n-1)$

- el multiconjunto de entrada $cod_{n+2np}(\varphi)$;
- un objeto γ_{n+2np}; y
- p copias de T_i (respectivamente, F_i) siendo $1 \leq i \leq n$ si está el objeto t_i (respectivamente, f_i) correspondiente en dicha rama.

□

Primera fase de chequeo

En esta fase, tratamos de determinar qué cláusulas son satisfechas por la valoración de verdad codificada en cada rama. Para ello, las reglas de 2.1 serán aplicadas de tal manera que en el m-ésimo paso, siendo $m = ln + k$ $(1 \leq k \leq n, 0 \leq l \leq p-1)$, la cláusula C_{l+1} será evaluada con la k-ésima variable de la fórmula φ. Esta fase tarda np pasos.

Proposición 7.10. *Sea $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ una computación del sistema $\Pi(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$.*

(*a*) *Para cada k $(1 \leq k \leq n)$ y l $(0 \leq l \leq p-1)$ en la configuración $\mathcal{C}_{n+2np+ln+k}$ tenemos lo siguiente:*

- $\mathcal{C}_{n+2np+ln+k}(0) = \{\alpha_{n+2np+ln+k}, \beta_{n+2np+ln+k}\}$
- *Hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene*
  - *m objetos $c_{j,t}$ $(1 \leq j \leq l+1,\ 0 \leq t \leq ln+k-1)$; es decir, las cláusulas que han sido satisfechas por alguna variable; y*
  - *$n + ln + k - m$ objetos $\#$.*
- *Hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *los $(n-k)$-ésimos últimos elementos de $cod_{n+2np+ln+k}(\varphi)_{l+1}^{l+1}$;*
  - *el multiconjunto de entrada $cod_{n+2np+ln+k}(\varphi)_{l+2}^{p}$;*
  - *un objeto $\gamma_{n+2np+ln+k}$; y*
  - *$p-l$ copias de objetos T_i o F_i, $k+1 \leq i \leq n$, $p-l-1$ copias de lo contrario, correspondiendo a la valoración de verdad asignada a la rama.*

(*b*) *$\mathcal{C}_{n+3np}(0) = \{\alpha_{n+3np}, \beta_{n+3np}\}$, y en la configuración $\mathcal{C}_{n+3np}$ hay 2^n membranas etiquetadas por 1, tales que cada una de ellas contiene m objetos $c_{j,t}$ $(1 \leq j \leq p,\ 0 \leq t \leq np-1)$; es decir, las cláusulas satisfechas por*

alguna variable y $n+np-m$ objetos #; y 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene un objeto γ_{n+3np}.

Demostración. (a) será demostrado por inducción sobre l

- El caso base $l = 0$ será demostrado por inducción sobre k
  - El caso base $k = 1$ es trivial dado que:
    - en la configuración $\mathcal{C}_{n+2np}$ tenemos que: $\mathcal{C}_{n+2np}(0) = \{\alpha_{n+2np}, \beta_{n+2np}\}$ y hay 2^n membranas etiquetadas por 1, tales que cada una de ellas contiene; y 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene n objetos # el multiconjunto de entrada $cod_{n+2np}(\varphi)$, un objeto γ_{n+2np} y p copias de objetos T_i y F_i, $1 \leq i \leq n$, que representan la valoración de verdad correspondiente a dicha rama. Por tanto, la configuración $\mathcal{C}_{n+2np}$ da lugar a la configuración $\mathcal{C}_{n+2np+1}$ por la aplicación de las reglas:
      $[\, T_1 x_{1,1,n+2np} \,]_2 \to c_{1,0} [\ \ ]_2$
      $[\, T_1 \overline{x}_{1,1,n+2np} \,]_2 \to \# [\ \ ]_2$
      $[\, T_1 x^*_{1,1,n+2np} \,]_2 \to \# [\ \ ]_2$ [14]
      $[\, F_1 x_{1,1,n+2np} \,]_2 \to \# [\ \ ]_2$
      $[\, F_1 \overline{x}_{1,1,n+2np} \,]_2 \to c_{1,0} [\ \ ]_2$
      $[\, F_1 x^*_{1,1,n+2np} \,]_2 \to \# [\ \ ]_2$
      $[\, \alpha_{n+2np} \to \alpha_{n+2np+1} \,]_0$
      $[\, \beta_{n+2np} \to \beta_{n+2np+1} \,]_0$
      $[\, \gamma_{n+2np} \to \gamma_{n+2np+1} \,]_2$
      $\left.\begin{array}{l} [\, x_{i,j,n+2np} \to x_{i,j,n+2np+1} \,]_2 \\ [\, \overline{x}_{i,j,n+2np} \to \overline{x}_{i,j,n+2np+1} \,]_2 \\ [\, x^*_{i,j,n+2np} \to x^*_{i,j,n+2np+1} \,]_2 \end{array}\right\}$ para $1 \leq i \leq n, 1 \leq j \leq p$

      Por lo tanto, $\mathcal{C}_{n+2np+1}(0) = \{\alpha_{n+2np+1}, \beta_{n+2np+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen n objetos # y un objeto $c_{1,0}$ si la valoración de verdad correspondiente hace verdadera la cláusula 1 con la variable 1, otro objeto # de lo contrario; y 2^n membranas etiquetadas por 2 que contienen los últimos $n-1$ elementos de $cod_{n+2np+1}(\varphi)^1_1$, el multiconjunto de entrada $cod_{n+2np+1}(\varphi)^p_2$, p copias de T_i o F_i, siendo $2 \leq i \leq n$, y $p-1$ copias de T_1 o F_1.
  - Suponiendo, por inducción, que el resultado es cierto para k $(1 \leq k \leq n)$

[14] Si $k = 1, l = 0$, entonces $i = 1, j = 1$, así que $2np + n + n(j-1) + (i-1) = n + 2np$.

- $\mathcal{C}_{n+2np+k}(0) = \{\alpha_{n+2np+k}, \beta_{n+2np+k}\}$
- En la configuración $\mathcal{C}_{n+2np+k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
  - m objetos $c_{1,t}$ $(0 \leq t \leq k-1)$; es decir, el número de variables que hace verdadera la fórmula φ con la valoración de verdad correspondiente; y
  - $n + k - m$ objetos #.
- En la configuración $\mathcal{C}_{n+2np+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
  - los $(n-k)$-ésimos últimos elementos de $cod_{n+2np+k}(\varphi)_1^1$;
  - el multiconjunto de entrada $cod_{n+2np+k}(\varphi)_2^p$;
  - un objeto $\gamma_{n+2np+k}$; y
  - p copias de objetos T_i o F_i, $k+1 \leq i \leq n$, $p-1$ copias si $1 \leq i \leq k$, correspondiendo a la valoración de verdad asignada a la rama.

  Por tanto, la configuración $\mathcal{C}_{n+2np+k}$ da lugar a la configuración $\mathcal{C}_{n+2np+k+1}$ por la aplicación de las reglas:

$$
\begin{array}{l}
[\, T_k x_{1,1,n+2np+k} \,]_2 \to c_{1,0} [\quad]_2 \\
[\, T_k \overline{x}_{1,1,n+2np+k} \,]_2 \to \# [\quad]_2 \\
[\, T_k x^*_{1,1,n+2np+k} \,]_2 \to \# [\quad]_2 \qquad {}^{15} \\
[\, F_k x_{1,1,n+2np+k} \,]_2 \to \# [\quad]_2 \\
[\, F_k \overline{x}_{1,1,n+2np+k} \,]_2 \to c_{1,0} [\quad]_2 \\
[\, F_k x^*_{1,1,n+2np+k} \,]_2 \to \# [\quad]_2 \\
[\, \alpha_{n+2np+k} \to \alpha_{n+2np+k+1} \,]_0 \\
[\, \beta_{n+2np+k} \to \beta_{n+2np+k+1} \,]_0 \\
[\, \gamma_{n+2np+k} \to \gamma_{n+2np+k+1} \,]_2 \\
\left.
\begin{array}{l}
[\, x_{i,j,n+2np+k} \to x_{i,j,n+2np+k+1} \,]_2 \\
[\, \overline{x}_{i,j,n+2np+k} \to \overline{x}_{i,j,n+2np+k+1} \,]_2 \\
[\, x^*_{i,j,n+2np+k} \to x^*_{i,j,n+2np+k+1} \,]_2
\end{array}
\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p \\
[\, c_{1,t} \to c_{1,t+1} \,]_1 \text{ para } 0 \leq t \leq k-1
\end{array}
$$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{n+2np+k+1} = \{\alpha_{n+2np+k+1}, \beta_{n+2np+k+1}\}$
- En la configuración $\mathcal{C}_{n+2np+k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene

[15] Si $l = 0$, entonces $i = k+1, j = 1$, así que $2np + 2n + n(j-1) + (i-1) = 2n + 2np + k$.

◇ m objetos $c_{1,t}$ $(0 \leq t \leq k)$; es decir, el número de variables que hace verdadera la cláusula C_1 con la valoración de verdad correspondiente; y

◇ $n + k + 1 - m$ objetos #.

○ En la configuración $\mathcal{C}_{n+2np+k+1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene

◇ los $(n-k+1)$-ésimos últimos elementos de $cod_{n+2np+k+1}(\varphi)_1^1$;

◇ el multiconjunto de entrada $cod_{n+2np+k+1}(\varphi)_2^p$,

◇ un objeto $\gamma_{n+2np+k+1}$; y

◇ p copias de objetos T_i o F_i, $k+2 \leq i \leq n$, $p-1$ copias si $1 \leq i \leq k+1$, correspondiendo a la valoración de verdad asignada a la rama.

▪ Suponiendo, por inducción, que el resultado es cierto para l $(0 \leq l \leq p-1)$

• El caso base $k = 1$ es trivial dado que:

○ en la configuración $\mathcal{C}_{n+2np+(l+1)n}$ tenemos que: $\mathcal{C}_{n+2np+(l+1)n}(0) =$

$\{\alpha_{n+2np+(l+1)n}, \beta_{n+2np+(l+1)n}\}$ y hay 2^n membranas etiquetadas por 1 que contienen m objetos $c_{j,t}$ $(1 \leq j \leq l,\ 0 \leq t \leq ln-1)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_l con la valoración de verdad correspondiente y $n + (l+1)n - m$ objetos #; y 2^n membranas etiquetadas por 2 que contienen el multiconjunto de entrada $cod_{n+2np+(l+1)n}(\varphi)_{l+1}^p$, un objeto $\gamma_{n+2np+(l+1)n}$ y $p-l$ copias de objetos T_i o F_i, $1 \leq i \leq n$. Por tanto, la configuración $\mathcal{C}_{n+2np+(l+1)n}$ da lugar a la configuración $\mathcal{C}_{n+2np+(l+1)n+1}$ por la aplicación de las reglas:

$[\, T_1 x_{1,1,n+2np+(l+1)n} \,]_2 \to c_{l+1,0} [\ \ ]_2$
$[\, T_1 \overline{x}_{1,1,n+2np+(l+1)n} \,]_2 \to \# [\ \ ]_2$
$[\, T_1 x^*_{1,1,n+2np+(l+1)n} \,]_2 \to \# [\ \ ]_2$
$[\, F_1 x_{1,1,n+2np+(l+1)n} \,]_2 \to c_{l+1,0} [\ \ ]_2$
$[\, F_1 \overline{x}_{1,1,n+2np+(l+1)n} \,]_2 \to \# [\ \ ]_2$
$[\, F_1 x^*_{1,1,n+2np+(l+1)n} \,]_2 \to \# [\ \ ]_2$
$[\, \alpha_{n+2np+(l+1)n} \to \alpha_{n+2np+(l+1)n+1} \,]_0$
$[\, \beta_{n+2np+(l+1)n} \to \beta_{n+2np+(l+1)n+1} \,]_0$
$[\, \gamma_{n+2np+(l+1)n} \to \gamma_{n+2np+(l+1)n+1} \,]_2$

$$\left.\begin{array}{l}[\, x_{i,j,n+2np+(l+1)n} \to x_{i,j,n+2np+(l+1)n+1}\,]_2 \\ [\, \overline{x}_{i,j,n+2np+(l+1)n} \to \overline{x}_{i,j,n+2np+(l+1)n+1}\,]_2 \\ [\, x^*_{i,j,n+2np+(l+1)n} \to x^*_{i,j,n+2np+(l+1)n+1}\,]_2 \end{array}\right\} \text{para } \begin{array}{l} 1 \le i \le n \\ 1 \le j \le p \end{array}$$
$$[\, c_{j,t} \to c_{1,t+1}\,]_1 \text{ para } 1 \le j \le l+1, 0 \le t \le ln-1$$

Por lo tanto, $\mathcal{C}_{n+2np+(l+1)n+1}(0) = \{\alpha_{n+2np+(l+1)n+1}, \beta_{n+2np+(l+1)n+1}\}$, y hay 2^n membranas etiquetadas por 1 que contienen m objetos $c_{j,t}$ $(1 \le j \le l+1,\, 0 \le t \le ln)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_{l+1} con la valoración de verdad correspondiente y $n + (l+1)n + 1 - m$ objetos #; y 2^n membranas etiquetadas por 2 que contienen los últimos $n-1$ elementos de $cod_{n+2np+(l+1)n+1}(\varphi)^{l+1}_{l+1}$, el multiconjunto de entrada $cod_{n+2np+(l+1)n+1}(\varphi)^{p}_{l+2}$, $p-l$ copias de T_i o F_i, siendo $2 \le i \le n$, y $p-l-1$ copias de T_1 o F_1.

- Suponiendo, por inducción, que el resultado es cierto para k $(1 \le k \le n)$
  - $\mathcal{C}_{n+2np+(l+1)n+k}(0) = \{\alpha_{n+2np+(l+1)n+k}, \beta_{n+2np+(l+1)n+k}\}$
  - En la configuración $\mathcal{C}_{n+2np+(l+1)n+k}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
    - m objetos $c_{j,t}$ $(1 \le j \le l+1,\ 0 \le t \le ln+k-1)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_{l+1} con la valoración de verdad correspondiente; y
    - $n + (l+1)n + k + 1 - m$ objetos #.
  - En la configuración $\mathcal{C}_{n+2np+(l+1)n+k}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
    - los $(n-k)$-ésimos últimos elementos de $cod_{n+2np+(l+1)n+k}(\varphi)^{l+1}_{l+1}$;
    - el multiconjunto de entrada $cod_{n+2np+(l+1)n+k}(\varphi)^{p}_{l+2}$,
    - un objeto $\gamma_{n+2np+(l+1)n+k}$; y
    - $p-l$ copias de objetos T_i o F_i, $k+1 \le i \le n$, $p-l-1$ copias si $1 \le i \le k$, correspondiendo a la valoración de verdad asignada a la rama.

    Por tanto, la configuración $\mathcal{C}_{n+2np+(l+1)n+k}$ da lugar a la configuración $\mathcal{C}_{n+2np+(l+1)n+k+1}$ por la aplicación de las reglas:

$[\, T_k x_{1,1,n+2np+(l+1)n+k} \,]_2 \to c_{l+1} [\quad]_2$
$[\, T_k \overline{x}_{1,1,n+2np+(l+1)n+k} \,]_2 \to \# [\quad]_2$
$[\, T_k x^*_{1,1,n+2np+(l+1)n+k} \,]_2 \to \# [\quad]_2$
$[\, F_k x_{1,1,n+2np+(l+1)n+k} \,]_2 \to \# [\quad]_2$
$[\, F_k \overline{x}_{1,1,n+2np+(l+1)n+k} \,]_2 \to c_{l+1} [\quad]_2$
$[\, F_k x^*_{1,1,n+2np+(l+1)n+k} \,]_2 \to \# [\quad]_2$
$[\, \alpha_{n+2np+(l+1)n+k} \to \alpha_{n+2np+(l+1)n+k+1} \,]_0$
$[\, \beta_{n+2np+(l+1)n+k} \to \beta_{n+2np+(l+1)n+k+1} \,]_0$
$[\, \gamma_{n+2np+(l+1)n+k} \to \gamma_{n+2np+(l+1)n+k+1} \,]_2$
$\left.\begin{array}{l} [\, x_{i,j,n+2np+(l+1)n+k} \to x_{i,j,n+2np+(l+1)n+k+1} \,]_2 \\ [\, \overline{x}_{i,j,n+2np+(l+1)n+k} \to \overline{x}_{i,j,n+2np+(l+1)n+k+1} \,]_2 \\ [\, x^*_{i,j,n+2np+(l+1)n+k} \to x^*_{i,j,n+2np+(l+1)n+k+1} \,]_2 \end{array}\right\}$ para $\begin{array}{r} 1 \le i \le n \\ 1 \le j \le p \end{array}$
$[\, c_{j,t} \to c_{j,t+1} \,]_1$ para $1 \le j \le l+1, 0 \le t \le ln+k-1$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{n+2np+(l+1)n+k+1}(0) = \{\alpha_{n+2np+(l+1)n+k+1}, \beta_{n+2np+(l+1)n+k+1}\}$
- En la configuración $\mathcal{C}_{n+2np+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
  - m objetos $c_{j,t}$ $(1 \le j \le l+1,\ 0 \le t \le ln+k)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_{l+1} con la valoración de verdad correspondiente; y
  - $n + (l+1)n + k + 1 - m$ objetos $\#$.
- En la configuración $\mathcal{C}_{n+2np+(l+1)n+k+1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
  - los $(n-(k+1))$-ésimos últimos elementos de $cod_{n+2np+(l+1)n+k+1}(\varphi)^{l+1}_{l+1}$,
  - el multiconjunto de entrada $cod_{n+2np+(l+1)n+k+1}(\varphi)^{p}_{l+1}$,
  - un objeto $\gamma_{n+2np+(l+1)n+k+1}$;
  - $p-l$ copias de objetos T_i o F_i, $k+2 \le i \le n$, $p-l-1$ copias si $1 \le i \le k+1$, correspondiendo a la valoración de verdad asignada a la rama.

- Para probar (b) basta darse cuenta que, por una parte, por (a) la configuración $\mathcal{C}_{n+3np-1}$ [16] es válido:

  - $\mathcal{C}_{n+3np-1}(0) = \{\alpha_{n+3np-1}, \beta_{n+3np-1}\}$

[16] Obsérvese que $n + 3np - 1 = n + 3n(p-1) + (n-1)$

- En la configuración $\mathcal{C}_{n+3np-1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
  - m objetos $c_{j,t}$ $(1 \leq j \leq p,\ 0 \leq t \leq np-2)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_p con la valoración de verdad correspondiente; y
  - $n + np - 1 - m$ objetos #.
- En la configuración $\mathcal{C}_{n+3np-1}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
  - el último elemento de $cod_{n+3np-1}(\varphi)_p^p$;
  - un objeto $\gamma_{n+3np-1}$; y
  - un objeto T_n o F_n correspondiendo a la valoración de verdad asignada a la rama.

  Por tanto, la configuración $\mathcal{C}_{n+3np-1}$ da lugar a la configuración $\mathcal{C}_{n+3np}$ por la aplicación de las reglas:

$$
\begin{array}{l}
[\, T_n x_{n,p,n+3np-1} \,]_2 \to c_{p,0} [\ \ ]_2 \\
[\, T_n \overline{x}_{n,p,n+3np-1} \,]_2 \to \# [\ \ ]_2 \\
[\, T_n x^*_{n,p,n+3np-1} \,]_2 \to \# [\ \ ]_2 \\
[\, F_n x_{n,p,n+3np-1} \,]_2 \to \# [\ \ ]_2 \\
[\, F_n \overline{x}_{n,p,n+3np-1} \,]_2 \to c_{p,0} [\ \ ]_2 \\
[\, F_n x^*_{n,p,n+3np-1} \,]_2 \to \# [\ \ ]_2 \\
[\, \alpha_{n+2np+(l+1)n+k} \to \alpha_{n+2np+(l+1)n+k+1} \,]_0 \\
[\, \beta_{n+2np+(l+1)n+k} \to \beta_{n+2np+(l+1)n+k+1} \,]_0 \\
[\, \gamma_{n+2np+(l+1)n+k} \to \gamma_{n+2np+(l+1)n+k+1} \,]_2 \\
\left.
\begin{array}{l}
[\, x_{i,j,n+2np+(l+1)n+k} \to x_{i,j,n+2np+(l+1)n+k+1} \,]_2 \\
[\, \overline{x}_{i,j,n+2np+(l+1)n+k} \to \overline{x}_{i,j,n+2np+(l+1)n+k+1} \,]_2 \\
[\, x^*_{i,j,n+2np+(l+1)n+k} \to x^*_{i,j,n+2np+(l+1)n+k+1} \,]_2
\end{array}
\right\} \text{para } \begin{array}{c} 1 \leq i \leq n \\ 1 \leq j \leq p \end{array} \\
[\, c_{j,t} \to c_{j,t+1} \,]_2 \text{ para } 1 \leq j \leq l+1, 0 \leq t \leq ln+k-1
\end{array}
$$

  Por lo tanto, lo siguiente es válido
  - $\mathcal{C}_{n+3np}(0) = \{\alpha_{n+3np}, \beta_{n+3np}\}$
  - En la configuración $\mathcal{C}_{n+3np}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene
    - m objetos $c_{j,t}$ $(1 \leq j \leq p,\ 0 \leq t \leq np-1)$; es decir, el número de variables que hace verdaderas desde la cláusula C_1 hasta la cláusula C_p con la valoración de verdad correspondiente; y
    - $n + np - m$ objetos #.

- En la configuración $\mathcal{C}_{n+3np}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene un objeto γ_{n+3np}.

□

Segunda fase de chequeo

En esta fase, que empieza en la configuración $\mathcal{C}_{n+3np}$, se podrá ver si hay alguna valoración de verdad que haga verdadera la fórmula φ, usando las reglas de 3.1. Dividiremos esta fase en dos partes. La primera estará destinada a enviar «hacia dentro» todos los objetos c_j, para $1 \leq j \leq p$ para prepararlos para la siguiente parte.

Sea $k = ln + i$ $(0 \leq l \leq p-1, 1 \leq i \leq n)$, así podremos referirnos a cada cláusula $(l+1)$ cuando estemos viendo la verificación. Sea $m = \sum_{j=1}^{p} m_j$, siendo m_j el número de objetos $c_{j,k}$ en cada membrana etiquetada por 2 en la configuración $\mathcal{C}_{n+3np}$. En esta parte, no podemos asegurar cuantos objetos $c_{l+1,k}$ están presentes en cada membrana cuando $i \neq 0$ [17], así que al no poder estar seguros, diremos que habrá $\widetilde{m}_j$ objetos en una membrana etiquetada por 2 (recordemos que $\widetilde{m}_j$ es siempre menor que o igual a m_j). Ignoraremos los objetos # dado que no tienen efecto a lo largo de la computación.

Proposición 7.11. *Sea $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ una computación del sistema $\Pi(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$.*

(a) *Para cada k $(1 \leq k \leq np-1)$ en la configuración $\mathcal{C}_{n+3np+k}$ tenemos lo siguiente:*

- $\mathcal{C}_{n+3np+k}(0) = \{\alpha_{n+3np+k}, \beta_{n+3np+k}\}$
- *Hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene $\widetilde{m}_{l+1}$ objetos $c_{l+1,t}$ $((p-1)n+1 \leq t \leq np-1)$ y m_j objetos $c_{j,t}$ $(l+2 \leq j \leq p, ln+i \leq t \leq np-1)$*
- *Hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *un objeto $\gamma_{n+3np+k}$; y*
  - *m_j objetos c_j para $1 \leq j \leq l$ y $m_{l+1} - \widetilde{m}_{l+1}$ objetos c_{l+1}*

[17] Esto es debido a que los objetos $c_{j,k}$ no tienen por qué haber sido creados consecutivamente.

(*b*) $\mathcal{C}_{n+4np}(0) = \{\alpha_{n+4np}, \beta_{n+4np}\}$, *hay* 2^n *membranas vacías etiquetadas por 1; y* 2^n *membranas etiquetadas por 2, tales que cada una de ellas contiene* m *objetos* c_j $(1 \leq j \leq p)$ *y un objeto* γ_{n+4np}.

Demostración. (*a*) será demostrado por inducción sobre k

- El caso base $k = 1$ es trivial dado que: en la configuración $\mathcal{C}_{n+3np}$ tenemos que: $\mathcal{C}_{n+3np}(0) = \{\alpha_{n+3np}, \beta_{n+3np}\}$ y hay 2^n membranas etiquetadas por 1 que contienen m objetos $c_{j,t}$ $(1 \leq j \leq k, 0 \leq t \leq np-1)$; y 2^n membranas etiquetadas por 2 que contienen un objeto γ_{n+3np}. Por tanto, la configuración $\mathcal{C}_{n+3np}$ da lugar a la configuración $\mathcal{C}_{n+3np+1}$ por la aplicación de las reglas:

  $[\,\alpha_{n+3np} \to \alpha_{n+3np+1}\,]_0$
  $[\,\beta_{n+3np} \to \beta_{n+3np+1}\,]_0$
  $[\,\gamma_{n+3np} \to \gamma_{n+3np+1}\,]_2$

  $[\,c_{j,t} \to c_{j,t+1}\,]_1$, para$1 \leq j \leq p$, $0 \leq k \leq np-2$

  $c_{1,np-1}[\quad]_2 \to [\,c_1\,]_2$

  Por lo tanto, $\mathcal{C}_{n+3np+1}(0) = \{\alpha_{n+3np+1}, \beta_{n+3np+1}\}$, y hay 2^n membranas etiquetadas por 2 que contienen $\widetilde{m}_1$ objetos c_1 y m_j objetos c_j $(2 \leq j \leq p)$; y 2^n membranas etiquetadas por 1 que contienen un objeto $\gamma_{n+3np+1}$ y $m_1 - \widetilde{m}_1$ objetos c_1 [18]. Por lo tanto, el resultado es válido para $k = 1$.

- Suponiendo, por inducción, que el resultado es cierto para k $(1 \leq k \leq np-1)$

  - $\mathcal{C}_{n+3np+k}(0) = \{\alpha_{n+3np+k}, \beta_{n+3np+k}\}$
  - En la configuración $\mathcal{C}_{n+3np+k}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene $\widetilde{m}_{l+1}$ objetos $c_{l+1,t}$ $((p-1)n+1 \leq t \leq np-1)$ y m_j objetos $c_{j,t}$ $(l+2 \leq j \leq p, ln+i \leq t \leq np-1)$.
  - En la configuración $\mathcal{C}_{n+3np+k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - un objeto $\gamma_{n+3np+k}$; y
    - m_j objetos c_j para $1 \leq j \leq l$ y $m_{l+1} - \widetilde{m}_{l+1}$ objetos c_{l+1}.

[18] Es decir, si la valoración de verdad de la variable 1 hace verdadera la cláusula 1, entonces un objeto $c_{1,0}$ fue creado en el $(2n + 2np + 1)$-ésimo paso, y será enviado a la membrana etiquetada por 2 correspondiente. Así que, $m_1 - \widetilde{m}_1$ puede ser 0 o 1 en este paso.

Por tanto, la configuración $\mathcal{C}_{n+3np+k}$ da lugar a la configuración $\mathcal{C}_{n+3np+k}$ por la aplicación de las reglas:

$[\,\alpha_{n+3np+k} \to \alpha_{n+3np+k+1}\,]_0$
$[\,\beta_{n+3np+k} \to \beta_{n+3np+k+1}\,]_0$
$[\,\gamma_{n+3np+k} \to \gamma_{n+3np+k+1}\,]_2$
$[\,c_{j,t} \to c_{j,t+1}\,]_1$, para$l+1 \leq j \leq p$, $0 \leq k \leq np-2$
$c_{l+1,np-1}[\quad]_2 \to [\,c_{l+1}\,]_2$

Por lo tanto, lo siguiente es válido

- $\mathcal{C}_{n+3np+k+1}(0) = \{\alpha_{n+3np+k+1}, \beta_{n+3np+k+1}\}$
- En la configuración $\mathcal{C}_{n+3np+k+1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene $\widetilde{m}_{l+1}$ objetos $c_{l+1,t+1}$ $((p-1)n+1 \leq t \leq np-1)$ y m_j objetos $c_{j,t+1}$ $(l+2 \leq j \leq p, ln+i \leq t \leq np-1)$.
- En la configuración $\mathcal{C}_{n+3np+k+1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
  - un objeto $\gamma_{n+3np+k+1}$; y
  - m_j objetos c_j para $1 \leq j \leq l$ y $m_{l+1} - \widetilde{m}_{l+1}$ objetos c_{l+1}.

  Por lo tanto, el resultado es válido para $k+1$.

- Para probar (b) basta darse cuenta que, por una parte, por (a) la configuración $\mathcal{C}_{n+4np-1}$ es válido:

  - $\mathcal{C}_{n+4np-1}(0) = \{\alpha_{n+4np-1}, \beta_{n+4np-1}\}$
  - En la configuración $\mathcal{C}_{n+4np-1}$ hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene
  - En la configuración $\mathcal{C}_{n+4np-1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene $\widetilde{m}_p$ objetos $c_{p,np}$.
    - un objeto $\gamma_{n+4np-1}$; y
    - m_j objetos c_j para $1 \leq j \leq p-1$ y $m_p - \widetilde{m}_p$ [19] objetos c_p.

    Por tanto, la configuración $\mathcal{C}_{n+4np-1}$ da lugar a la configuración $\mathcal{C}_{n+4np}$ por la aplicación de las reglas:

    $[\,\alpha_{n+4np-1} \to \alpha_{n+4np}\,]_0$
    $[\,\beta_{n+4np-1} \to \beta_{n+4np}\,]_0$
    $[\,\gamma_{n+4np-1} \to \gamma_{n+4np}\,]_2$
    $c_{p,np}[\quad]_2 \to [\,c_p\,]_2$

[19] En este caso, $\widetilde{m}_p$ puede tomar sólo dos valores: 0 o 1.

Por tanto, tenemos que $\mathcal{C}_{n+4np}(0) = \{\alpha_{n+4np}, \beta_{n+4np}\}$, y hay 2^n membranas vacías etiquetadas por 1; y hay 2^n membranas etiquetadas por 2 que contienen un objeto γ_{n+4np} y m objetos c_j $(1 \leq j \leq p)$.

□

Cuando los objetos c_j se mueven a las membranas etiquetadas por 2, podemos empezar a chequear si todas las cláusulas de la fórmula φ son satisfechas por alguna valoración de verdad. Como estamos usando los objetos c_j para denotar si la cláusula C_j es satisfecha por alguna variable, es posible que algún objeto c_j puede faltar; es decir, que para algún $j, 1 \leq j \leq p$, c_j no aparece en ninguna membrana etiquetada por 2 en $\mathcal{C}_{n+4np}$. Sea $\widetilde{j}$ el índice j [20] de dicha cláusula. Se hará en $2p$ pasos de computación.

Proposición 7.12. *Sea $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ una computación del sistema $\Pi(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$.*

(a_0) *Para cada $2k+1$ $(0 \leq k \leq p-1)$ en la configuración $\mathcal{C}_{n+4np+2k+1}$ tenemos lo siguiente:*

- $\mathcal{C}_{n+4np+2k+1}(0) = \{\alpha_{n+4np+2k+1}, \beta_{n+4np+2k+1}\}$
- *Hay 2^n membranas etiquetadas por 1 tales que cada una de ellas contiene un objeto d_{k+1} si y sólo si la valoración de verdad asociada a dicha rama hace verdaderas las $k+1$ primeras cláusulas.*
- *Hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene*
  - *un objeto γ_{n+4np} o $d_{\widetilde{j}-1}$ (respectivamente, un objeto d_k) si la valoración de verdad correspondiente no hace verdadera (respectivamente, hace verdadera) la cláusula C_1 o C_j $(2 \leq j \leq p)$ (respectivamente, las k primeras cláusulas); y*
  - *$m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, k+1)$ y m_j objetos c_j para $min(\widetilde{j}, k+2) \leq j \leq p$.*

(a_1) *Para cada $2k$ $(1 \leq k \leq p-2)$ en la configuración $\mathcal{C}_{n+4np+2k}$ tenemos lo siguiente:*

- $\mathcal{C}_{n+4np+2k}(0) = \{\alpha_{n+4np+2k}, \beta_{n+4np+2k}\}$
- *hay 2^n membranas vacías etiquetadas por 1.*

[20] Si $\widehat{j}$ no está definido, suponemos que es igual a $p+1$.

- *Hay 2^n membranas vacías etiquetadas por 2 tales que cada una de ellas contiene*
  - *un objeto γ_{n+4np} o $d_{\widetilde{j-1}}$ si la valoración de verdad correspondiente no hace verdadera la cláusula C_1 o la cláusula C_j ($2 \leq j \leq p$); y*
  - *$m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, k)$ y m_j objetos c_j para $min(\widetilde{j}, k+1) \leq j \leq p$.*

(*b*) *$\mathcal{C}_{n+4np+2p-1}(0) = \{\alpha_{n+4np+2p-1}, \beta_{n+4np+2p-1}\}$, y en la configuración $\mathcal{C}_{n+4np+2p-1}$ hay 2^n membranas etiquetadas por 1, tales que cada una de ellas contiene un objeto d_p si y sólo si la valoración de verdad correspondiente la fórmula φ ($d_{\widetilde{j-1}}$ de lo contrario); y 2^n membranas etiquetadas por 2, tales que cada una de ellas contiene $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, p+1)$, m_j objetos c_j para $min(\widetilde{j}, p+1) \leq j \leq p$ y un objeto γ_{n+4np} (respectivamente, $d_{\widetilde{j}}$) si la cláusula C_1 (respectivamente, C_j) no es satisfecha por la valoración de verdad correspondiente.*

Demostración. (*a*) será demostrado por inducción sobre k

- El caso base $k = 1$ es trivial dado que:

  (a_0) en la configuración $\mathcal{C}_{n+4np}$ tenemos que: $\mathcal{C}_{n+4np}(0) = \{\alpha_{n+4np}, \beta_{n+4np}\}$ y hay 2^n membranas vacías etiquetadas por 1; y hay 2^n membranas etiquetadas por 2 que contienen un objeto γ_{n+4np} y m objetos c_j ($1 \leq j \leq p$). Así, la configuración $\mathcal{C}_{n+4np}$ da lugar a la configuración $\mathcal{C}_{n+4np+1}$ por la aplicación de las reglas:
  $$[\,\alpha_{n+4np} \to \alpha_{n+4np+1}\,]_0$$
  $$[\,\beta_{n+4np} \to \beta_{n+4np+1}\,]_0$$
  $$[\,\gamma_{4np+2n}c_1\,]_2 \to d_1\,[\quad]_2$$

  (a_1) en la configuración $\mathcal{C}_{n+4np+1}(0) = \{\alpha_{n+4np+1}, \beta_{n+4np+1}\}$ y hay 2^n membranas etiquetadas por 1 que contienen un objeto d_1 si y sólo si había al menos un objeto c_1 en dicha membrana etiquetada por 1 en la configuración $\mathcal{C}_{n+4np}$; y 2^n membranas etiquetadas por 2 que contienen un objeto γ_{n+4np} si y sólo si no había objetos c_1 en la configuración $\mathcal{C}_{n+4np}$, $m_1 - 1$ (respectivamente, m_1) objetos c_1 si había algún objeto c_j en dicha membrana en la configuración anterior (respectivamente, m_1) y m_j objetos c_j para $2 \leq j \leq p$. Así, la configuración $\mathcal{C}_{n+4np+1}$ da lugar a la configuración $\mathcal{C}_{n+4np+2}$ por la aplicación de las reglas:

$[\,\alpha_{n+4np+1} \to \alpha_{n+4np+2}\,]_0$
$[\,\beta_{n+4np+1} \to \beta_{n+4np+2}\,]_0$
$d_1\,[\quad]_2 \to [\,d_1\,]_2$

Por lo tanto, $\mathcal{C}_{n+4np+2}(0) = \{\alpha_{n+4np+2}, \beta_{n+4np+2}\}$, y hay 2^n membranas vacías etiquetadas por 1; y hay 2^n membranas etiquetadas por 2 que contienen un objeto d_1 (respectivamente, γ_{n+4np}) si la valoración de verdad correspondiente hace verdadera (respectivamente, no hace verdadera) la cláusula C_1, $m_1 - 1$ (respectivamente, m_1) objetos c_1 y m_j objetos c_j para $1 \leq j \leq p$. Por lo tanto, el resultado es válido para $k = 1$.

- Suponiendo, por inducción, que el resultado es cierto para k ($0 \leq k \leq p-1$)

  - $\mathcal{C}_{n+4np+2k}(0) = \{\alpha_{n+4np+2k}, \beta_{n+4np+2k}\}$
  - En la configuración $\mathcal{C}_{n+4np+2k}$ hay 2^n membranas vacías etiquetadas por 1.
  - En la configuración $\mathcal{C}_{n+4np+2k}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - un objeto γ_{n+4np} o $d_{\widetilde{j-1}}$ (respectivamente, un objeto d_k) si la valoración de verdad correspondiente no hace verdadera (respectivamente, hace verdadera) la cláusula C_1 o C_j ($2 \leq j \leq p$) (respectivamente, las k primeras cláusulas); y
    - $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, k+1)$ y m_j objetos c_j para $min(\widetilde{j}, k+2) \leq j \leq p$.

  Por tanto, la configuración $\mathcal{C}_{n+4np+2k}$ da lugar a la configuración $\mathcal{C}_{n+4np+2k+1}$ por la aplicación de las reglas:

  $[\,\alpha_{n+4np+2k} \to \alpha_{n+4np+2k+1}\,]_0$
  $[\,\beta_{n+4np+2k} \to \beta_{n+4np+2k+1}\,]_0$
  $[\,d_k c_{k+1}\,]_2 \to d_{k+1}\,[\quad]_2$

  Por lo tanto, lo siguiente es válido

  - $\mathcal{C}_{n+4np+2k+1}(0) = \{\alpha_{n+4np+2k+1}, \beta_{n+4np+2k+1}\}$
  - En la configuración $\mathcal{C}_{n+4np+2k+1}$ hay 2^n membranas etiquetadas por 1 tal que cada una de ellas contiene un objeto d_{k+1} si y sólo si la valoración de verdad correspondiente hace verdaderas las $k+1$ primeras cláusulas.

- En la configuración $\mathcal{C}_{n+4np+2k+1}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
  - un objeto γ_{n+4np} o $d_{\widetilde{j-1}}$ si la valoración de verdad correspondiente no hace verdadera la cláusula C_1 o la cláusula C_j $(2 \leq j \leq p)$; y
  - $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, k)$ y m_j objetos c_j para $min(\widetilde{j}, k+1) \leq j \leq p$.

  Por tanto, la configuración $\mathcal{C}_{n+4np+2k+1}$ da lugar a la configuración $\mathcal{C}_{n+4np+2k+2}$ por la aplicación de las reglas:
  $[\, \alpha_{n+4np+2k+1} \rightarrow \alpha_{n+4np+2k+2} \,]_0$
  $[\, \beta_{n+4np+2k+1} \rightarrow \beta_{n+4np+2k+2} \,]_0$
  $d_{k+1}[\quad]_2 \rightarrow [\, d_{k+1} \,]_2$
  Por lo tanto, lo siguiente es válido
- $\mathcal{C}_{n+4np+2k+2}(0) = \{\alpha_{n+4np+2k+2}, \beta_{n+4np+2k+2}\}$
- En la configuración $\mathcal{C}_{n+4np+2k+2}$ hay 2^n membranas vacías etiquetadas por 1.
- En la configuración $\mathcal{C}_{n+4np+2k+2}$ hay 2^n membranas etiquetadas por 2 tales que cada una de ellas contiene
  - un objeto γ_{n+4np} o $d_{\widetilde{j-1}}$ (respectivamente, un objeto d_{k+1}) si la valoración de verdad correspondiente no hace verdadera (respectivamente, hace verdadera) la cláusula C_1 o C_j $(2 \leq j \leq p)$ (respectivamente, las $k+1$ primeras cláusulas); y
  - $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, k+2)$ y m_j objetos c_j para $min(\widetilde{j}, k+3) \leq j \leq p$.

  Por lo tanto, el resultado es válido para $k+1$.

- Para probar (b) basta con observar que, por una parte, de (a), la configuración $\mathcal{C}_{n+4np+2p-2}$ tiene que:
  - $\mathcal{C}_{n+4np+2p-2}(0) = \{\alpha_{n+4np+2p-2}, \beta_{n+4np+2p-2}\}$
  - En la configuración $\mathcal{C}_{n+4np+2p-2}$ hay 2^n membranas vacías etiquetadas por 1.
  - En la configuración $\mathcal{C}_{n+4np+2p-2}$ hay 2^n membranas etiquetadas por 2 tal que cada una de ellas contiene
    - un objeto γ_{n+4np} o $d_{\widetilde{j-1}}$ (respectivamente, d_{p-1}) si la valoración de verdad correspondiente no hace verdadera la cláusula C_1 o la

cláusula C_j $(2 \leq j \leq p-1)$ (respectivamente, hace verdaderas las cláusulas C_j $(1 \leq j \leq p-1)$); y

- $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, p-1)$ y m_j objetos c_j para $min(\widetilde{j}, p) \leq j \leq p$

Por tanto, la configuración $\mathcal{C}_{n+4np+2p-2}$ da lugar a la configuración $\mathcal{C}_{n+4np+2p-1}$ por la aplicación de las reglas:

$[\,\alpha_{n+4np+2p-2} \to \alpha_{n+4np+2p-1}\,]_0$
$[\,\beta_{n+4np+2p-2} \to \beta_{n+4np+2p-1}\,]_0$
$[\,d_{p-1}c_p\,]_2 \to d_p\,[\quad]_2$

Por tanto, tenemos que $\mathcal{C}_{n+4np+2p-1}(0) = \{\alpha_{n+4np+2p-1}, \beta_{n+4np+2p-1}\}$, y en la configuración $\mathcal{C}_{n+4np+2p-1}$ hay 2^n membranas etiquetadas por 1, tales que cada una de ellas contiene un objeto d_p si y sólo si la valoración de verdad correspondiente la fórmula φ ($d_{\widetilde{j-1}}$ de lo contrario); y 2^n membranas etiquetadas por 2, tales que cada una de ellas contiene $m_j - 1$ objetos c_j para $1 \leq j \leq min(\widetilde{j}, p+1)$, m_j objetos c_j para $min(\widetilde{j}, p+1) \leq j \leq p$ y un objeto γ_{n+4np} (respectivamente, $d_{\widetilde{j}}$) si la cláusula C_1 (respectivamente, C_j) no es satisfecha por la valoración de verdad correspondiente.

□

Fase de salida

La fase de salida comienza en la configuración $\mathcal{C}_{n+4np+2p-1}$, y tarda exactamente 2 pasos de computación en caso de tener una respuesta afirmativa y 3 pasos de computación en caso de obtener una respuesta negativa. Las reglas de 4.1 son las usadas para realizar esta fase.

- *Respuesta afirmativa*: En este caso, en la configuración $\mathcal{C}_{n+4np+2p-1}$, en alguna membrana etiquetada por 1 habrá un objeto d_p. Por la aplicación de la regla $[\,d_p\,]_1 \to d_p\,[\quad]_1$ (al mismo tiempo que las reglas $[\,\alpha_{n+4np+2p-1} \to \alpha_{n+4np+2p}\,]_0$ y $[\,\beta_{n+4np+2p-1} \to \beta_{n+4np+2p}\,]_0$ son disparadas), un objeto d_p es enviado a la membrana piel. Entonces, se aplican las reglas $[\,\alpha_{4np+n+2p}d_p\,]_0 \to \texttt{yes}\,[\quad]_0$ y $[\,\beta_{n+4np+2p} \to \beta_{n+4np+2p+1}\,]_0$ y un objeto `yes` será enviado al entorno, y la computación para.

- *Respuesta negativa*: En este caso, en la configuración $\mathcal{C}_{n+4np+2p-1}$, no habrá ninguna membrana etiquetada por 1 que tenga un objeto d_p, así que las únicas reglas que se ejecutarán serán las reglas $[\,\alpha_{n+4np+2p-1} \to$

$\alpha_{n+4np+2p}\,]_0$ y $[\,\beta_{n+4np+2p-1} \to \beta_{n+4np+2p}\,]_0$. La regla $[\,\beta_{n+4np+2p} \to \beta_{n+4np+2p+1}\,]_0$ es disparada en el siguiente paso de computación. Por lo tanto, en la configuración $\mathcal{C}_{n+4np+2p+1}$ en la membrana piel hay una copia del objeto $\alpha_{n+4np+2p}$ y una copia del objeto $\beta_{n+4np+2p+1}$. Finalmente, por la aplicación de la regla $[\,\alpha_{4np+n+2p}\beta_{4np+n+2p+1}\,]_0 \to \texttt{no}[\quad]_0$, un objeto `no` es enviado al entorno y la computación para.

7.1.2.2. Resultado obtenido

Teorema 7.2. $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{DAM}^0(+e_s,mcmp_{out},-d,+n)}$.

Demostración. La familia $\mathbf{\Pi}$ de sistemas P descrita verifica lo siguiente:

(*a*) La familia $\mathbf{\Pi}$ es polinomialmente uniforme por máquinas de Turing, dado que para cada $n, p \in \mathbb{N}$, las reglas de $\Pi(\langle n, p\rangle)$ de la familia son recursivas recursivamente de $n, p \in \mathbb{N}$, y la cantidad de recursos necesarios para construir un elemento de dicha familia es de orden polinomial con respecto a n y p, como se muestra a continuación:

- Tamaño del alfabeto: $\frac{15n^2p^2}{2} + 3n^2p + 3n^2 + np^2 + \frac{35np}{2} + 5n + 6p + 6 \in \Theta(n^2p^2)$.
- Número inicial de membranas: $3 \in \Theta(1)$.
- Número inicial de objetos en membranas: $3np + n + 3 \in \Theta(np)$.
- Número de reglas: $\frac{15n^2p^2}{2} + 7n^2p + np^2 + \frac{33np}{2} + 4n + 6p + 4 \in \Theta(n^2p^2)$.
- Número máximo de objetos envueltos en cualquier regla: $3 \in \Theta(1)$.

(*b*) La familia $\mathbf{\Pi}$ está polinomialmente acotada con respecto a $(\texttt{SAT}, cod, s)$: de hecho, para cada instancia φ de `SAT`, cualquier computación del sistema $\mathbf{\Pi}(s(\varphi))$ con multiconjunto de entrada $cod(\varphi)$ tarda, a lo sumo, $2n + 4np + 2p + 3$ pasos de computación.

(*c*) La familia $\mathbf{\Pi}$ es adecuada con respecto a $(\texttt{SAT}, cod, s)$: de hecho, para cada instancia φ de `SAT`, si la computación de $\mathbf{\Pi}(s(\varphi)) + cod(\varphi)$ es una computación de aceptación, entonces φ es satisfactible.

(*d*) La familia $\mathbf{\Pi}$ es completa con respecto a $(\texttt{SAT}, cod, s)$: de hecho, para cada instancia φ de `SAT` tal que φ sea satisfactible, cualquier computación de $\mathbf{\Pi}(s(\varphi)) + cod(\varphi)$ será una computación de aceptación.

Por lo tanto, la familia $\mathbf{\Pi}$ de sistemas P descrita anteriormente resuelve el problema `SAT` en tiempo polinomial de manera uniforme.

□

Corolario 7.2. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{DAM}^0(+e_s,mcmp_{out},-d,+n)}$.

Demostración. Basta notar que el problema SAT es un problema **NP**-completo, $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{DAM}^0(+e_s,mcmp_{out},-d,+n)}$, y la clase de complejidad $\mathbf{PMC}_{\mathcal{DAM}^0(+e_s,mcmp_{out},-d,+n)}$ es cerrada bajo reducibilidad en tiempo polinomial y bajo complementario.

□

7.1.3. Nuevas fronteras obtenidas

Al igual que en el Capítulo 6, donde se había encontrado una frontera con respecto a los sistemas P con membranas activas sin polarizaciones clásicos, donde la cooperación entre objetos no estaba permitida, en el actual capítulo se reseña que una cooperación minimal en alguno de los tipos de comunicación es suficiente para obtener soluciones eficientes de problemas presuntamente intratables. Es decir, en el marco $\mathcal{DAM}^0(+e_s, \beta, -d, +n)$, pasar de usar reglas de comunicación no cooperativas a usar cooperación minimal en cualquiera de los dos tipos corresponde a pasar de la no eficiencia a la presumible eficiencia.

7.2. $\mathcal{SAM}^0(-d)$: Cooperación en reglas de comunicación

En esta sección, se mostrarán varios resultados obtenidos a lo largo de la elaboración del presente documento, remarcando la clara diferencia entre las reglas de división y las de separación cuando se usa cooperación en las reglas de separación, y la importancia que parece tener la generación de un espacio exponencial en términos tanto de objetos como de membranas.

Definición 7.2. *Un sistema P con membranas activas sin polarización con cooperación minimal en reglas de comunicación y separación de grado* $q \geq 1$ *es una tupla* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, H, H_0, H_1, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$*, donde* Π *es un sistema P con membranas activas sin polarización y reglas de separación y reglas de evolución cooperativas, con* $H_0 \cup H_1 = H$ *y* $H_0 \cap H_1 = \emptyset$.

Para denotar las familias de sistemas de membranas estudiados aquí, usamos $\mathcal{SAM}^0(\alpha, \beta, \delta, \gamma)$ con los parámetros $\alpha, \beta, \delta, \gamma$ especificados en la sección anterior.

A continuación se demostrará que, el uso de la cooperación en reglas de comunicación cuando se cambian las reglas de división por reglas de separación

no es suficiente para alcanzar la presumible eficiencia cuando se usan *reglas de evolución simple* incluso usando reglas de disolución y reglas de separación para membranas elementales y no elementales.

7.2.1. No eficiencia de los sistemas P de $\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)$

A continuación se demostrará, usando la técnica algorítmica, la no eficiencia de los sistemas P con membranas activas, reglas de evolución simple, comunicación con cooperación, disolución y separación de membranas elementales y no elementales.

Sea $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \Sigma, H, H_0, H_1, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$ un sistema de $\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)$. Previo a la demostración, se van a introducir algunos conceptos que se usarán a lo largo de la misma.

- Denotaremos por $p(i)$ (respectivamente, $ch(i)$) la etiqueta de la membrana padre (respectivamente, una membrana hija) de la membrana etiquetada por i, teniendo presente que el padre de la membrana piel es el entorno, y lo denotamos con $p(1) = 0$. Denotamos con $\mathcal{R}_E$ (respectivamente, $\mathcal{R}_C$, $\mathcal{R}_D$ y $\mathcal{R}_S$) el conjunto de reglas de evolución (respectivamente, comunicación, disolución y separación) de Π. Fijamos un orden total en $\mathcal{R}_E$, $\mathcal{R}_C$, $\mathcal{R}_D$ y $\mathcal{R}_S$.

- Sea $\mathcal{C}$ una computación de Π y $\mathcal{C}_t$ una configuración arbitraria de $\mathcal{C}$. Con respecto al número de objetos del sistema, debemos observar que al aplicar una regla, el número de objetos se mantiene igual o decrece en 1 (por la aplicación de reglas de separación de membranas elementales). Por lo tanto, el número total de objetos en $\mathcal{C}_t$ es, a lo sumo, M, siendo $M = |\mathcal{M}_0 + \cdots + \mathcal{M}_q|$.

  Con respecto al número de membranas del sistema, mediante la aplicación de reglas de separación para membranas elementales, se elimina un objeto del sistema, no se añade ningún objeto nuevo y se crea una nueva membrana. Por lo tanto, a lo sumo M membranas pueden ser creadas por este proceso. Además, mediante la aplicación de reglas de separación de membranas no elementales, el número de objetos se mantiene intacto pero se crea una nueva membrana (cuando una regla de separación de membranas no elementales es aplicada a una membrana, no podrá ser aplicada a la misma nunca más a lo largo de la computación). De esta manera, no más de $q - 2$ membranas nuevas pueden ser creadas de esta

forma. Consecuentemente, $q + M + (q - 2) = M + 2q - 2$ se una cota superior del número de membranas en $\mathcal{C}_t$ [21].

- Para identificar las membranas creadas por la aplicación de las reglas de separación, modificaremos las etiquetas de las nuevas membranas creadas de forma recursiva como sigue:
  - La etiqueta de una membrana será un par (i, σ) donde $0 \leq i \leq q$ y $\sigma \in \{0, 1\}^*$. En la configuración inicial, todas las membranas son de la forma $(1, \lambda), \ldots, (q, \lambda)$. La etiqueta del entorno será representada por $(0, \lambda)$.
  - Si se aplica una regla de separación a una membrana etiquetada por (i, σ), las nuevas membranas creadas serán etiquetadas una con $(i, \sigma 0)$ y la otra con $(i, \sigma 1)$. La membrana $(i, \sigma 0)$ contendrá únicamente los objetos de la membrana original (i, σ) que pertenezcan a Γ_0 y la membrana $(i, \sigma 1)$ únicamente los objetos de la membrana original (i, σ) que pertenezcan a Γ_1. Si la membrana (i, σ) era una membrana no elemental, además de lo anterior, se entenderá que la membrana $(i, \sigma 0)$ contendrá las membranas de la membrana original que pertenezcan a H_0 y la membrana $(i, \sigma 1)$ contendrá las membranas de la membrana original que pertenezcan a H_1.
  - Si se aplica una regla de evolución o de comunicación, a la membrana etiquetada por (i, σ), la etiqueta no cambia.
- Nótese que el número de etiquetas utilizadas para identificar a todas las membranas a lo largo de toda la computación del sistema es del orden $O(M + q)$.
- Una configuración $\mathcal{C}_t$ del sistema P de $\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)$ es descrita por la estructura de membranas del sistema en el instante t y por el multiconjunto de objetos del tipo:

$$\{(a, i, \sigma) \mid a \in \Gamma, 0 \leq i \leq q, \sigma \in \{0, 1\}^*\}$$

  Decir que $(a, i, \sigma) \in \mathcal{C}_t$ significa que el objeto a pertenece a la membrana etiquetada por (i, σ) en el instante t.

[21] Podría darse el caso de poder realizarse dos veces la misma regla de separación de membranas no elementales a la misma membrana, si se ha disparado alguna regla de disolución a alguna hija de dicha membrana. Pero, dado el caso, el número de membranas disminuiría, al menos, en 1 por la aplicación de dicha regla. Por lo tanto, la cota superior se seguiría manteniendo.

- Sea $r = [\, a \to b \,]_i \in \mathcal{R}$ una regla de evolución del sistema Π. Denotaremos con $n \cdot LHS(r,(i,\sigma)), n \in \mathbb{N}$ el multiconjunto de objetos $(a,i,\sigma)^n$. Denotamos con $n \cdot RHS(r,(i,\sigma)), n \in \mathbb{N}$ el multiconjunto de objetos $(b,i,\sigma)^n$ producidos al aplicar n veces la regla r en la membrana (i,σ).

- Sea $r = [\, ab \,]_i \to c\,[\ \ ]_i \in \mathcal{R}$ una regla de comunicación hacia fuera del sistema Π. Denotaremos con $LHS(r,(i,\sigma))$ los objetos $(a,i,\sigma)(b,i,\sigma)$. Denotamos con $RHS(r,(i,\sigma))$ el objeto $(c,p(i),\tau)$ producido al aplicar la regla r en la membrana (i,σ), donde $(p(i),\tau)$ es la membrana padre de la membrana (i,σ). Este concepto es aplicado de manera similar en las reglas del tipo $[\, a \,]_i \to c\,[\ \ ]_i$.

- Sea $r = ab\,[\ \ ]_i \to [\, c \,]_i \in \mathcal{R}$ una regla de comunicación hacia dentro del sistema Π. Denotaremos con $LHS(r,(i,\sigma))$ los objetos $(a,p(i),\tau)$ $(b,p(i),\tau)$. Denotamos con $RHS(r,(i,\sigma))$ el objeto (c,i,σ) producido al aplicar la regla r en la membrana (i,σ), donde $(p(i),\tau)$ es la membrana padre de la membrana (i,σ). Este concepto es aplicado de manera similar en las reglas del tipo $a\,[\ \ ]_i \to [\, b \,]_i$.

- Sea $\mathcal{C}_t$ una configuración del sistema Π, denotamos con $\mathcal{C}_t + \{(x,i,\sigma)/\sigma'\}$ el multiconjunto obtenido de reemplazar en $\mathcal{C}_t$ toda aparición de (x,i,σ) por (x,i,σ'). De manera similar, denotamos con $\mathcal{C}_t + \{(x,i,\sigma)/(i',\sigma')\}$ el multiconjunto obtenido de reemplazar en $\mathcal{C}_t$ toda aparición de (x,i,σ) por (x,i',σ'). Además, usaremos $\mathcal{C}_t + m$ (respectivamente, $\mathcal{C}_t \backslash m$) para denotar que el multiconjunto de objetos m se ha añadido (respectivamente, eliminado) de la configuración $\mathcal{C}_t$.

A continuación, se describirá un algoritmo determinista $\mathcal{A}$ que se ejecuta en tiempo polinomial, el cuál recibe como entrada un sistema P reconocedor Π de $\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)$ junto con un multiconjunto de entrada m. El algoritmo $\mathcal{A}$ reproduce el funcionamiento de una computación de dicho sistema con el multiconjunto de entrada.

El pseudocódigo está descrito en el Algoritmo 7.2.1.

La fase de selección y la de ejecución simulan un paso de transición del sistema P Π. Específicamente, la fase de selección recibe como entrada una configuración $\mathcal{C}_t$ de Π en el instante t. La salida es un par $(\mathcal{C}'_t, A)$, donde A es un multiconjunto de reglas que son aplicadas en el paso $t+1$ en Π, y $\mathcal{C}'_t$ es la configuración $\mathcal{C}_t$ cuando los objetos correspondientes a la aplicación de las reglas del multiconjunto A son eliminados. La fase de ejecución recibe como entrada el par $(\mathcal{C}'_t, A)$ de la fase de selección y devuelve la siguiente

Algoritmo 7.2.1 Algoritmo $\mathcal{A}$ de simulación de un sistema Π de $\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)$

Entrada: un sistema P Π de $\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)$ y un multiconjunto de entrada m

Salida: *Sí* si $\mathcal{C}_t$ es una configuración de aceptación, *No* en cualquier otro caso

Fase de inicialización: la configuración inicial $\mathcal{C}_0$ de $\Pi + m$
$t \leftarrow 0$
mientras $\mathcal{C}_t$ no se una configuración de parada **hacer**
  Fase de selección: entrada $\mathcal{C}_t$, salida $(\mathcal{C}'_t, A)$
  Fase de ejecución: entrada $(\mathcal{C}'_t, A)$, salida $\mathcal{C}_{t+1}$
  $t \leftarrow t + 1$
fin mientras

configuración $\mathcal{C}_{t+1}$. Específicamente, $\mathcal{C}'_t$ da lugar a $\mathcal{C}_{t+1}$ añadiendo los objetos producidos por la aplicación de las reglas de A.

El Algoritmo 7.2.2 es la descripción del algoritmo de selección. Este algoritmo es determinista y es ejecutado en tiempo polinomial. De hecho, el coste es en tiempo polinomial con respecto al tamaño de Π dado que el número de ciclos del bucle principal es de orden $O(M + q)$, y el número de ciclos de los tres bucles internos son de orden $O(|\mathcal{R}|)$. Además, el coste de cada uno de los bucles internos es de orden $O(M + q)$, dado que cada regla r asociada a una membrana i ya que puede haber varias membranas con la misma etiqueta i.

Observemos que el número de tuplas en el multiconjunto A es de orden $O(M)$ dado que cada objeto del sistema puede estar envuelto en, a lo sumo, una regla y en cada configuración $\mathcal{C}_t$ el número total de objetos está acotado por M. En A se considera un orden natural (producto de las reglas, objetos que intervienen en las mismas y etiquetas de las membranas asociadas). Para completar la simulación de un paso de computación, en el Algoritmo 7.2.3 se tienen en cuenta los efectos de las reglas seleccionadas en la fase previa, añadiendo los objetos correspondientes a la parte derecha de las reglas y realizando las acciones oportunas en caso que sea necesario. Dicho algoritmo es determinista y se ejecuta en tiempo polinomial. De hecho, por una parte, el número de ciclos del bucle principal es de orden $O(M)$. Por otra parte, los bucles internos tienen un orden $O(M^2 + Mq)$ y $O(M + q)$, respectivamente.

Teorema 7.3. $\mathbf{P} = \mathbf{PMC}_{\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)}$

Demostración. Basta probar que $\mathbf{PMC}_{\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)} \in \mathbf{P}$. Para ello, sea $X = (I_X, \theta_X)$ un problema de decisión resoluble eficientemente por familias de sistemas de $\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)$. Sea $\mathbf{\Pi} = \{\Pi(n) \mid n \in$

Algoritmo 7.2.2 Algoritmo de la fase de selección

Entrada: Una configuración $\mathcal{C}_t$ de Π en el instante t
Salida: Un par $(\mathcal{C}'_t, A)$

$\mathcal{C}'_t \leftarrow \mathcal{C}_t$; $A \leftarrow \emptyset$; $B \leftarrow \emptyset$
para cada membrana (i, σ) de $\mathcal{C}'_t$ de acuerdo al orden lexicográfico **hacer**
  para cada $r \in \mathcal{R}_E$ de acuerdo al orden escogido **hacer**
    $n_r \leftarrow$ número máximo de veces que r es aplicable a (i, σ)
    si $n_r > 0$ **entonces**
      $\mathcal{C}'_t \leftarrow \mathcal{C}'_t \setminus n_r \cdot LHS(r, (i, \sigma))$
      $A \leftarrow A \cup \{(r, n_r, (i, \sigma))\}$
    fin si
  fin para
  para cada $r \in \mathcal{R}_C$ de acuerdo al orden escogido **hacer**
    si $(i, \sigma) \notin B$ y r es aplicable a (i, σ) en $\mathcal{C}'_t$ **entonces**
      $\mathcal{C}'_t \leftarrow \mathcal{C}'_t \setminus n_r \cdot LHS(r, (i, \sigma))$
      $A \leftarrow A \cup \{(r, 1, (i, \sigma))\}$
      $B \leftarrow B \cup \{(i, \sigma)\}$
    fin si
  fin para
  para cada $r = [\, a \,]_i \rightarrow b \in \mathcal{R}_D$ de acuerdo al orden escogido **hacer**
    si $(i, \sigma) \notin B$ y r es aplicable a (i, σ) en $\mathcal{C}'_t$ **entonces**
      $\mathcal{C}'_t \leftarrow \mathcal{C}'_t \setminus n_r \cdot \{(a, (i, \sigma))\}$
      $A \leftarrow A \cup \{(r, 1, (i, \sigma))\}$
      $B \leftarrow B \cup \{(i, \sigma)\}$
    fin si
  fin para
  para cada $r \in \mathcal{R}_S$ de acuerdo al orden escogido **hacer**
    si $(i, \sigma) \notin B$ y r es aplicable a (i, σ) en $\mathcal{C}'_t$ **entonces**
      $\mathcal{C}'_t \leftarrow \mathcal{C}'_t \setminus n_r \cdot LHS(r, (i, \sigma))$
      $A \leftarrow A \cup \{(r, 1, (i, \sigma))\}$
      $B \leftarrow B \cup \{(i, \sigma)\}$
    fin si
  fin para
fin para

Algoritmo 7.2.3 Algoritmo de la fase de ejecución

Entrada: El par $(\mathcal{C}'_t,\ A)$ de la fase de selección
Salida: La configuración siguiente $\mathcal{C}_{t+1}$

para cada $(r, n_r, (i,\sigma)) \in A$ de acuerdo al orden establecido **hacer**
si $r \in \mathcal{R}_E$ **entonces**
$\mathcal{C}'_t \leftarrow \mathcal{C}'_t + n_r \cdot RHS(r,(i,\sigma))$
si no si $r \in \mathcal{R}_C$ **entonces**
$\mathcal{C}'_t \leftarrow \mathcal{C}'_t + RHS(r,(i,\sigma))$
si no si $r \in \mathcal{R}_D$ **entonces**
$\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(b,(i,\sigma))\}$
$\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x,i,\sigma)/p(i,\sigma)\}$
Actualizar la estructura de membranas de acuerdo a la disolución de la membrana (i,σ)
si no si $r \in \mathcal{R}_S$ de acuerdo al orden escogido **entonces**
$\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x,i,\sigma)/\sigma 0\}$
$\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x,i,\sigma)/\sigma 1\}$
para cada $(x',i,\sigma) \in \mathcal{C}'_t$ de acuerdo al orden lexicográfico **hacer**
si $x' \in \Gamma_0$ **entonces**
$\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x',i,\sigma)/\sigma 0\}$
si no
$\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x',i,\sigma)/\sigma 1\}$
fin si
fin para
si r es una regla de separación de membranas no elementales **entonces**
para cada $(j,\tau) \in \mathcal{C}'_t \mid p(j,\tau) = (i,\sigma)$ **hacer**
si $j \in H_0$ **entonces**
$p(j,\tau) = (i,\sigma 0)$
si no si $j \in H_1$ **entonces**
$p(j,\tau) = (i,\sigma 1)$
fin si
fin para
fin si
fin si
fin para
$\mathcal{C}_{t+1} \leftarrow \mathcal{C}'_t$

$\mathbb{N}\}$ una familia de dichos sistemas que resuelve X. Sea (cod, s) una codificación polinomial asociada a dicha solución. Recordemos que la instancia $u \in I_X$ del problema X es procesada por el sistema $\Pi(s(u)) + cod(u)$. Se considera entonces el Algoritmo 7.2.4.

Algoritmo 7.2.4 Algoritmo $\mathcal{A}'$

Entrada: Una instancia u del problema de decisión X

Salida: *Sí* si $\Pi(s(u)) + cod(u)$ tiene una computación de aceptación, *No* en cualquier otro caso

Construir el sistema $\Pi(s(u)) + cod(u)$

Ejecutar el algoritmo $\mathcal{A}$ con entrada $\Pi(s(u)) + cod(u)$

Dada una instancia u del problema de decisión $X = (I_X, \theta_X)$, las siguientes afirmaciones son equivalentes:

1. $\theta_X(u) = 1$; es decir, la respuesta de la instancia u del problema X es afirmativa.

2. Todas las computaciones de $\Pi(s(u)) + cod(u)$ son computaciones de aceptación.

3. La salida del algoritmo $\mathcal{A}'$ con entrada u es *Sí*.

Por lo tanto, el algoritmo $\mathcal{A}'$ provee una solución al problema de decisión X. Teniendo en cuenta que dicho algoritmo se ejecuta en tiempo polinomial, deducimos que $X \in \mathbf{P}$.

La inclusión contraria es trivial dado que la clase $\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)$ es no vacía y es cerrada bajo reducibilidad en tiempo polinomial, por tanto, $\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{SAM}^0(+e_s, mcmp_{in-out}, +d, +n)}$.

□

7.2.2. Nuevas fronteras obtenidas

En esta sección, se ha obtenido una nueva frontera de la tratabilidad. Por una parte, el marco $\mathcal{DAM}^0(+e_s, \beta, -d, +n), \beta \in \{mcmp_{in}, mcmp_{out}\}$ proporciona sistemas capaces de resolver problemas **NP**-completos de manera eficiente. Por el contrario, cuando se cambian las reglas de división por reglas de separación, incluso permitiendo *cooperación* en reglas de comunicación tanto hacia dentro como hacia fuera no se puede alcanzar la eficiencia. Por tanto,

pasar de usar reglas de separación a usar reglas de división en este marco de sistemas de membranas equivale a pasar de la no eficiencia a la presumible eficiencia.

En la Figura 7.3 se puede observar el panorama de las clases de complejidad incluidas en los sistemas estudiados en este capítulo.

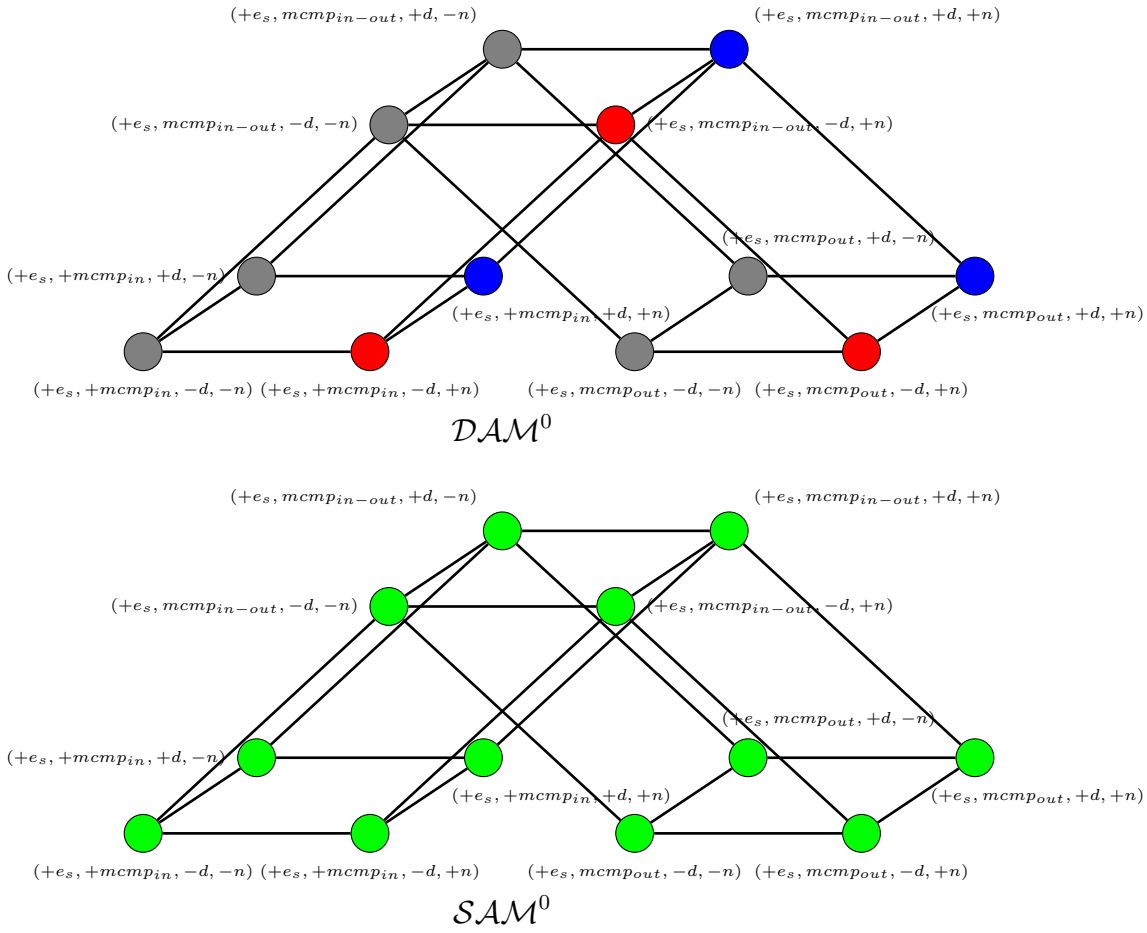


Figura 7.3: Eficiencia computacional en sistemas P con membranas activas sin polarizaciones.

Parte III

Comunicación evolutiva

Capítulo 8

Sistemas P a modo de tejidos con reglas de comunicación evolutiva

En la Sección 4.3 se ha presentado el marco de los sistemas P a modo de tejidos con reglas de comunicación evolutiva. La peculiaridad de estos sistemas con respecto a los sistemas P a modo de tejidos con reglas de comunicación del tipo symport/antiport clásicas radica en que en estos últimos, las reglas son del tipo $(i, u/v, j)$ y su aplicación hace que los objetos de los *multiconjuntos* u y v se intercambien de zona, pero no evolucionen en el proceso; sin embargo, en los primeros, las reglas tienen el formato $[\,u\,]_i\,[\,v\,]_j \rightarrow [\,v'\,]_i\,[\,u'\,]_j$ y su aplicación permite que además de cambiar de zona, los objetos de los multiconjuntos u y v puedan evolucionar en ese proceso proporcionando los multiconjuntos u' y v', respectivamente. Estas reglas pueden considerarse como una generalización de las reglas symport/antiport, ya que cualquier regla del tipo $(i, u/v, j)$ puede ser representada por una regla de comunicación evolutiva del tipo $[\,u\,]_i\,[\,v\,]_j \rightarrow [\,v\,]_i\,[\,u\,]_j$.

En relación con las reglas de comunicación evolutiva existen dos medidas de complejidad distintas, en este caso referidas al tamaño de las reglas, se tratarán ambas a lo largo del capítulo. La primera medida define el tamaño de una regla como el número total de objetos involucrados en la misma [60]. La segunda medida define el tamaño de una regla mediante un par ordenado n donde la primera componente (respectivamente, la segunda componente) es el número total de objetos involucrados en la parte izquierda de la regla (respectivamente, la parte derecha de la regla) [39]. En la Sección 5.2 se han presentado distintas

fronteras de la complejidad que ya han sido establecidas en este marco.

Cuando se usan reglas de división, en primer lugar, usando la medida definida en [60], la frontera está definida en el cambio del tamaño de las reglas de 2 a 4; es decir, mientras que con reglas symport/antiport evolucionales de tamaño 2 permiten resolver eficientemente sólo problemas de la clase **P**, se provee en el mismo artículo una solución al problema SAT en tiempo polinomial. En esta solución, además, se puede observar que las reglas tienen una longitud máxima de 2 en la parte izquierda y de 2 en la parte derecha. Como se indica en [39], los sistemas de membranas de $\mathcal{TDEC}(1,k), k \geq 1$ [1] son no eficientes debido, básicamente, a que, al ser sistemas no cooperativos, se puede usar la técnica del grafo de dependencia.

El uso de las reglas de separación cambia un poco el «paisaje» de las fronteras. Mientras que con la primera medida de complejidad se mantiene la misma frontera, con la segunda medida de complejidad se tienen nuevos resultados. En [39] se presenta una solución eficiente del problema SAT en $\mathcal{TSEC}(3,2)$. Mientras que la restricción de problemas resolubles eficientemente aumenta con respecto al uso de reglas de división a que una sola de las partes tenga tamaño 1. Esta parte del trabajo se va a dedicar a algunos de los problemas abiertos existentes que aparecen en círculos grises en las Figuras 5.6, 5.7, 5.8 y 5.9).

8.1. Sistemas P a modo de tejidos con reglas de comunicación evolutiva y división

En [60] se da una solución al problema SAT mediante una familia de sistemas de $\mathcal{TDEC}(2,2)$. Este resultado es, en cierta manera, trivial dado el resultado $\mathbf{PMC}_{\mathcal{TDC}(k)} \subseteq \mathbf{PMC}_{\mathcal{TDEC}(k,k)}$, dado que las reglas symport/antiport evolutivas son una generalización de las reglas symport/antiport clásicas. En esta sección se presentará una mejora de este resultado reduciendo la longitud de la parte derecha de las reglas de comunicación a 1.

8.1.1. Solución al problema SAT en $\mathcal{TDEC}(2,1)$

Para cada $n, p \in \mathbb{N}$, consideramos el sistema P
$\Pi(\langle n,p\rangle) = (\Gamma, \Sigma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$ de $\mathcal{TDEC}(2,1)$ definido como sigue:

[1] En la referencia se refiere a los sistemas que usan separación celular en lugar de división celular, pero el argumento es válido para ambos tipos de reglas.

1. Alfabeto de trabajo Γ:
   $\{\texttt{yes}, \texttt{no}, y_1, y_2, n_1, n_2, \#\} \cup$
   $\{a_{i,j}, T_{i,j}, F_{i,j} \mid 1 \leq i \leq n, 0 \leq j \leq p\} \cup$
   $\{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 1 \leq i \leq n, 1 \leq j \leq p, 0 \leq k \leq 2np\} \cup$
   $\{\alpha_j \mid 1 \leq j \leq p+1\} \cup$
   $\{\delta_k \mid 0 \leq k \leq 2np+2\} \cup$
   $\{\delta'_k \mid 0 \leq k \leq 2np\}$.

2. Alfabeto de entrada Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,k,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.

3. Alfabeto del entorno $\mathcal{E}$: $\{\gamma\}$.

4. $\mathcal{M}_k = \emptyset, 1 \leq k \leq np$,
   $\mathcal{M}_{np+1} = \{\delta_0\}$,
   $\mathcal{M}_{np+2} = \{a_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq p\} \cup \{\alpha_j \mid 1 \leq j \leq p+1\}$,
   $\mathcal{M}_{np+3} = \{\delta'_0\}$

5. El conjunto de reglas $\mathcal{R}$ consiste en las siguientes reglas:

   1.1 Reglas para generar p copias de las 2^n posibles valoraciones de verdad. Para ello, se generarán 2^{np} valoraciones de verdad parciales.

   $$\left.\begin{array}{l} [\, a_{i,j}\,]_{np+2} \to [\, T_{i,j}\,]_{np+2}\,[\, F_{i,j}\,]_{np+2} \\ [\, T_{i,j}F_{i,j'}\,]_{np+2}\,[\quad]_0 \to [\quad]_{np+2}\,[\,\#\,]_0 \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j, j' \leq p \end{array}$$

   1.2 Reglas para generar 2^{np} copias de $cod(\varphi)$.

   $$\left.\begin{array}{l} [\, x_{i,j,0}\,]_{np+1}\,[\quad]_{i\perp j} \to [\quad]_{np+1}\,[\, x_{i,j,1}\,]_{i\perp j} \\ [\, \overline{x}_{i,j,0}\,]_{np+1}\,[\quad]_{i\perp j} \to [\quad]_{np+1}\,[\, \overline{x}_{i,j,1}\,]_{i\perp j} \\ [\, x^*_{i,j,0}\,]_{np+1}\,[\quad]_{i\perp j} \to [\quad]_{np+1}\,[\, x^*_{i,j,1}\,]_{i\perp j} \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$$

   $$\left.\begin{array}{l} [\, x_{i,j,k}\,]_{i\perp j}\,[\,\gamma\,]_0 \to [\, x_{i,j,k+1}\,]_{i\perp j}\,[\quad]_0 \\ [\, \overline{x}_{i,j,k}\,]_{i\perp j}\,[\,\gamma\,]_0 \to [\, \overline{x}_{i,j,k+1}\,]_{i\perp j}\,[\quad]_0 \\ [\, x^*_{i,j,k}\,]_{i\perp j}\,[\,\gamma\,]_0 \to [\, x^*_{i,j,k+1}\,]_{i\perp j}\,[\quad]_0 \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 1 \leq k \leq np \end{array}$$

   $$\left.\begin{array}{l} [\, x_{i,j,k}\,]_{i\perp j} \to [\, x_{i,j,k+1}\,]_{i\perp j}\,[\, x_{i,j,k+1}\,]_{i\perp j} \\ [\, \overline{x}_{i,j,k}\,]_{i\perp j} \to [\, \overline{x}_{i,j,k+1}\,]_{i\perp j}\,[\, \overline{x}_{i,j,k+1}\,]_{i\perp j} \\ [\, x^*_{i,j,k}\,]_{i\perp j} \to [\, x^*_{i,j,k+1}\,]_{i\perp j}\,[\, x^*_{i,j,k+1}\,]_{i\perp j} \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ np+1 \leq k \leq 2np \end{array}$$

   $$[\, \delta'_k\,]_{np+3}\,[\,\gamma\,]_0 \to [\, \delta'_{k+1}\,]_{np+3}\,[\quad]_0 \quad \text{, para } 0 \leq k \leq 2np$$

   $$[\, \delta'_k\,]_{np+3} \to [\, \delta'_{k+1}\,]_{np+3}\,[\, \delta'_{k+1}\,]_{np+3} \quad \text{, para } np+1 \leq k \leq 2np$$

   2.1 Reglas para chequear las cláusulas satisfechas.

$$\left.\begin{array}{l}
[\,T_{i,j}\,]_{np+2}\,[\,x_{i,j,2np+1}\,]_{i\perp j} \to [\,c_j\,]_{np+2}\,[\quad]_{i\perp j}\\
[\,T_{i,j}\,]_{np+2}\,[\,\overline{x}_{i,j,2np+1}\,]_{i\perp j} \to [\,\#\,]_{np+2}\,[\quad]_{i\perp j}\\
[\,T_{i,j}\,]_{np+2}\,[\,x^*_{i,j,2np+1}\,]_{i\perp j} \to [\,\#\,]_{np+2}\,[\quad]_{i\perp j}\\
[\,F_{i,j}\,]_{np+2}\,[\,x_{i,j,2np+1}\,]_{i\perp j} \to [\,\#\,]_{np+2}\,[\quad]_{i\perp j}\\
[\,F_{i,j}\,]_{np+2}\,[\,\overline{x}_{i,j,2np+1}\,]_{i\perp j} \to [\,c_j\,]_{np+2}\,[\quad]_{i\perp j}\\
[\,F_{i,j}\,]_{np+2}\,[\,x^*_{i,j,2np+1}\,]_{i\perp j} \to [\,\#\,]_{np+2}\,[\quad]_{i\perp j}
\end{array}\right\} \text{para } \begin{array}{l} 1 \le i \le n,\\ 1 \le j \le p \end{array}$$

3.1 Reglas para chequear si todas las cláusulas son satisfechas por una valoración de verdad.

$$[\,\alpha_{p+1}\,]_{np+2}\,[\,\delta'_{2np+1}\,]_{np+3} \to [\,\alpha'_{p+1}\,]_{np+2}\,[\quad]_{np+3}$$
$$[\,c_j\alpha_j\,]_{np+2}\,[\quad]_0 \to [\quad]_{np+2}\,[\,\#\,]_0 \quad \text{, para } 1 \le j \le p$$

4.1 Contador general.

$$[\,\delta_k\,]_{np+1}\,[\,\gamma\,]_0 \to [\,\delta_{k+1}\,]_{np+1}\,[\quad]_0 \quad \text{, para } 0 \le k \le 2np+2$$

4.2 Reglas para devolver una respuesta negativa.

$$[\,\alpha_j\alpha'_{p+1}\,]_{np+2}\,[\quad]_0 \to [\quad]_{np+2}\,[\,n_1\,]_0 \quad \text{, para } 1 \le j \le p$$
$$[\,n_1\,]_0\,[\quad]_{np+2} \to [\quad]_0\,[\,n_1\,]_{np+2}$$
$$[\,n_1\,]_{np+1}\,[\,\delta_{2np+3}\,]_{np+1} \to [\,n_2\,]_{np+1}\,[\quad]_{np+1}$$
$$[\,n_2\,]_{np+2}\,[\quad]_0 \to [\quad]_{np+2}\,[\,\texttt{no}\,]_0$$

4.3 Reglas para devolver una respuesta positiva.

$$[\,\alpha'_{p+1}\,]_{np+2}\,[\,\delta_{2np+3}\,]_{np+1} \to [\,y_1\,]_{np+2}\,[\quad]_{np+1}$$
$$[\,y_1\,]_{np+2}\,[\,\gamma\,]_0 \to [\,y_2\,]_{np+2}\,[\quad]_0$$
$$[\,y_2\,]_{np+2}\,[\quad]_0 \to [\quad]_{np+2}\,[\,\texttt{yes}\,]_0$$

6. La célula de entrada es la célula etiquetada por $np+1$ ($i_{in} = np+1$) y la región de salida es el entorno ($i_{out} = env$).

8.1.2. Descripción de la computación

Consideramos la codificación de la fórmula φ como se sigue. Sea $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_p, C_j = l_{j_1} \vee \cdots \vee l_{j_{r_j}}, 1 \le j \le p, l_{j_k} \in \{x_i, \neg x_i \mid 1 \le i \le n\}$ una instancia del problema `SAT`. Entonces la fórmula Booleana φ será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$, en donde $s(\varphi) = \langle n, p\rangle$ y $cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$.

La solución propuesta sigue un algoritmo de fuerza bruta en el marco de los sistemas P reconocedores a modo de tejidos con reglas de comunicación evolucional con tamaño a lo sumo $(2,1)$ y reglas de división, y consiste en las siguientes etapas:

1. *Fase de generación*: Usando las reglas de 1.1, se generan p copias de cada una de las 2^n posibles valoraciones de verdad asociadas a $\{x_1, \ldots, x_n\}$ en las células etiquetadas por $np+2$. Para ello, $2^{np} - 2^n$ valoraciones de verdad «parciales» son producidas por la eliminación de los elementos que tengan dos valores Booleanos asociados a una misma variable.

   En paralelo, 2^{np} copias de cada objeto $x_{i,j,2np+1}, \overline{x}_{i,j,2np+1}, x^*_{i,j,2np+1}$ que aparezca en $cod(\varphi)$ son generadas por la aplicación de las reglas de 1.1. Para sincronizarlo con el comienzo de la siguiente fase, el tercer índice, k, «evolucionará» hasta llegar a $2np+1$. Esta fase tomará exactamente $2np+1$ pasos de computación.

2. *Primera fase de chequeo*: Las p copias de cada 2^n valoraciones de verdad asociadas con $\{x_1, \ldots, x_n\}$ generadas en las células etiquetadas por $np+2$ cooperan con las 2^n copias de los objetos $x_{i,j,2np+1}, \overline{x}_{i,j,2np+1}, x^*_{i,j,2np+1}$ usando las reglas de 2.1, para saber las cláusulas que son satisfechas por la correspondiente valoración de verdad. Merece la pena destacar que mediante esta fase de chequeo, las $2^{np} - 2^n$ valoraciones de verdad parciales no pueden satisfacer todas las cláusulas $C_1, \ldots, C_p$ en caso que φ sea insatisfactible. Esta fase dura 1 paso de computación.

3. *Segunda fase de chequeo*: Mediante las reglas de 3.1, el objeto α_j es borrado de la célula etiquetada por $np+2$ si y sólo si la valoración de verdad asociada a esa célula hace verdadera la cláusula C_j. Por tanto, una célula etiquetada por $np+2$ codificará una valoración de verdad que hace verdadera la fórmula φ si y sólo si no queda ningún objeto α_j al terminar esta fase, que dura exactamente 1 paso de computación.

4. *Fase de salida*: Para terminar, las reglas de 4.2 y 4.3 pueden devolver una respuesta afirmativa o negativa dependiendo de si la fórmula φ es satisfactible o no. Esta fase dura exactamente 4 pasos de computación, sea verdadera o falsa la respuesta.

Teorema 8.1. $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{TDEC}(2,1)}$

Demostración. La familia de sistemas P construida previamente verifica lo siguiente:

- Todo sistema de la familia $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ es un sistema P reconocedor de $\mathcal{TDEC}(2,1)$.

- La familia $\mathbf{\Pi}$ es polinomialmente uniforme por máquinas de Turing dado que para cada $n, p \in \mathbb{N}$, las reglas de $\Pi(\langle n, p\rangle)$ de la familia son recursivamente definidas por $n, p \in \mathbb{N}$, y la cantidad de recursos necesitados para construir un elemento de la familia es de orden polinomial con respecto a n y a p, como se muestra debajo:

  - Tamaño del alfabeto: $\Theta(n^2p^2)$.
  - Número inicial de células: $3 + np \in \Theta(np)$.
  - Número inicial de objetos en células: $np + p + 3 \in \Theta(np)$.
  - Número de reglas: $\Theta(n^2p^2)$.
  - Número máximo de objetos involucrados en cualquier regla: $3 \in \Theta(1)$.

- El par de funciones computables en tiempo polinomial (cod, s) definido cumple lo siguiente: para cada fórmula φ del problema `SAT`, $s(\varphi)$ es un número natural, $cod(\varphi)$ es un multiconjunto de entrada del sistema $\Pi(s(\varphi))$ y para cada $t \in \mathbb{N}$, $s^{-1}(t)$ es un conjunto finito.

- La familia $\mathbf{\Pi}$ está polinomialmente acotada en tiempo: de hecho, para cada fórmula φ del problema `SAT`, el sistema P reconocedor $\Pi(s(\varphi)) + cod(\varphi)$ tarda exactamente $\lfloor \frac{3np}{2} \rfloor + np + 6$ pasos de computación en devolver una respuesta positiva o negativa, siendo n el número de variables y p el número de cláusulas de φ.

- La familia $\mathbf{\Pi}$ es adecuada con respecto a (X, cod, s): de hecho, para cada fórmula φ, si las computaciones de $\Pi(s(\varphi)) + cod(\varphi)$ son de *aceptación*, entonces φ es satisfactible.

- La familia $\mathbf{\Pi}$ es completa con respecto a (X, cod, s): de hecho, para cada fórmula φ que sea satisfactible, todas las computaciones de $\Pi(s(\varphi)) + cod(\varphi)$ son de *aceptación*.

□

Corolario 8.1. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{TDEC}(2,1)}$

Demostración. Basta con observar que `SAT` es un problema **NP**-completo, `SAT` $\in \mathbf{PMC}_{\mathcal{TDEC}(2,1)}$ y la clase de complejidad $\mathbf{PMC}_{\mathcal{TDEC}(2,1)}$ está cerrada bajo reducibilidad en tiempo polinomial y bajo complementario. □

Dado que el tamaño máximo de las reglas con la métrica definida en [39] es $(2,1)$, podemos deducir de esto, según la métrica definida en [60], que a lo sumo se tienen que usar 3 objetos en el tamaño total de las reglas de comunicación evolutiva para poder resolver problemas presuntamente intratables; es decir:

Corolario 8.2. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{TDEC}(3)}$

8.2. Sistemas P a modo de tejidos con reglas de comunicación evolutiva y separación

El mejor resultado para la resolución de un problema presuntamente intratable en el marco del que trata la actual sección se presentó en [39], usando sistemas de $\mathcal{TSEC}(3,2)$. En dicha solución, puede atisbarse que las reglas symport/antiport que tienen longitud 3 pueden evitarse y reducir dicha longitud a 2. Aquí se presentará una solución al problema `SAT` con dicha restricción. A continuación, se presentará una solución al problema `SAT` mediante una familia de sistemas de $\mathcal{TSEC}(2,2)$.

8.2.1. Solución al problema `SAT` en $\mathcal{TSEC}(2,2)$

Para cada $n,p \in \mathbb{N}$, consideramos el sistema P $\Pi(\langle n,p\rangle) = (\Gamma, \Sigma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$ de $\mathcal{TSEC}(2,2)$ definido como sigue:

1. Alfabeto de trabajo Γ:
   $\{\texttt{yes}, \texttt{no}, y_1, y_2, n_1, n_2, \#\} \cup$
   $\{a_{i,j} \mid 1 \leq i \leq n, 0 \leq j \leq i\} \cup$
   $\{a'_{i,j} \mid 2 \leq i \leq n, 0 \leq j \leq i-1\} \cup$
   $\{a^L_{i,j}, a^R_{i,j} \mid 2 \leq i \leq n, 1 \leq j \leq i-1\} \cup$
   $\{\alpha_j, \alpha'_j, \alpha^L_j, \alpha^R_j \mid 1 \leq j \leq p+1\} \cup$
   $\{t_i, f_i, t'_i, t''_i f''_i, t^L_i, t^R_i, f^L_i, f^R_i \mid 1 \leq i \leq n\} \cup$
   $\{\beta_{l,k}, \beta'_{l,k}, \beta^L_{l,k}, \beta^R_{l,k} \mid 0 \leq k \leq n, 1 \leq l \leq n\} \cup$
   $\{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n+j-1\} \cup$
   $\{x'_{i,j,k}, \overline{x}'_{i,j,k}, {x^*}'_{i,j,k}, x''_{i,j,k}, \overline{x}''_{i,j,k}, {x^*}''_{i,j,k}, x'''_{i,j,k}, \overline{x}'''_{i,j,k}, {x^*}'''_{i,j,k}, |$
   $0 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n\} \cup$
   $\{c_{j,k} \mid 1 \leq j \leq p, j \leq k \leq p\} \cup \{\delta_i \mid 0 \leq i \leq 4n+p+2\} \cup$
   $\{\delta'_i \mid 0 \leq i \leq 4n+p\}$.

2. La partición $\{\Gamma_0, \Gamma_1\}$ está definida como sigue:
   $\Gamma_1 = \Gamma \setminus \Gamma_0, \Gamma_0 = \{a^L_{i,j} \mid 2 \leq i \leq n, 1 \leq j \leq i-1\} \cup$

$\{\alpha_j^L \mid 1 \leq j \leq p+1\} \cup \{t_i^L, f_i^L \mid 1 \leq i \leq n\} \cup$
$\{\beta_{l,k}^L \mid 0 \leq k \leq n, k+1 \leq l \leq n\}$.

3. Alfabeto de entrada Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,k,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.

4. Alfabeto del entorno $\mathcal{E}$: $\{\gamma\}$.

5. $\mathcal{M}_1 = \{\delta_0, \delta_0'\} \cup \{\beta_{l,0}^{n+p+1} \mid 1 \leq l \leq n\}$,
$\mathcal{M}_2 = \{a_{i,0} \mid 1 \leq i \leq n\} \cup \{\alpha_j \mid 1 \leq j \leq p+1\}$.

6. El conjunto $\mathcal{R}$ está formado por las siguientes reglas:

1.1 Reglas para los pasos $4k+1$.

$$[\,a_{i,i-1}\,]_2\,[\,\gamma\,]_0 \to [\,a'_{i,i-1}t'_i\,]_2\,[\quad]_0 \quad, \text{para } 1 \leq i \leq n$$

$$\left.\begin{array}{l}[\,t_i\,]_2\,[\,\gamma\,]_0 \to [\,t''_i\,]_2\,[\quad]_0 \\ [\,f_i\,]_2\,[\,\gamma\,]_0 \to [\,f''_i\,]_2\,[\quad]_0\end{array}\right\} \text{para } 1 \leq i \leq n$$

$$[\,a_{i,j}\,]_2\,[\,\gamma\,]_0 \to [\,a'_{i,j}\,]_2\,[\quad]_0 \quad, \text{para } 2 \leq i \leq n, 0 \leq j \leq i-2$$

$$[\,\alpha_j\,]_2\,[\,\gamma\,]_0 \to [\,\alpha'_j\,]_2\,[\quad]_0 \quad, \text{ para } 1 \leq j \leq p+1$$

$$\left.[\,\beta_{l,k}\,]_1\,[\,\gamma\,]_0 \to [\,\beta'_{l,k}\,]_1\,[\quad]_0\right\} \text{para } \begin{array}{l}0 \leq k \leq n, \\ k+1 \leq l \leq n\end{array}$$

$$\left.\begin{array}{l}[\,x_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,x'_{i,j,k}\,]_1\,[\quad]_0 \\ [\,\overline{x}_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}'_{i,j,k}\,]_1\,[\quad]_0 \\ [\,x^*_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,x^{*\prime}{}_{i,j,k}\,]_1\,[\quad]_0\end{array}\right\} \text{para } \begin{array}{r}1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 0 \leq k \leq n-1\end{array}$$

1.2 Reglas para los pasos $4k+2$.

$$\left.\begin{array}{l}[\,a'_{i,i-1}\,]_2\,[\,\gamma\,]_0 \to [\,a_{i,i}f_i^R\,]_2\,[\quad]_0 \\ [\,t'_i\,]_2\,[\,\gamma\,]_0 \to [\,t_i^L\,]_2\,[\quad]_0\end{array}\right\} \text{para } 1 \leq i \leq n$$

$$\left.\begin{array}{l}[\,t''_i\,]_2\,[\,\gamma\,]_0 \to [\,t_i^L\,t_i^R\,]_2\,[\quad]_0 \\ [\,f''_i\,]_2\,[\,\gamma\,]_0 \to [\,f_i^L\,f_i^R\,]_2\,[\quad]_0\end{array}\right\} \text{para } 1 \leq i \leq n$$

$$[\,a'_{i,j}\,]_2\,[\,\gamma\,]_0 \to [\,a_{i,j+1}^L\,a_{i,j+1}^R\,]_2\,[\quad]_0 \quad, \text{para } \begin{array}{l}2 \leq i \leq n, \\ 0 \leq j \leq i-1\end{array}$$

$$[\,\alpha'_j\,]_2\,[\,\gamma\,]_0 \to [\,\alpha_j^L\,\alpha_j^R\,]_2\,[\quad]_0 \quad, \text{ para } 1 \leq j \leq p+1$$

$$[\,\beta'_{l,k}\,]_1\,[\,\gamma\,]_0 \to [\,\beta_{l,k+1}^L\,\beta_{l,k+1}^R\,]_1\,[\quad]_0 \quad, \text{ para } \begin{array}{l}0 \leq k \leq n, \\ k+1 \leq l \leq n\end{array}$$

$$\left.\begin{array}{l}[\,x'_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,x''^2_{i,j,k+1}\,]_1\,[\quad]_0 \\ [\,\overline{x}'_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}''^2_{i,j,k+1}\,]_1\,[\quad]_0 \\ [\,x^{*\prime}{}_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,x^{*\prime\prime 2}{}_{i,j,k+1}\,]_1\,[\quad]_0\end{array}\right\} \begin{array}{l}1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 0 \leq k \leq n-1\end{array}$$

1.3 Reglas para los pasos $4k+3$.

$$[\,a_{i,i}\,]_2 \to [\,\Gamma_0\,]_2\,[\,\Gamma_1\,]_2 \quad \text{, para } 1 \le i \le n$$

$$\left.\begin{array}{l}[\,\beta^O_{k,k}\,]_1\,[\quad]_0 \to [\quad]_1\,[\,\beta^O_{k,k}\,]_0\\ [\,\beta^O_{l,k}\,]_1\,[\quad]_0 \to [\quad]_1\,[\,\beta_{l,k}\,]_0\end{array}\right\} \text{para } \begin{array}{r} O \in \{L,R\},\\ 1 \le k \le n,\\ k+1 \le l \le n\end{array}$$

$$\left.\begin{array}{l}[\,x''_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,x'''_{i,j,k}\,]_1\,[\quad]_0\\ [\,\overline{x}''_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}'''_{i,j,k}\,]_1\,[\quad]_0\\ [\,x^{*\prime\prime}_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,x^{*\prime\prime\prime}_{i,j,k}\,]_1\,[\quad]_0\end{array}\right\} \text{para } \begin{array}{r} 1 \le i \le n,\\ 1 \le j \le p,\\ 1 \le k \le n\end{array}$$

1.4 Reglas para los pasos $4k$.

$$\left.\begin{array}{l}[\,a^O_{i,j}\,]_2\,[\,\beta^O_{k,k}\,]_0 \to [\,a_{i,j}\,]_2\,[\quad]_0\\ [\,r^O_i\,]_2\,[\,\beta^O_{k,k}\,]_0 \to [\,r_i\,]_2\,[\quad]_0\end{array}\right\} \text{para } \begin{array}{r} O \in \{L,R\},\\ r \in \{t,f\},\\ 1 \le i \le n,\\ 1 \le j \le n,\\ 1 \le k \le n\end{array}$$

$$[\,\alpha^O_j\,]_2\,[\,\beta^O_{k,k}\,]_0 \to [\,\alpha_j\,]_2\,[\quad]_0 \quad \text{, para } \begin{array}{r} O \in \{L,R\},\\ 1 \le j \le p+1,\\ 0 \le k \le n\end{array}$$

$$\left.\begin{array}{l}[\,x'''_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,x_{i,j,k}\,]_1\,[\quad]_0\\ [\,\overline{x}'''_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}_{i,j,k}\,]_1\,[\quad]_0\\ [\,x^{*\prime\prime\prime}_{i,j,k}\,]_1\,[\,\gamma\,]_0 \to [\,x^*_{i,j,k}\,]_1\,[\quad]_0\end{array}\right\} \text{para } \begin{array}{r} 1 \le i \le n,\\ 1 \le j \le p,\\ 0 \le k \le n\end{array}$$

$$[\,\beta_{l,k}\,]_0\,[\quad]_1 \to [\quad]_0\,[\,\beta_{l,k}\,]_1 \quad \text{, para } 0 \le k \le n, k+1 \le l \le n$$

2.1 Reglas para chequear las cláusulas satisfechas.

$$\left.\begin{array}{l}[\,t_i\,]_2\,[\,x_{i,j,n+j-1}\,]_1 \to [\,c_{j,j}\,t_i\,]_2\,[\quad]_1\\ [\,t_i\,]_2\,[\,\overline{x}_{i,j,n+j-1}\,]_1 \to [\,t_i\,]_2\,[\quad]_1\\ [\,t_i\,]_2\,[\,x^*_{i,j,n+j-1}\,]_1 \to [\,t_i\,]_2\,[\quad]_1\\ [\,f_i\,]_2\,[\,x_{i,j,n+j-1}\,]_1 \to [\,f_i\,]_2\,[\quad]_1\\ [\,f_i\,]_2\,[\,\overline{x}_{i,j,n+j-1}\,]_1 \to [\,c_{j,j}\,f_i\,]_2\,[\quad]_1\\ [\,f_i\,]_2\,[\,x^*_{i,j,n+j-1}\,]_1 \to [\,f_i\,]_2\,[\quad]_1\end{array}\right\} \text{para } 1 \le i \le n, 1 \le j \le p$$

$$\left.\begin{array}{l}[\,x_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,x_{i,j,n+k+1}\,]_1\,[\quad]_0\\ [\,\overline{x}_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}_{i,j,n+k+1}\,]_1\,[\quad]_0\\ [\,x^*_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,x^*_{i,j,n+k+1}\,]_1\,[\quad]_0\end{array}\right\} \text{para } \begin{array}{r} 1 \le i \le n,\\ 1 \le j \le p,\\ 0 \le k \le j-2\end{array}$$

$$[\,c_{j,k}\,]_2\,[\,\gamma\,]_0 \to [\,c_{j,k+1}\,]_2\,[\quad]_0 \quad \text{, para } 1 \le j \le p, j \le k \le p-1$$

3.1 Reglas para chequear si todas las cláusulas son satisfechas por una valoración de verdad.

$$[\,\alpha_{p+1}\,]_2\,[\,\delta'_{4n+p}\,]_1 \to [\,\alpha'_{p+1}\,]_2\,[\quad]_1$$

$$[\,\alpha_j\,c_{j,p}\,]_2\,[\quad]_0 \to [\quad]_2\,[\,\#\,]_0 \quad \text{, para } 1 \le j \le p$$

4.1 Contadores generales.

$$[\,\delta_i\,]_1\,[\,\gamma\,]_0 \to [\,\delta_{i+1}\,]_1\,[\quad]_0 \quad \text{, para } 0 \leq i \leq 4n+p+1$$
$$[\,\delta'_{4i+1}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'^2_{4i+2}\,]_1\,[\quad]_0 \quad \text{, para } 0 \leq i \leq n-1$$
$$[\,\delta'_{4i+k}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'_{4i+k+1}\,]_1\,[\quad]_0 \quad \text{, para } 0 \leq i \leq n-1, k \in \{0,2,3\}$$
$$[\,\delta'_{4n+i}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'_{4n+i+1}\,]_1\,[\quad]_0 \quad \text{, para } 0 \leq i \leq p-1$$

4.2 Reglas para devolver una respuesta negativa.

$$[\,\alpha_j\,\alpha'_{p+1}\,]_2\,[\quad]_0 \to [\quad]_2\,[\,n_1\,]_0 \quad \text{, para } 1 \leq j \leq p$$
$$[\,n_1\,]_0\,[\quad]_2 \to [\quad]_0\,[\,n_1\,]_2$$
$$[\,n_1\,]_2\,[\,\delta_{4n+p+2}\,]_1 \to [\,n_2\,]_2\,[\quad]_1$$
$$[\,n_2\,]_2\,[\quad]_0 \to [\quad]_2\,[\,\texttt{no}\,]_0$$

4.3 Reglas para devolver una respuesta positiva.

$$[\,\alpha'_{p+1}\,]_2\,[\,\delta_{4n+p+2}\,]_1 \to [\,y_1\,]_2\,[\quad]_1$$
$$[\,y_1\,]_2\,[\,\gamma\,]_0 \to [\,y_2\,]_2\,[\quad]_0$$
$$[\,y_2\,]_2\,[\quad]_0 \to [\quad]_2\,[\,\texttt{yes}\,]_0$$

7. La célula de entrada es la célula etiquetada por 1 ($i_{in} = 1$) y la región de salida es el entorno ($i_{out} = env$).

Consideramos la codificación de la fórmula φ como se sigue. Sea $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_p, C_j = l_{j_1} \vee \cdots \vee l_{j_{r_j}}, 1 \leq j \leq p, l_{j_k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$ una instancia del problema SAT. Entonces la fórmula Booleana φ será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$, en donde $s(\varphi) = \langle n, p\rangle$ y $cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$.

Definimos $cod_k(\varphi)$ como los elementos de $cod(\varphi)$ a los cuales su tercer subíndice se le ha asignado el valor k. De la misma manera, definimos $cod'_k(\varphi)$, $cod''_k(\varphi)$ y $cod'''_k(\varphi)$ como los elementos de $cod_k(\varphi)$ que se les añade una prima, doble prima y triple prima, respectivamente. Por conveniencia de notación, denotamos por $cod^j_k(\varphi)$ el subconjunto de elementos de $cod_k(\varphi)$ cuyo segundo subíndice sea mayor o igual que j.

La fórmula φ será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$. La solución propuesta sigue, como en la sección anterior, un algoritmo de fuerza bruta en el marco de los sistemas P reconocedores a modo de tejidos con reglas de comunicación evolucional con tamaño, a lo sumo, $(2, 2)$ y, en esta ocasión, reglas de separación. Consiste en las siguientes fases:

- *Fase de generación*: Usando reglas de separación cada 4 pasos, se producen 2^n células etiquetadas por 2 que tienen todas las posibles valoraciones

de verdad. Al mismo tiempo, generamos 2^n copias de $cod_n(\varphi)$. Esta fase tarda exactamente n pasos de computación, siendo n el número de variables de φ.

- *Primera fase de chequeo*: Utilizando las reglas de 2.1, podemos chequear qué cláusulas de la fórmula φ son satisfechas por una valoración de verdad en concreto. Esta fase tarda p pasos de computación, siendo p el número de cláusulas de φ.

- *Segunda fase de chequeo*: Mediante el uso de las reglas de 3.1, se eliminan los objetos α_j tales que son eliminados de una célula si y sólo si la valoración de verdad asociada a dicha célula hace verdadera la cláusula C_j. Esta fase tarda 1 paso de computación.

- *Fase de salida*: Con las reglas de 4.2 y 4.3, devolvemos una respuesta afirmativa o negativa dependiendo de si la fórmula φ es satisfactible o no. Esta fase tarda exactamente 4 pasos de computación, independientemente de si φ es satisfactible o no.

8.2.1.1. Verificación formal

Aquí se detallará una verificación exhaustiva del sistema.

Fase de generación

En esta fase, todas las valoraciones de verdad de las variables asociadas a la fórmula $\varphi(x_1, \ldots, x_n)$ serán generadas, mediante la aplicación de reglas de separación de 1.2 en las células etiquetadas por 2. De tal manera que en los pasos $4i + 2$ $(1 \leq i \leq n - 1)$ de esta fase, la regla de separación asociada al objeto $a_{i,i}$ será disparada, y los objetos t_i y f_i serán distribuidos entre las dos nuevas células creadas. En el último paso de esta fase, cada célula etiquetada por 2 contendrá una valoración de verdad de la fórmula.

Proposición 8.1. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a_0) *Para cada* $4k$ $(0 \leq k \leq n-1)$ *en la configuración* $\mathcal{C}_{4k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{4k}(1) = \{\delta_{4k}, {\delta'}_{4k}^{2^k}, cod_k(\varphi)^{2^k}\} \cup \{\beta_{l,k}^{2^k} \mid k + 1 \leq l \leq n\}$
- *Hay* 2^k *células etiquetadas por 2 tales que contienen*

- *objetos* $a_{k+1,k}, \ldots, a_{n,k}$;
- *objetos* $r_1, \ldots, r_k$, *siendo* $r \in \{t, f\}$; *y*
- *objetos* $\alpha_1, \ldots, \alpha_{p+1}$.

(a_1) *Para cada* $4k+1$ *(*$0 \leq k \leq n-1$*) en la configuración* $Config_{4k+1}$ *tenemos lo siguiente:*

- $\mathcal{C}_{4k+1}(1) = \{\delta_{4k+1}, {\delta'}^{2^k}_{4k+1}, cod'_k(\varphi)^{2^k}\} \cup \{{\beta'}^{2^k}_{l,k} \mid k+1 \leq l \leq n\}$
- *Hay* 2^k *células etiquetadas por 2 tales que contienen*
  - *objetos* $a'_{k+1,k}, \ldots, a'_{n,k}$;
  - *objetos* $r''_1, \ldots, r''_k$, *siendo* $r \in \{t, f\}$
  - *un objeto* t'_{k+1}; *y*
  - *objetos* $\alpha'_1, \ldots, \alpha'_{p+1}$.

(a_2) *Para cada* $4k+2$ *(*$0 \leq k \leq n-1$*) en la configuración* $\mathcal{C}_{4k+2}$ *tenemos lo siguiente:*

- $\mathcal{C}_{4k+2}(1) = \{\delta_{4k+2}, {\delta'}^{2^{k+1}}_{4k+3}, cod''_{k+1}(\varphi)^{2^{k+1}}\} \cup$
  $\{{\beta^O}^{2^k}_{l,k} \mid O \in \{L, R\}, k+1 \leq l \leq n\}$
- *Hay* 2^k *células etiquetadas por 2 tales que contienen*
  - *objetos* $a_{k+1,k}, \ldots, a_{n,k}$;
  - *objetos* $r_1, \ldots, r_k$, *siendo* $r \in \{t, f\}$; *y*
  - *objetos* $\alpha_1, \ldots, \alpha_{p+1}$.

(a_3) *Para cada* $4k+3$ *(*$0 \leq k \leq n-1$*) en la configuración* $\mathcal{C}_{4k+3}$ *tenemos lo siguiente:*

- $\mathcal{C}_{4k+3}(0) = \{{\beta^O}^{2^k}_{k+1,k+1}\} \cup \{\beta^{2^{k+1}}_{l,k+1} \mid k+2 \leq l \leq n\}$
- $\mathcal{C}_{4k+3}(1) = \{\delta_{4k+3}, {\delta'}^{2^{k+1}}_{4k+3}, cod'''_{k+1}(\varphi)^{2^{k+1}}\} \cup \{\beta^{2^k}_{l,k} \mid k+1 \leq l \leq n\}$
- *Hay* 2^{k+1} *células etiquetadas por 2 tales que contienen*
  - *objetos* $a_{k+1,k}, \ldots, a_{n,k}$;
  - *objetos* $r_1, \ldots, r_k$, *siendo* $r \in \{t, f\}$; *y*
  - *objetos* $\alpha_1, \ldots, \alpha_{p+1}$.

(*b*) $\mathcal{C}_{4n}(1) = \{\delta_{4n}, {\delta'}^{2^n}_{4n}, cod_{4n}(\varphi)^{2^n}\}$, *y hay* 2^n *células etiquetadas por 2 tales que cada una de ellas contiene los objetos* $\alpha_1, \ldots, \alpha_{p+1}$, *así como un subconjunto distinto* $\{r_1, \ldots, r_n\}$, *siendo* $r \in \{t, f\}$.

(a) será demostrado por inducción sobre k.

(a_0) El caso base $k = 0$ es trivial ya que en la configuración inicial tenemos: $\mathcal{C}_0(1) = \{\delta_0, \delta'_0, cod_0(\varphi)\} \cup \{\beta_{l,0} \mid 1 \leq l \leq n\}$ y existe una sola célula etiquetada por 2 que tiene los objetos $\alpha_1, \ldots, \alpha_{p+1}$ y los objetos $a_{1,0}, \ldots, a_{n,0}$. Entonces, la configuración $\mathcal{C}_0$ da lugar a la configuración $\mathcal{C}_1$ por la aplicación de las reglas:

$$[\, a_{1,0} \,]_2 \, [\, \gamma \,]_0 \to [\, a'_{1,0} t'_1 \,]_2 \, [\quad]_0$$

$$[\, a_{i,0} \,]_2 \, [\, \gamma \,]_0 \to [\, a'_{i,0} \,]_2 \, [\quad]_0 \quad \text{, para } 2 \leq i \leq n$$

$$[\, \alpha_j \,]_2 \, [\, \gamma \,]_0 \to [\, \alpha'_j \,]_2 \, [\quad]_0 \quad \text{, para } 1 \leq j \leq p+1$$

$$[\, \beta_{l,0} \,]_1 \, [\, \gamma \,]_0 \to [\, \beta'_{l,0} \,]_1 \, [\quad]_0 \quad \text{, para } 1 \leq l \leq n$$

$$\left.\begin{array}{l} [\, x_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, x'_{i,j,1} \,]_1 \, [\quad]_0 \\ [\, \overline{x}_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, \overline{x}'_{i,j,1} \,]_1 \, [\quad]_0 \\ [\, x^*_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, x^{*\prime}{}_{i,j,1} \,]_1 \, [\quad]_0 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

$$[\, \delta_0 \,]_1 \, [\, \gamma \,]_0 \to [\, \delta_1 \,]_1 \, [\quad]_0$$

$$[\, \delta'_0 \,]_1 \, [\, \gamma \,]_0 \to [\, \delta'_1 \,]_1 \, [\quad]_0$$

(a_1) Así, $\mathcal{C}_1(1) = \{\delta_1, \delta'_1, cod'_1(\varphi)\} \cup \{\beta'_{l,0} \mid 1 \leq l \leq n\}$ y en $\mathcal{C}_1$ hay una célula etiquetada por 2 tal que contiene el conjunto de objetos $\{a'_{1,0}, \ldots, a'_{n,0}\}$, el objeto t'_1 y los objetos $\alpha'_1, \ldots, \alpha'_{p+1}$. Por tanto, la configuración $\mathcal{C}_1$ da lugar a la configuración $\mathcal{C}_2$ por la aplicación de las reglas:

$$\begin{array}{l} [\, a'_{1,0} \,]_2 \, [\, \gamma \,]_0 \to [\, a_{1,1} f_1^R \,]_2 \, [\quad]_0 \\ [\, t'_1 \,]_2 \, [\, \gamma \,]_0 \to [\, t_1^L \,]_2 \, [\quad]_0 \end{array}$$

$$[\, a'_{i,0} \,]_2 \, [\, \gamma \,]_0 \to [\, a_{i,1}^L \, a_{i,1}^R \,]_2 \, [\quad]_0 \quad \text{, para} 2 \leq i \leq n$$

$$[\, \alpha'_j \,]_2 \, [\, \gamma \,]_0 \to [\, \alpha_j^L \alpha_j^R \,]_2 \, [\quad]_0 \quad \text{, para } 1 \leq j \leq p+1$$

$$[\, \beta'_{l,k} \,]_1 \, [\, \gamma \,]_0 \to [\, \beta^L_{l,k+1} \beta^R_{l,k+1} \,]_1 \, [\quad]_0 \quad \text{, para } k+1 \leq l \leq n$$

$$\left.\begin{array}{l} [\, x'_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, x''^2_{i,j,1} \,]_1 \, [\quad]_0 \\ [\, \overline{x}'_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, \overline{x}''^2_{i,j,1} \,]_1 \, [\quad]_0 \\ [\, x^{*\prime}{}_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, x^{*\prime\prime 2}_{i,j,1} \,]_1 \, [\quad]_0 \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$$

$$[\, \delta_1 \,]_1 \, [\, \gamma \,]_0 \to [\, \delta_2 \,]_1 \, [\quad]_0$$

$$[\, \delta'_1 \,]_1 \, [\, \gamma \,]_0 \to [\, \delta'^2_2 \,]_1 \, [\quad]_0$$

(a_2) Así, $\mathcal{C}_2(1) = \{\delta_2, \delta'^2_2, cod''_1(\varphi)\} \cup \{\beta^O_{l,1} \mid O \in \{L, R\}, 1 \leq l \leq n\}$ y en la $\mathcal{C}_2$ hay una célula etiquetada por 2 tal que contiene el conjunto de objetos

$\{a_{1,1}, \ldots, a_{n,1}\}$, los objetos t_1^L y f_1^R y los objetos $\alpha_1^O, \ldots, \alpha_{p+1}^O$, para $O \in \{L, R\}$. Por tanto, la configuración $\mathcal{C}_2$ da lugar a la configuración $\mathcal{C}_3$ por la aplicación de las reglas:

$$[\, a_{1,1} \,]_2 \to [\, \Gamma_0 \,]_2 \, [\, \Gamma_1 \,]_2 \quad \text{, para } 1 \leq i \leq n$$

$$\left.\begin{array}{l} [\, \beta_{1,1}^O \,]_1 \, [\quad]_0 \to [\quad]_1 \, [\, \beta_{1,1}^O \,]_0 \\ [\, \beta_{l,1}^O \,]_1 \, [\quad]_0 \to [\quad]_1 \, [\, \beta_{l,1} \,]_0 \end{array}\right\} \text{para } \begin{array}{l} O \in \{L, R\}, \\ k+1 \leq l \leq n \end{array}$$

$$\left.\begin{array}{l} [\, x''_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, x'''_{i,j,0} \,]_1 \, [\quad]_0 \\ [\, \overline{x}''_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, \overline{x}'''_{i,j,0} \,]_1 \, [\quad]_0 \\ [\, x^{*\prime\prime}_{i,j,0} \,]_1 \, [\, \gamma \,]_0 \to [\, x^{*\prime\prime\prime}_{i,j,0} \,]_1 \, [\quad]_0 \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p, \end{array}$$

$$[\, \delta_2 \,]_1 \, [\, \gamma \,]_0 \to [\, \delta_3 \,]_1 \, [\quad]_0$$

$$[\, \delta'_2 \,]_1 \, [\, \gamma \,]_0 \to [\, \delta'_3 \,]_1 \, [\quad]_0$$

(a_3) Así, $\mathcal{C}_3(1) = \{\delta_3, \delta'^2_3, cod'''_1(\varphi)\}$, en el entorno está el multiconjunto $\{\beta_{1,1}^O \mid O \in \{L, R\}\} \cup \{\beta_{l,1}^2 \mid 2 \leq l \leq n\}$ y en $\mathcal{C}_2$ hay dos células etiquetadas por 2 tales que contienen el conjunto de objetos $\{a_{2,1}^O, \ldots, a_{n,1}^O\}$ con $O = L$ (respectivamente, $O = R$), el objeto t_1^L (respectivamente, f_1^R) y los objetos $\alpha_1^O, \ldots, \alpha_{p+1}^O$, para $O = L$ (respectivamente, $O = R$). Por lo tanto, el resultado es válido para $k = 0$

- Suponiendo que, por inducción, el resultado es verdadero para k ($1 \leq k \leq n-1$); es decir,

  (a_0) Para cada $4k$ ($0 \leq k \leq n-1$) en la configuración $\mathcal{C}_{4k}$ tenemos lo siguiente:

  - $\mathcal{C}_{4k}(1) = \{\delta_{4k}, \delta'^{2^k}_{4k}, cod_k(\varphi)^{2^k}\} \cup \{\beta_{l,k}^{2^k} \mid k+1 \leq l \leq n\}$
  - Hay 2^k células etiquetadas por 2 tales que contienen
    - objetos $a_{k+1,k}, \ldots, a_{n,k}$;
    - objetos $r_1, \ldots, r_k$, siendo $r \in \{t, f\}$; y
    - objetos $\alpha_1, \ldots, \alpha_{p+1}$.

  (a_1) Para cada $4k+1$ ($0 \leq k \leq n-1$) at configuración $\mathcal{C}_{4k+1}$ tenemos lo siguiente:

  - $\mathcal{C}_{4k+1}(1) = \{\delta_{4k+1}, \delta'^{2^k}_{4k+1}, cod'_k(\varphi)^{2^k}\} \cup \{\beta'^{2^k}_{l,k} \mid k+1 \leq l \leq n\}$
  - Hay 2^k células etiquetadas por 2 tales que contienen
    - objetos $a'_{k+1,k}, \ldots, a'_{n,k}$;
    - objetos $r''_1, \ldots, r''_k$, siendo $r \in \{t, f\}$

- un objeto t'_{k+1}; y
- objetos $\alpha'_1, \ldots, \alpha'_{p+1}$.

(a_2) Para cada $4k+2$ ($0 \leq k \leq n-1$) en la configuración $\mathcal{C}_{4k+2}$ tenemos lo siguiente:

- $\mathcal{C}_{4k+2}(1) = \{\delta_{4k+2}, , {\delta'}^{2^{k+1}}_{4k+2} cod''_{k+1}(\varphi)^{2^{k+1}}\} \cup \{{\beta^O}^{2^k}_{l,k} \mid O \in \{L,R\}, k+1 \leq l \leq n\}$
- Hay 2^k células etiquetadas por 2 tales que contienen
  - objetos $a_{k+1,k}, \ldots, a_{n,k}$;
  - objetos $r_1, \ldots, r_k$, siendo $r \in \{t,f\}$; y
  - objetos $\alpha_1, \ldots, \alpha_{p+1}$.

(a_3) Para cada $4k+3$ ($0 \leq k \leq n-1$) en la configuración $\mathcal{C}_{4k+3}$ tenemos lo siguiente:

- $\mathcal{C}_{4k+3}(0) = \{{\beta^O}^{2^k}_{k+1,k+1}\} \cup \{\beta^{2^{k+1}}_{l,k+1} \mid k+2 \leq l \leq n\}$
- $\mathcal{C}_{4k+3}(1) = \{\delta_{4k+3}, {\delta'}^{2^{k+1}}_{4k+3}, cod'''_{k+1}(\varphi)^{2^{k+1}}\} \cup \{\beta^{2^k}_{l,k} \mid k+1 \leq l \leq n\}$
- Hay 2^{k+1} células etiquetadas por 2 tales que contienen
  - objetos $a_{k+1,k}, \ldots, a_{n,k}$;
  - objetos $r_1, \ldots, r_k$, siendo $r \in \{t,f\}$; y
  - objetos $\alpha_1, \ldots, \alpha_{p+1}$.

- Por tanto, por hipótesis de inducción, queremos probar el resultado para $k+1$.

(a_0) Entonces, la configuración $\mathcal{C}_{4k+3}$ da lugar a la configuración $\mathcal{C}_{4(k+1)}$ por la aplicación de las reglas:

$$\left.\begin{array}{l} [\, a^O_{i,j}\,]_2\,[\,\beta^O_{k+1,k+1}\,]_0 \to [\,a_{i,j}\,]_2\,[\quad]_0 \\ [\, r^O_i\,]_2\,[\,\beta^O_{k+1,k+1}\,]_0 \to [\,r_i\,]_2\,[\quad]_0 \end{array}\right\} \text{para } \begin{array}{l} O \in \{L,R\}, \\ r \in \{t,f\}, \\ 1 \leq i \leq n, \\ 1 \leq j \leq n \end{array}$$

$$[\,\alpha^O_j\,]_2\,[\,\beta^O_{k+1,k+1}\,]_0 \to [\,\alpha_j\,]_2\,[\quad]_0 \quad \text{, para } O \in \{L,R\}, 1 \leq j \leq p+1$$

$$\left.\begin{array}{l} [\, x'''_{i,j,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,x_{i,j,k+1}\,]_1\,[\quad]_0 \\ [\,\overline{x}'''_{i,j,k+1}\,]_1\,[\,\gamma\ \,]_0 \to [\ \overline{x}_{i,j,k+1}\,]_1\,[\quad]_0 \\ [\, {x^*}'''_{i,j,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,x^*_{i,j,k+1}\,]_1\,[\quad]_0 \end{array}\right\} \text{para } 1 \leq i \leq n, 1 \leq j \leq p$$

$$[\,\beta_{l,k+1}\,]_0\,[\quad]_1 \to [\quad]_0\,[\,\beta_{l,k+1}\,]_1 \quad \text{, para } k+2 \leq l \leq n$$

$$[\,\delta_{4k+3}\,]_1\,[\,\gamma\,]_0 \to [\,\delta_{4(k+1)}\,]_1\,[\quad]_0$$

$[\,\delta'_{4k+3}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'_{4(k+1)}\,]_1\,[\quad]_0$

Por lo tanto, lo siguiente es válido:

- $\mathcal{C}_{4(k+1)}(1) = \{\delta_{4(k+1)}, {\delta'}^{2^{k+1}}_{4(k+1)}, cod_{k+1}(\varphi)^{2^{k+1}}\} \cup \{\beta^{2^{k+1}}_{l,k+1} \mid k+2 \leq l \leq n\}$
- Hay 2^{k+1} células etiquetadas por 2 tales que contienen
  - objetos $a_{k+2,k+1}, \ldots, a_{n,k+1}$;
  - objetos $r_1, \ldots, r_{k+1}$, siendo $r \in \{t, f\}$; y
  - objetos $\alpha_1, \ldots, \alpha_{p+1}$.

(a_1) Entonces, la configuración $\mathcal{C}_{4(k+1)}$ da lugar a la configuración $\mathcal{C}_{4(k+1)+1}$ por la aplicación de las reglas:

$[\,a_{k+1,k}\,]_2\,[\,\gamma\,]_0 \to [\,a'_{k+1,k}t'_{k+1}\,]_2\,[\quad]_0$

$\left.\begin{array}{l}[\,t_i\,]_2\,[\,\gamma\,]_0 \to [\,t''_i\,]_2\,[\quad]_0 \\ [\,f_i\,]_2\,[\,\gamma\,]_0 \to [\,f''_i\,]_2\,[\quad]_0\end{array}\right\}$ para $1 \leq i \leq k$

$[\,a_{i,k+1}\,]_2\,[\,\gamma\,]_0 \to [\,a'_{i,k+1}\,]_2\,[\quad]_0$, para $2 \leq i \leq n$

$[\,\alpha_j\,]_2\,[\,\gamma\,]_0 \to [\,\alpha'_j\,]_2\,[\quad]_0$, para $1 \leq j \leq p+1$

$\left.[\,\beta_{l,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,\beta'_{l,k+1}\,]_1\,[\quad]_0\right\}$ para $k+2 \leq l \leq n$

$\left.\begin{array}{l}[\,x_{i,j,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,x'_{i,j,k+1}\,]_1\,[\quad]_0 \\ [\,\overline{x}_{i,j,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}'_{i,j,k+1}\,]_1\,[\quad]_0 \\ [\,x^*_{i,j,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,{x^*}'_{i,j,k+1}\,]_1\,[\quad]_0\end{array}\right\}$ para $1 \leq i \leq n, 1 \leq j \leq p$

$[\,\delta_{4(k+1)}\,]_1\,[\,\gamma\,]_0 \to [\,\delta_{4(k+1)+1}\,]_1\,[\quad]_0$

$[\,\delta'_{4(k+1)}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'_{4(k+1)+1}\,]_1\,[\quad]_0$

Por tanto, lo siguiente es válido:

- $\mathcal{C}_{4(k+1)+1}(1) = \{\delta_{4(k+1)+1}, {\delta'}^{2^k}_{4(k+1)+1}, cod'_{k+1}(\varphi)^{2^{k+1}}\} \cup \{{\beta'}^{2^k}_{l,k} \mid k+1 \leq l \leq n\}$
- Hay 2^{k+1} células etiquetadas por 2 tales que contienen
  - objetos $a'_{k+2,k+1}, \ldots, a'_{n,k+1}$;
  - objetos $r''_1, \ldots, r''_{k+1}$, siendo $r \in \{t, f\}$
  - un objeto t'_{k+2}; y
  - objetos $\alpha'_1, \ldots, \alpha'_{p+1}$.

(a_2) Then, configuración $\mathcal{C}_{4(k+1)+1}$ da lugar a la configuración $\mathcal{C}_{4(k+1)+2}$ por la aplicación de las reglas:

$[\,a'_{k+1,k}\,]_2\,[\,\gamma\,]_0 \to [\,a_{k+1,k+1}f^R_{k+1}\,]_2\,[\quad]_0$

$[\,t'_{k+1}\,]_2\,[\,\gamma\,]_0 \to [\,t^L_{k+1}\,]_2\,[\quad]_0$

$$\left.\begin{array}{l}[\,t_i''\,]_2\,[\,\gamma\,]_0 \to [\,t_i^L t_i^R\,]_2\,[\quad]_0 \\ [\,f_i''\,]_2\,[\,\gamma\,]_0 \to [\,f_i^L f_i^R\,]_2\,[\quad]_0\end{array}\right\} \text{para } 1 \leq i \leq k$$

$$[\,a'_{i,k+1}\,]_2\,[\,\gamma\,]_0 \to [\,a^L_{i,k+2}a^R_{i,k+2}\,]_2\,[\quad]_0 \quad \text{, para } 2 \leq i \leq n$$

$$[\,\alpha'_j\,]_2\,[\,\gamma\,]_0 \to [\,\alpha_j^L\alpha_j^R\,]_2\,[\quad]_0 \quad \text{, para } 1 \leq j \leq p+1$$

$$[\,\beta'_{l,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,\beta^L_{l,k+2}\beta^R_{l,k+2}\,]_1\,[\quad]_0 \quad \text{, para } k+2 \leq l \leq n$$

$$\left.\begin{array}{l}[\,x'_{i,j,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,x''^2_{i,j,k+2}\,]_1\,[\quad]_0 \\ [\,\overline{x}'_{i,j,k+1}\,]_1\,[\,\gamma\,\,]_0 \to [\,\overline{x}''^2_{i,j,k+2}\,]_1\,[\quad]_0 \\ [\,x^{*\prime}_{i,j,k+1}\,]_1\,[\,\gamma\,]_0 \to [\,x^{*\prime\prime 2}_{i,j,k+2}\,]_1\,[\quad]_0\end{array}\right\} \begin{array}{l}1 \leq i \leq n, \\ 1 \leq j \leq p\end{array}$$

$$[\,\delta_{4(k+1)+1}\,]_1\,[\,\gamma\,]_0 \to [\,\delta_{4(k+1)+2}\,]_1\,[\quad]_0$$

$$[\,\delta'_{4(k+1)+1}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'^2_{4(k+1)+2}\,]_1\,[\quad]_0$$

Por tanto, lo siguiente es válido:

- $\mathcal{C}_{4(k+1)+2}(1) = \{\delta_{4(k+1)+2}, \delta'^{2^{k+2}}_{4(k+1)+2}, cod''_{k+2}(\varphi)^{2^{k+2}}\} \cup \{\beta^{2^{k+1}}_{l,k} \mid k+1 \leq l \leq n\}$
- Hay 2^{k+1} células etiquetadas por 2 tales que contienen
  - objetos $a_{k+2,k+1}, \ldots, a_{n,k+1}$;
  - objetos $r_1, \ldots, r_{k+1}$, siendo $r \in \{t, f\}$; y
  - objetos $\alpha_1, \ldots, \alpha_{p+1}$.

(a_3) Entonces, la configuración $\mathcal{C}_{4(k+1)+2}$ da lugar a la configuración $\mathcal{C}_{4(k+1)+3}$ por la aplicación de las reglas:

$$[\,a_{k+1,k+1}\,]_2 \to [\,\Gamma_0\,]_2\,[\,\Gamma_1\,]_2 \quad \text{, para } 1 \leq i \leq n$$

$$\left.\begin{array}{l}[\,\beta^O_{k+1,k+1}\,]_1\,[\quad]_0 \to [\quad]_1\,[\,\beta^O_{k+1,k+1}\,]_0 \\ [\,\beta^O_{l,k+1}\,]_1\,[\quad]_0 \to [\quad]_1\,[\,\beta_{l,k+1}\,]_0\end{array}\right\} \text{para } O \in \{L, R\}, k+2 \leq l \leq n$$

$$\left.\begin{array}{l}[\,x''_{i,j,k+2}\,]_1\,[\,\gamma\,]_0 \to [\,x'''_{i,j,k+2}\,]_1\,[\quad]_0 \\ [\,\overline{x}''_{i,j,k+2}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}'''_{i,j,k+2}\,]_1\,[\quad]_0 \\ [\,x^{*\prime\prime}_{i,j,k+2}\,]_1\,[\,\gamma\,]_0 \to [\,x^{*\prime\prime\prime}_{i,j,k+2}\,]_1\,[\quad]_0\end{array}\right\} \text{para } \begin{array}{l}1 \leq i \leq n, \\ 1 \leq j \leq p\end{array}$$

$$[\,\delta_{4(k+1)+2}\,]_1\,[\,\gamma\,]_0 \to [\,\delta_{4(k+1)+3}\,]_1\,[\quad]_0$$

$$[\,\delta'_{4(k+1)+2}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'_{4(k+1)+3}\,]_1\,[\quad]_0$$

Por tanto, lo siguiente es válido:

- $\mathcal{C}_{4(k+1)+3}(0) = \{\beta^{O^{2^{k+1}}}_{k+2,k+2}\} \cup \{\beta^{2^{k+2}}_{l,k+2} \mid k+3 \leq l \leq n\}$
- $\mathcal{C}_{4(k+1)+3}(1) = \{\delta_{4(k+1)+3}, \delta'^{2^{k+2}}_{4(k+1)+3}, cod'''_{k+2}(\varphi)^{2^{k+2}}\} \cup \{\beta^{2^{k+1}}_{l,k+1} \mid k+2 \leq l \leq n\}$
- Hay 2^{k+2} células etiquetadas por 2 tales que contienen
  - objetos $a_{k+2,k+1}, \ldots, a_{n,k+1}$;

  - objetos $r_1, \ldots, r_{k+1}$, siendo $r \in \{t, f\}$; y
  - objetos $\alpha_1, \ldots, \alpha_{p+1}$.

- Para probar (b) es suficiente con darse cuenta de que, por una parte, de (a_3) la configuración ${\mathcal{C}_{4n-1}}^2$ lleva:

  - $\mathcal{C}_{4n-1}(1) = \{\delta_{4n-1}, {\delta'}^{2^n}_{4n-1}, cod'''_n(\varphi)\}$.
  - Hay 2^n células etiquetadas por 2 tales que contienen
    - un subconjunto diferente $\{r_1^O, \ldots, r_n^O\}$, siendo $r \in \{t, f\}$ y $O \in \{L, R\}$; y
    - objetos $\alpha^O, \ldots, \alpha_{p+1}^O$, for $O \in \{L, R\}$.

- Por otra parte, la configuración $\mathcal{C}_{4n-1}$ da lugar a la configuración $\mathcal{C}_{4n}$ por la aplicación de las reglas:

$$[\, r_i^O \,]_2\, [\, \beta_{n,n}^O \,]_0 \to [\, r_i \,]_2\, [\quad]_0 \quad \text{, para } \begin{array}{l} O \in \{L, R\}, \\ r \in \{t, f\}, \\ 1 \le i \le n \end{array}$$

$$[\, \alpha_j^O \,]_2\, [\, \beta_{n,n}^O \,]_0 \to [\, \alpha_j \,]_2\, [\quad]_0 \quad \text{, para } O \in \{L, R\}, 1 \le j \le p+1$$

$$\left.\begin{array}{l} [\, x'''_{i,j,n} \,]_1\, [\, \gamma \,]_0 \to [\, x_{i,j,n} \,]_1\, [\quad]_0 \\ [\, \overline{x}'''_{i,j,n} \,]_1\, [\, \gamma \,]_0 \to [\, \overline{x}_{i,j,n} \,]_1\, [\quad]_0 \\ [\, {x^*}'''_{i,j,n} \,]_1\, [\, \gamma \,]_0 \to [\, x^*_{i,j,n} \,]_1\, [\quad]_0 \end{array}\right\} \text{para } 1 \le i \le n, 1 \le j \le p$$

$$[\, \delta_{4n-1} \,]_1\, [\, \gamma \,]_0 \to [\, \delta_{4n} \,]_1\, [\quad]_0$$

$$[\, \delta'_{4n-1} \,]_1\, [\, \gamma \,]_0 \to [\, \delta'_{4n} \,]_1\, [\quad]_0$$

- Entonces, tenemos que $\mathcal{C}_{4n}(1) = \{\delta_{4n}, {\delta'}^{2^n}_{4n}, cod_{4n}(\varphi)^{2^n}\}$, y que hay 2^n células etiquetadas por 2 tales que contienen los objetos $\alpha_1, \ldots, \alpha_{p+1}$, así como un subconjunto diferente $\{r_1, \ldots, r_n\}$, siendo $r \in \{t, f\}$.

Primera fase de chequeo

Siguiente a la fase de generación viene la primera fase de chequeo, donde los objetos $c_{j,k}$ son creados para saber qué cláusula C_j ha sido satisfecha por la valoración de verdad codificada en las células etiquetadas por 2. En cada paso, disparamos reglas para una sola cláusula, con lo que en p pasos obtendremos objetos $c_{j,k}$ si la cláusula es satisfecha. Esto puede ser debido a dos razones:

[2] Aquí, $4n - 1 = 4k + 3$ para $k = n - 1$.

- El literal x_i aparece en la cláusula C_j, y el valor de la variable x_i en una valoración de verdad es verdadera. Entonces, podemos decir que dicha valoración de verdad satisface esa cláusula; o

- El literal $\neg x_i$ aparece en la cláusula C_j, y el valor de la variable x_i en l valoración de verdad es falsa. Entonces, podemos decir que dicha valoración de verdad satisface esa cláusula.

En cualquier otro caso, la variable x_i no aporta nada a la cláusula C_j. En el último paso de esta fase, las células etiquetadas por 2 tendrán objetos $c_{j,p}$ donde C_j son cláusulas satisfechas por dicha valoración de verdad. Además, tenemos un objeto α'_{p+1} para usarlo en la siguiente fase.

Proposición 8.2. *Sea* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ *una computación del sistema* $\Pi(s(\varphi))$ *con multiconjunto de entrada* $cod(\varphi)$.

(a) *Para cada* k *(*$0 \leq k \leq p-1$*) en la configuración* $\mathcal{C}_{4n+k}$ *tenemos lo siguiente:*

- $\mathcal{C}_{4n+k}(1) = \{\delta_{4n+k}, \delta'^{2^n}_{4n+k}, cod^k_n(\varphi)^{2^n}\}$
- *Hay* 2^n *células etiquetadas por 2 tales que contienen*
  - *objetos* $r_1, \ldots, r_n$*, siendo* $r \in \{t, f\}$*;*
  - *objetos* $\alpha_1, \ldots, \alpha_{p+1}$*; y*
  - *objetos* $c_{1,k}, \ldots, c_{k,k}$*, donde* $c_{j,k}$ *representa que la cláusula* C_j *ha sido satisfecha por la valoración de verdad codificada en dicha célula.*

(b) $\mathcal{C}_{4n+p}(1) = \{\delta_{4n+p}, \delta'^{2^n}_{4n+p}\}$*, y hay* 2^n *células etiquetadas por 2 tales que cada una de ellas contiene los objetos* $\alpha_1, \ldots, \alpha_{p+1}$*, un subconjunto distinto* $\{r_1, \ldots, r_n\}$ *y objetos* c_j *cuando la cláusula* C_j *sea satisfecha en esa célula.*

(a) será demostrado por inducción sobre k.

(a) El caso base $k = 0$ es trivial ya que en la configuración inicial tenemos: $\mathcal{C}_{4n}(1) = \{\delta_{4n}, \delta'^{2^n}_{4n}, cod_{4n}(\varphi)\}$ y hay 2^n células etiquetadas por 2 que contienen los objetos $\alpha_1, \ldots, \alpha_{p+1}$ y un subconjunto distinto $\{r_1, \ldots, r_n\}$, siendo $r \in \{t, f\}$. Entonces, la configuración $\mathcal{C}_{4n}$ da lugar a la configuración $\mathcal{C}_{4n+1}$ por la aplicación de las reglas:

$$\left.\begin{array}{l}
[\,t_i\,]_2\,[\,x_{i,1,n}\,]_1 \to [\,c_{1,1}t_i\,]_2\,[\quad]_1\\
[\,t_i\,]_2\,[\,\overline{x}_{i,1,n}\,]_1 \to [\,t_i\,]_2\,[\quad]_1\\
[\,t_i\,]_2\,[\,x^*_{i,1,n}\,]_1 \to [\,t_i\,]_2\,[\quad]_1\\
[\,f_i\,]_2\,[\,x_{i,1,n}\,]_1 \to [\,f_i\,]_2\,[\quad]_1\\
[\,f_i\,]_2\,[\,\overline{x}_{i,1,n}\,]_1 \to [\,c_{1,1}f_i\,]_2\,[\quad]_1\\
[\,f_i\,]_2\,[\,x^*_{i,1,n}\,]_1 \to [\,f_i\,]_2\,[\quad]_1
\end{array}\right\} \text{para } 1 \le i \le n, 1 \le j \le p$$

$$\left.\begin{array}{l}
[\,x_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,x_{i,j,n+k+1}\,]_1\,[\quad]_0\\
[\,\overline{x}_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}_{i,j,n+k+1}\,]_1\,[\quad]_0\\
[\,x^*_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,x^*_{i,j,n+k+1}\,]_1\,[\quad]_0
\end{array}\right\} \text{para } 1 \le i \le n, 2 \le j \le p$$

$$[\,\delta_{4n}\,]_1\,[\,\gamma\,]_0 \to [\,\delta_{4n+1}\,]_1\,[\quad]_0$$

$$[\,\delta'_{4n}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'_{4n+1}\,]_1\,[\quad]_0$$

Así, $\mathcal{C}_{4n+1}(1) = \{\delta_{4n+1}, {\delta'}^{2^n}_{4n+1}, cod^2_{4n+1}(\varphi)^{2^n}\}$ y en $\mathcal{C}_{4n+1}$ hay 2^n células etiquetadas por 2 tales que contienen los objetos $\alpha_1, \ldots, \alpha_{p+1}$, un subconjunto distinto $\{r_1, \ldots, r_n\}$, siendo $r \in \{t, f\}$ y objetos $c_{1,1}$ si algún literal presente en C_1 la satisface[3]. Por tanto, el resultado es válido para $k = 1$.

Suponiendo que, por inducción, el resultado es verdadero para k ($0 \le k \le p-1$); es decir,

- $\mathcal{C}_{4n+k}(1) = \{\delta_{4n+k}, {\delta'}^{2^n}_{4n+k}, cod^{k+1}_n(\varphi)^{2^k}\}$
- Hay 2^n células etiquetadas por 2 tales que contienen
  - objetos $r_1, \ldots, r_n$, siendo $r \in \{t, f\}$;
  - objetos $\alpha_1, \ldots, \alpha_{p+1}$; y
  - objetos $c_{1,k}, \ldots, c_{k,k}$, donde $c_{j,k}$ representa que la cláusula C_j ha sido satisfecha por la valoración de verdad codificada en dicha célula.

Entonces, la configuración $\mathcal{C}_{4n+k}$ da lugar a la configuración $\mathcal{C}_{4n+k+1}$ por la aplicación de las reglas:

$$\left.\begin{array}{l}
[\,t_i\,]_2\,[\,x_{i,k+1,n+k}\,]_1 \to [\,c_{k+1,k+1}t_i\,]_2\,[\quad]_1\\
[\,t_i\,]_2\,[\,\overline{x}_{i,k+1,n+k}\,]_1 \to [\,t_i\,]_2\,[\quad]_1\\
[\,t_i\,]_2\,[\,x^*_{i,k+1,n+k}\,]_1 \to [\,t_i\,]_2\,[\quad]_1\\
[\,f_i\,]_2\,[\,x_{i,k+1,n+k}\,]_1 \to [\,f_i\,]_2\,[\quad]_1\\
[\,f_i\,]_2\,[\,\overline{x}_{i,k+1,n+k}\,]_1 \to [\,c_{k+1,k+1}f_i\,]_2\,[\quad]_1\\
[\,f_i\,]_2\,[\,x^*_{i,k+1,n+k}\,]_1 \to [\,f_i\,]_2\,[\quad]_1
\end{array}\right\} \text{para } 1 \le i \le n, 1 \le j \le p$$

[3] Aquí, los objetos # son creados, pero no serán usados más a lo largo de la computación, así que no serán anotados en esta verificación.

$$\left.\begin{array}{l}[\,x_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,x_{i,j,n+k+1}\,]_1\,[\quad]_0\\ [\,\overline{x}_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,\overline{x}_{i,j,n+k+1}\,]_1\,[\quad]_0\\ [\,x^*_{i,j,n+k}\,]_1\,[\,\gamma\,]_0 \to [\,x^*_{i,j,n+k+1}\,]_1\,[\quad]_0\end{array}\right\} \text{para } 1 \le i \le n, k+2 \le j \le p$$

$$[\,c_{j,k}\,]_2\,[\,\gamma\,]_0 \to [\,c_{j,k+1}\,]_2\,[\quad]_0 \quad \text{, para } 1 \le j \le p, k \le k \le p-1$$

$$[\,\delta_{4n+k}\,]_1\,[\,\gamma\,]_0 \to [\,\delta_{4n+k+1}\,]_1\,[\quad]_0$$

$$[\,\delta'_{4n+k}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'_{4n+k+1}\,]_1\,[\quad]_0$$

Así, $\mathcal{C}_{4n+k+1}(1) = \{\delta_{4n+k+1}, {\delta'}^{2^n}_{4n+k+1}, cod^{k+2}_{4n+k+1}(\varphi)^{2^n}\}$ y in $\mathcal{C}_{4n+k+1}$ hay 2^n células etiquetadas por 2 tales que contienen los objetos $\alpha_1, \ldots, \alpha_{p+1}$, un subconjunto distinto $\{r_1, \ldots, r_n\}$, siendo $r \in \{t, f\}$ y objetos $c_{1,k}, \ldots, c_{k,k}$ si algún literal presente en C_j las satisface.

Para demostrar (b) es suficiente darse cuenta de que, por una parte, de (a) la configuración $\mathcal{C}_{4n+p-1}$ lleva:

- $\mathcal{C}_{4n+p-1}(1) = \{\delta_{4n+p-1}, {\delta'}^{2^n}_{4n+p-1}, cod^p_n(\varphi)^{2^n}\}$
- Hay 2^n células etiquetadas por 2 tales que contienen
  - objetos $r_1, \ldots, r_n$, siendo $r \in \{t, f\}$;
  - objetos $\alpha_1, \ldots, \alpha_{p+1}$; y
  - objetos $c_{1,p-1}, \ldots, c_{p-1,p-1}$, donde $c_{j,p-1}$ representa que la cláusula C_j ha sido satisfecha por la valoración de verdad codificada en dicha célula.

Por otra parte, la configuración $\mathcal{C}_{4n+p-1}$ da lugar a la configuración $\mathcal{C}_{4n+p}$ por la aplicación de las reglas:

$$\left.\begin{array}{l}[\,t_i\,]_2\,[\,x_{i,p,n+p-1}\,]_1 \to [\,c_{p,p}t_i\,]_2\,[\quad]_1\\ [\,t_i\,]_2\,[\,\overline{x}_{i,p,n+p-1}\,]_1 \to [\,t_i\,]_2\,[\quad]_1\\ [\,t_i\,]_2\,[\,x^*_{i,p,n+p-1}\,]_1 \to [\,t_i\,]_2\,[\quad]_1\\ [\,f_i\,]_2\,[\,x_{i,p,n+p-1}\,]_1 \to [\,f_i\,]_2\,[\quad]_1\\ [\,f_i\,]_2\,[\,\overline{x}_{i,p,n+p-1}\,]_1 \to [\,c_{p,p}f_i\,]_2\,[\quad]_1\\ [\,f_i\,]_2\,[\,x^*_{i,p,n+p-1}\,]_1 \to [\,f_i\,]_2\,[\quad]_1\end{array}\right\} \text{para } 1 \le i \le n, 1 \le j \le p$$

$$[\,c_{j,p-1}\,]_2\,[\,\gamma\,]_0 \to [\,c_{j,p}\,]_2\,[\quad]_0 \quad \text{, para } 1 \le j \le p-1$$

$$[\,\delta_{4n+p-1}\,]_1\,[\,\gamma\,]_0 \to [\,\delta_{4n+p}\,]_1\,[\quad]_0$$

$$[\,\delta'_{4n+p-1}\,]_1\,[\,\gamma\,]_0 \to [\,\delta'_{4n+p}\,]_1\,[\quad]_0$$

Entonces, tenemos $\mathcal{C}_{4n+p}(1) = \{\delta_{4n+p}, {\delta'}^{2^n}_{4n+p}\}$, y en $\mathcal{C}_{4n+p}$ hay 2^n células etiquetadas por 2 tales que cada una de ellas contiene un subconjunto distinto

$\{r_1, \ldots, r_n\}$, siendo $r \in \{t, f\}$[4], objetos $\alpha_1, \ldots, \alpha_{p+1}$ y objetos $c_{j,p}$ cuando la cláusula C_j haya sido satisfecha por la valoración de verdad codificada en dicha célula been satisfied by the truth assignment encoded in such membrane.

Segunda fase de chequeo

Aquí, las reglas de 3.1 son disparadas en el paso $(4n + p + 1)$, los objetos α_j en las células etiquetadas por 2 son eliminados si y sólo si la valoración de verdad asociados a dicha célula hace verdadera la cláusula C_j; es decir, si existe al menos un objeto c_j en dicha célula. En la configuración $\mathcal{C}_{4n+p}$ tenemos $\mathcal{C}_{4n+p}(1) = \{\delta_{4n+p}, \delta'^{2^n}_{4n+p}\}$ y cada célula etiquetada por 2 contiene objetos $\alpha_1, \ldots, \alpha_p$ y objetos c_j tales que la valoración de verdad correspondiente satisface la cláusula C_j. Por la aplicación de las reglas de 3.1 y la regla $[\,\delta_{4n+p}\,]_1\,[\,\gamma\,]_0 \rightarrow [\,\delta_{4n+p+1}\,]_1\,[\quad]_0$, el objeto δ_{4n+p} evoluciona en δ_{4n+p+1} en la célula etiquetada por 1, y en cada célula etiquetada por 2, los objetos α_j tales que su correspondiente objeto $c_{j,p}$ son «eliminados» del sistema, y dejan paso a la siguiente fase para chequear si están o no presentes, además del objeto α_{p+1}, que se prepara, evolucionando a α'_{p+1}, para cooperar con los objetos α_j restantes. Esta fase dura exactamente un paso de computación.

Fase de salida

La fase de salida comienza en el paso $(4n + p + 2)$, y toma exactamente 4 pasos, independientemente de si la fórmula es satisfactible o no por alguna valoración de verdad.

- *Respuesta afirmativa*: Si la fórmula φ es satisfactible, entonces al menos una de las valoraciones de verdad de una célula con etiqueta 2 ha satisfecho todas las cláusulas. Entonces, habrá una célula etiquetada por 2 tal que los objetos α_j, con $1 \leq j \leq p$ habrán desaparecido en el paso anterior. En la configuración $\mathcal{C}_{4n+p+1}$, tenemos $\mathcal{C}_{4n+p+1}(1) = \{\delta_{4n+p+1}\}$ y en cada célula etiquetada por 2 quedan objetos α_j si la valoración de verdad correspondiente **no** hace verdadera la cláusula C_j y un objeto α'_{p+1}. En este paso, sólo la regla $[\,\delta_{4n+p+1}\,]_1\,[\,\gamma\,]_0 \rightarrow [\,\delta_{4n+p+2}\,]_1\,[\quad]_0$ será disparada y las reglas $[\,\alpha_j\alpha'_{p+1}\,]_2\,[\quad]_0 \rightarrow [\quad]_2\,[\,n_1\,]_0$ serán disparadas en las células etiquetadas por 2 tales que al menos una cláusula no sea satisfecha por la valoración de verdad correspondiente. Entonces, en la configuración $\mathcal{C}_{4n+p+2}$, tenemos $\mathcal{C}_{4n+p+2}(1) = \{\delta_{4n+p+2}, n_1^t\}$, siendo t el número de valoraciones de verdad que tienen al menos una cláusula **no** satisfecha por

[4]Este conjunto no será usado más, así que no se tendrá más en cuenta.

la valoración de verdad correspondiente, y las células etiquetadas por 2 contienen un objeto α'_{p+1} si y sólo si la valoración de verdad correspondiente hace verdaderas todas las cláusulas de φ, y pueden tener objetos α_j, $1 \leq j \leq p$, si la cláusula C_j no es satisfecha por la valoración de verdad correspondiente.

En el siguiente paso, aplicando las reglas $[\,]_0[\quad]_2 \rightarrow [\quad]_0[\,n_1\,]_2 n_1$ y $[\,\alpha'_{p+1}\,]_2[\,\delta_{4n+p+2}\,]_1 \rightarrow [\,y_1\,]_2[\quad]_1$, obtenemos un objeto y_1 en una célula etiquetada por 2 si y sólo si la valoración de verdad correspondiente hace verdadera la fórmula φ. Es importante tener en cuenta que más de una célula etiquetada por 2 puede tener una valoración de verdad que haya verdadera la fórmula φ, pero como en ese caso, lo que queremos es saber si existe *al menos* una valoración de verdad que haga verdadera a la fórmula φ, sólo queremos un objeto y_1. Entonces, en la configuración $\mathcal{C}_{4n+p+3}$ tenemos $\mathcal{C}_{4n+p+3}(1) = \emptyset$ y en las células etiquetadas por 2, podemos tener objetos n_1, hasta t en todas las células etiquetadas por 2, siendo t el número de valoraciones de verdad que no hacen verdadera la fórmula, un objeto α'_{p+1} si la valoración de verdad correspondiente hace verdaderas todas las cláusulas, excepto una célula etiquetada por 2 cuya valoración de verdad haga verdadera la fórmula φ que tendrá un objeto y_1, y puede tener objetos α_j, $1 \leq j \leq p$, si la cláusula C_j no es satisfecha por la valoración de verdad correspondiente. En el siguiente paso, la única regla aplicable es $[\,y_1\,]_2[\,\gamma\,]_0 \rightarrow [\,y_2\,]_2[\quad]_0$, que será útil para sincronizar la respuesta positiva y negativa. Hay que tener en cuenta que la regla $[\,n_1\,]_2[\,\delta_{4n+p+2}\,]_1 \rightarrow [\,n_2\,]_2[\quad]_1$ no puede ser disparada ya que el objeto δ_{4n+3} ha sido consimido en el paso previo por un objeto α'_{p+1}. Por lo tanto, en la configuración $\mathcal{C}_{4n+p+4}$, tenemos $\mathcal{C}_{4n+p+4}(1) = \emptyset$ y en las células etiquetadas por 2, podemos tener objetos n_1, hasta t en todas las células etiquetadas por 2, siendo t el número de valoraciones de verdad que no hacen cierta la fórmula φ, un objeto α'_{p+1} si la valoración de verdad codificada en dicha célula hace verdadera la fórmula φ, excepto una de ellas que tendrá un objeto y_2, y puede tener objetos α_j, $1 \leq j \leq p$, si la cláusula C_j no es satisfecha por la valoración de verdad correspondiente. En el último paso de la computación, la regla $[\,y_2\,]_2[\quad]_0 \rightarrow [\quad]_2[\,\texttt{yes}\,]_0$ es disparada, enviando un objeto `yes` al entorno. Entonces, en la configuración $\mathcal{C}_{4n+p+5}$, tenemos $\mathcal{C}_{4n+p+5}(1) = \emptyset$ y en las células etiquetadas por 2, podemos tener objetos n_1, hasta t en todas las células etiquetadas por 2, siendo t el número de valoraciones de verdad que no hacen verdadera la fórmula φ, un objeto α'_{p+1} si la valoración de verdad correspondiente hace verdaderas todas las cláusulas de la fórmula φ, excepto una de ellas,

y puede tener objetos α_j, $1 \leq j \leq p$, si la cláusula C_j no es satisfecha por la valoración de verdad correspondiente, y habrá un objeto `yes` en el entorno. Aquí, la computación para y devuelve una respuesta afirmativa.

- *Respuesta negativa*: Si la fórmula φ no es satisfactible entonces ninguna de las valoraciones de verdad codificadas en las células etiquetadas por 2 hace la fórmula φ verdadera. Así, al menos un objeto α_j ($1 \leq j \leq p$) estará contenido en todas las células etiquetadas por 2. En la configuración $\mathcal{C}_{4n+p+1}$, tenemos $\mathcal{C}_{4n+p+1}(1) = \{\delta_{4n+p+1}\}$ y en cada célula etiquetada por 2 quedarán objetos α_j si la valoración de verdad correspondiente **no** hace verdadera la cláusula C_j. En este paso, sólo las reglas $[\,\alpha_j\alpha'_{p+1}\,]_2[\quad]_0 \rightarrow [\quad]_2[\,n_1\,]_0$, para $1 \leq j \leq p$ y la regla $[\,\delta_{4n+p+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta_{4n+p+2}\,]_1[\quad]_0$ serán disparadas. Por tanto, en la configuración $\mathcal{C}_{4n+p+2}$ tenemos en el entorno 2^n copias del objeto n_1, $\mathcal{C}_{4n+p+2}(1) = \{\delta_{4n+p+2}\}$ y las células etiquetadas por 2 tendrán objetos α_j ($1 \leq j \leq p$) cuando las cláusulas C_j no sean satisfechas por la valoración de verdad correspondiente. En el paso $(4n + p + 3)$, la regla $[\,]_0[\quad]_2 \rightarrow [\quad]_0[\,n_1\,]_2 n_1$ será disparada. Aquí, los objetos n_1 serán enviados a la célula etiquetada por 2. Entonces, en la configuración $\mathcal{C}_{4n+p+3}$ tenemos $\mathcal{C}_{4n+p+3}(1) = \{\delta_{4n+p+2}\}$ y las células etiquetadas por 2 contendrán objetos α_j ($1 \leq j \leq p$) si la cláusula C_j no es satisfecha por la valoración de verdad correspondiente, y pueden tener t objetos n_1 ($0 \leq t \leq 2^n$). En el paso $(4n + p + 4)$ la regla $[\,n_1\,]_2[\,\delta_{4n+p+2}\,]_1 \rightarrow [\,n_2\,]_2[\quad]_1$ es disparada, dado que el objeto δ_{4n+3} no ha sido consumida por ninguna regla de 4.3, creando un objeto n_2 en dicha célula etiquetada por 2. Por tanto, en la configuración $\mathcal{C}_{4n+p+4}$ tenemos $\mathcal{C}_{4n+p+4}(1) = \emptyset$ y las células etiquetadas por 2 tendrán objetos α_j ($1 \leq j \leq p$) si la cláusula C_j no es satisfecha por la valoración de verdad correspondiente, y pueden tener t objetos n_1 ($0 \leq t \leq 2^n$), y una de ellas tendrá un objeto n_2. En el último paso de computación, la regla $[\,n_2\,]_2[\quad]_0 \rightarrow [\quad]_2[\,\mathtt{no}\,]_0$ es disparada, enviando un objeto `no` al entorno. Por tanto, en la configuración $\mathcal{C}_{4n+p+5}$ tenemos $\mathcal{C}_{4n+p+5}(1) = \emptyset$ y las células etiquetadas por 2 2 tendrán objetos α_j ($1 \leq j \leq p$) si la cláusula C_j no es satisfecha por la valoración de verdad correspondiente, y puede tener t objetos n_1 ($0 \leq t \leq 2^n$), y habrá un objeto `no` en el entorno. Aquí, la computación para y devuelve una respuesta negativa.

Teorema 8.2. $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{TSEC}(2,2)}$

Demostración. La familia de sistemas P construida previamente verifica lo siguiente:

- Todo sistema de la familia $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ es un sistema P reconocedor de $\mathcal{TDEC}(2,1)$.
- La familia $\mathbf{\Pi}$ es polinomialmente uniforme por máquinas de Turing dado que para cada $n, p \in \mathbb{N}$, las reglas de $\Pi(\langle n, p\rangle)$ de la familia son recursivamente definidas por $n, p \in \mathbb{N}$, y la cantidad de recursos necesitados para construir un elemento de la familia es de orden polinomial con respecto a n y a p, como se muestra debajo:
  - Tamaño del alfabeto: $\Theta(max\{p^2, n^2p\})$.
  - Número inicial de células: $2 \in \Theta(1)$.
  - Número inicial de objetos en células: $n^2 + np + n + p + 3 \in \Theta(max\{n^2, np\})$.
  - Número de reglas: $\Theta(max\{n^3, n^2p\})$.
  - Número máximo de objetos involucrados en cualquier regla: $4 \in \Theta(1)$.
- El par de funciones computables en tiempo polinomial (cod, s) definido cumple lo siguiente: para cada fórmula φ del problema `SAT`, $s(\varphi)$ es un número natural, $cod(\varphi)$ es un multiconjunto de entrada del sistema $\Pi(s(\varphi))$ y para cada $t \in \mathbb{N}$, $s^{-1}(t)$ es un conjunto finito.
- La familia $\mathbf{\Pi}$ está polinomialmente acotada en tiempo: de hecho, para cada fórmula φ del problema `SAT`, el sistema P reconocedor $\Pi(s(\varphi)) + cod(\varphi)$ tarda exactamente $\lfloor \frac{3np}{2} \rfloor + np + 6$ pasos de computación en devolver una respuesta positiva o negativa, siendo n el número de variables y p el número de cláusulas de φ.
- La familia $\mathbf{\Pi}$ es adecuada con respecto a (X, cod, s): de hecho, para cada fórmula φ, si las computaciones de $\Pi(s(\varphi)) + cod(\varphi)$ son de *aceptación*, entonces φ es satisfactible.
- La familia $\mathbf{\Pi}$ es completa con respecto a (X, cod, s): de hecho, para cada fórmula φ que sea satisfactible, todas las computaciones de $\Pi(s(\varphi)) + cod(\varphi)$ son de *aceptación*.

□

Corolario 8.3. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{TSEC}(2,2)}$

Demostración. Basta con observar que `SAT` es un problema **NP**-completo, $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{TDEC}(2,1)}$ y la clase de complejidad $\mathbf{PMC}_{\mathcal{TDEC}(2,1)}$ está cerrada bajo reducibilidad en tiempo polinomial y bajo complementario. □

8.3. El papel del entorno en los sistemas P a modo de tejidos con reglas de comunicación evolutiva

En el marco de los sistemas P a modo de tejidos con reglas symport/antiport clásicos, el papel del entorno era diferente dependiendo del uso de reglas de división o separación. Mientras que usando reglas de división, el entorno pasaba a tener un papel irrelevante [49], el cambio de estas reglas por reglas de separación hace que pasar de no usar el entorno [27] a usar el entorno [54] (cuando las reglas symport/antiport pueden tener tamaño mayor o igual que 3) equivale a pasar de la *no eficiencia* a la *presumible eficiencia*.

En esta sección mostraremos como el uso del entorno en los sistemas P a modo de tejidos con reglas symport/antiport evolutivas no es determinante para la resolución eficiente de problemas presuntamente intratables.

Teorema 8.3. $\mathbf{PMC}_{\mathcal{TSEC}(k_1,k_2)} = \mathbf{PMC}_{\widehat{\mathcal{TSEC}}(k_1,k_2)}$

Teorema 8.4. $\mathbf{PMC}_{\mathcal{TDEC}(k_1,k_2)} = \mathbf{PMC}_{\widehat{\mathcal{TDEC}}(k_1,k_2)}$

Probaremos ambos teoremas en paralelo.

Demostración. Por una parte, siendo $\mathcal{F} \in \{\mathcal{D}, \mathcal{S}\}$, es fácil ver que $\mathbf{PMC}_{\widehat{\mathcal{TFEC}}(k_1,k_2)} \subseteq \mathbf{PMC}_{\mathcal{TFEC}(k_1,k_2)}$.

Por otra parte, vamos a probar como estos sistemas pueden evitar el uso del entorno mediante el uso de recursos precomputados en tiempo de ejecución.

En la solución proporcionada en la sección 8.2 mediante sistemas de $\mathcal{TSEC}(2,2)$, sólo se usan objetos γ para evolucionar, en cierta manera, otros objetos del sistema. De hecho, el sistema no podría disparar ninguna regla desde la configuración inicial de no ser por la existencia de este objeto en el entorno. Usaremos esto como ventaja para la simulación del entorno mediante la precarga de una cantidad suficiente de objetos γ. Aquí, necesitaremos el siguiente número de objetos γ:

objetos \ steps	$4k+1$	$4k+2$	$4k+3$	$4k$
$\alpha_1 \dots \alpha_p \alpha_{p+1}$	$(p+1)2^k$	$(p+1)2^k$		
$a_1 \dots a_n$	$n2^k$	$n2^k$		
$\beta_{l,k}$	$n(n+p+1)2^k$	$n(n+p+1)2^k$		
δ'_k	2^k	2^k	2^{k+1}	2^{k+1}
$x_{i,j,k}$	$np2^k$	$np2^k$	$np2^{k+1}$	$np2^{k+1}$

Tenemos que añadir, además, $4n + p + 1$ objetos γ para la evolución del objeto δ_k.

Recordando la serie

$$\sum_{k=0}^{n} 2^k = 2^{n-1} - 1,$$

para obtener la cantidad necesaria de objetos, vamos a generar 2^{n+2} objetos para estar seguros que tendremos suficientes para «evolucionar» todos los objetos del sistema a lo largo de la computación. Para este propósito, añadimos dos células al sistema, etiquetadas por 3 y 4, y con $\mathcal{M}_3 = \{\gamma^{2n^2+6np+4n+2p+6}\}$ y $\mathcal{M}_4 = \emptyset$. El número de células y de objetos iniciales se mantiene con orden polinomial. Además, se añaden las siguientes reglas a $\mathcal{R}$:

$$\left.\begin{array}{l} [\,\gamma_{2k}\,]_3\,[\quad]_4 \to [\quad]_3\,[\,\gamma_{2k+1}^2\,]_4 \\ [\,\gamma_{2k+1}\,]_4\,[\quad]_3 \to [\quad]_4\,[\,\gamma_{2(k+1)}^2\,]_3 \end{array}\right\} \text{para } 0 \leq k \leq n$$

En el n-ésimo paso, tendremos $(2n^2+6np+4n+2p+6)2^{2n+1}$ objetos γ_{2n+1} en la célula etiquetada por 3, suficientes para que cumplan el mismo rol que el objeto γ en la solución original. Entonces, tenemos que cambiar las reglas que interactúan con el objeto γ en el entorno y hacer que interactúen con el objeto γ_{2n+1} en la célula etiquetada por 3. Por tanto, hasta que los objetos γ_{2n+1} no sean generados, el sistema no evolucionará y, en cuanto estos sean generados, el sistema comenzará a funcionar de manera similar a como lo hacía en la solución original.

En el marco de $\mathcal{TDEC}(2,1)$, los únicos objetos que usan a γ para evolucionar son los objetos $x_{i,j,k}$, $\overline{x}_{i,j,k}$ y $x^*_{i,j,k}$ y los objetos δ_k y δ'_k. El número de objetos γ necesarios aquí es $np \cdot (np-1) + 2np + 3 + 2np + 1 = n^2p^2 + 3np + 4$, así que generaremos $7np$ objetos γ que harán el papel de sus análogos en el entorno en la solución original. En este caso sólo añadimos una nueva célula etiquetada por $np+4$ con $\mathcal{M}_{np+4} = \{\gamma^{n^2p^2+3np+4}\}$, y las reglas que interactúan con los objetos γ en el entorno en la solución original pasarán a interactuar con los mismos objetos, pero esta vez situados en la célula etiquetada por $np + 4$. El sistema tendrá el mismo comportamiento que en el diseño de la Sección 8.1. □

8.4. Nuevas fronteras obtenidas

En este capítulo se han refinado las fronteras previamente conocidas en el marco de los sistemas P a modo de tejidos con reglas de comunicación evolu-

tiva. Además, se han establecido nuevos resultados de estos sistemas cuando el entorno juega un papel *pasivo*.

Usando reglas de división, quedaban abiertas algunas incógnitas, dado que la mejor solución a un problema presuntamente intratable que se había dado era mediante sistemas de $\mathcal{TDEC}(4)/\mathcal{TDEC}(3,2)$. Por tanto, las clases de complejidad $\mathbf{PMC}_{\mathcal{TDEC}(3)}$, $\mathbf{PMC}_{\mathcal{TDEC}(2,k)}(k \geq 1)$ y $\mathbf{PMC}_{\mathcal{TDEC}(k,1)}(k \geq 2)$ eran desconocidas. Dada la solución mediante una familia de sistemas P de $\mathcal{TDEC}(2,1)$, ambas incógnitas han desaparecido, dado que SAT puede ser resuelto en ambos marcos. Siendo $k_1 \geq 1, k_2 \geq 2$:

Corolario 8.4. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{TDEC}(2,k_1)} \cap \mathbf{PMC}_{\mathcal{TDEC}(k_2,1)}$

Además, junto con este, hay otro resultado que es inmediato mediante la solución, dado que la longitud máxima de las reglas de comunicación evolutiva de los sistemas de $\mathcal{TDEC}(2,1)$ por la métrica de [60] es 3:

Corolario 8.5. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{TDEC}(3)}$

Por tanto, dos nuevas fronteras son obtenidas en este marco:

- Mientras que una longitud de la parte izquierda de la regla restringida a 1 permite sólo la resolución eficiente de problemas de la clase **P**, el paso a tener reglas de comunicación evolutiva con la longitud de la parte izquierda igual a 2 implica la capacidad de poder resolver eficientemente problemas presuntamente intratables.
- El paso de tener reglas de longitud total máxima 2 a 3 proporciona a los sistemas de membranas correspondientes la capacidad de resolver eficientemente sólo problemas de la clase **P** a problemas **NP**-completos.

El uso de las reglas de separación dejaba la clase $\mathbf{PMC}_{\mathcal{TSEC}(2,k)}(k \geq 2)$ como una incógnita. Dada la solución del problema SAT en $\mathcal{TSEC}(2,2)$, teniendo que $k \geq 2$, podemos concluir lo siguiente:

Corolario 8.6. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{TSEC}(2,k)}$

Entonces, dados los nuevos resultados, las siguientes fronteras «afinan» los resultados anteriores:

- Dado que los sistemas que utilizan comunicación evolutiva con longitud $(k,1), k \geq 1$ están restringidos a la clase **P**, la solución al problema SAT en el mismo marco con reglas de longitud $(2,2)$ genera una frontera mínima en el paso de $k = 1$ a $k = 2$ en $(2,k)$.

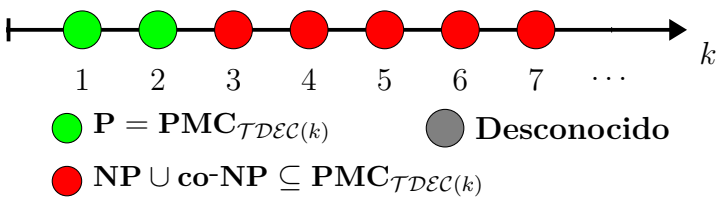


Figura 8.1: Fronteras de la eficiencia en sistemas P de tejidos con reglas de comunicación evolutiva y división celular (métrica [60])

- De igual forma, los sistemas P de $\mathcal{TSEC}(1,k), k \geq 1$ sólo pueden resolver eficientemente problemas de la clase **P**, con lo que el mismo resultado hace que podamos ver otra barrera en $\mathcal{TSEC}(k,2)$ entre $k=1$ y $k=2$.

En los sistemas anteriores el alfabeto del entorno, $\mathcal{E}$, no es vacío. En el entorno se puede encontrar un número arbitrario de estos objetos desde la configuración inicial, teniendo por tanto una cantidad de recursos «ilimitados» en cuanto a objetos. Como se ha indicado anteriormente, en ciertos sistemas el uso o no del entorno como elemento *activo* del sistema se traduce en la cantidad de problemas que presuntamente pueden resolver eficientemente. En este caso, dado que, por una parte, la no eficiencia de los sistemas de $\mathcal{TFEC}(k)$ (respectivamente, $\mathcal{TFEC}(k_1,k_2)$), $\mathcal{F} \in \{\mathcal{D},\mathcal{S}\}, k,k_1,k_2 \geq 1$ implica la no eficiencia de los sistemas de $\widehat{\mathcal{TFEC}}(k)$ (respectivamente, $\widehat{\mathcal{TFEC}}(k_1,k_2)$) y que de las soluciones del problema `SAT` pueden ser, mediante las indicaciones de la Sección 8.3, traducidas a los correspondientes marcos *sin entorno*, podemos observar que no existe frontera por el uso del entorno.

En las Figuras 8.1, 8.2, 8.3 y 8.4 se muestra el panorama de esta clase de sistemas P desde el punto de vista de la complejidad computacional.

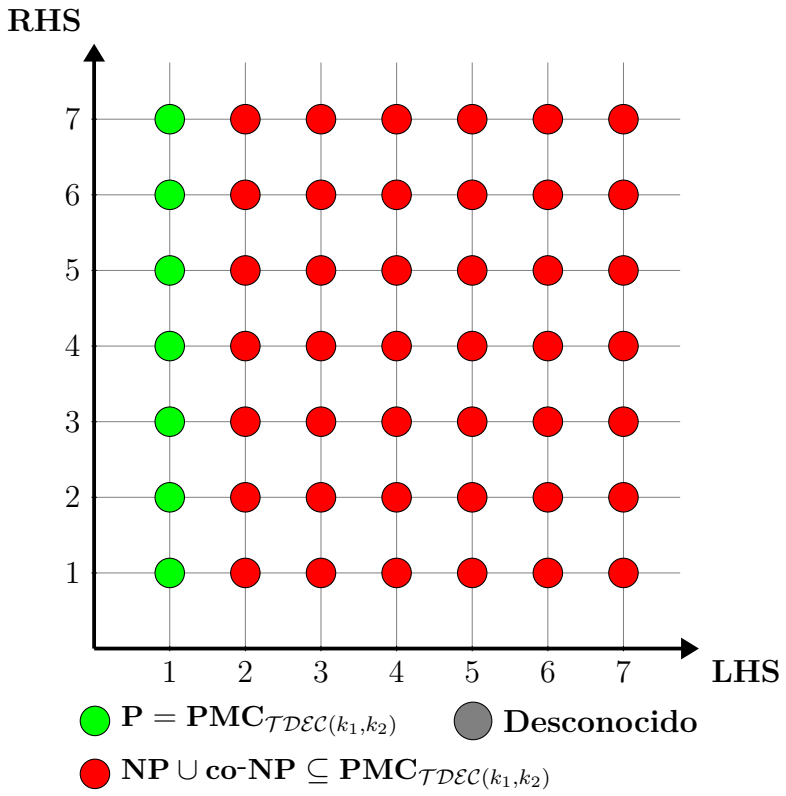


Figura 8.2: Fronteras de la eficiencia en sistemas P de tejidos con reglas de comunicación evolutiva y división celular (métrica [39])

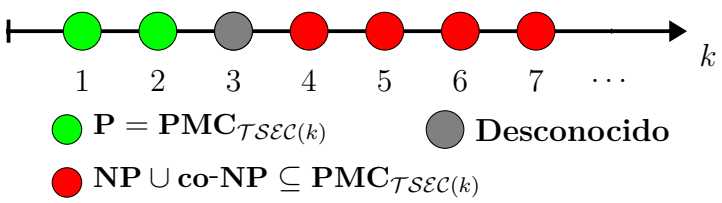


Figura 8.3: Fronteras de la eficiencia en sistemas P de tejidos con reglas de comunicación evolutiva y separación celular (métrica [60])

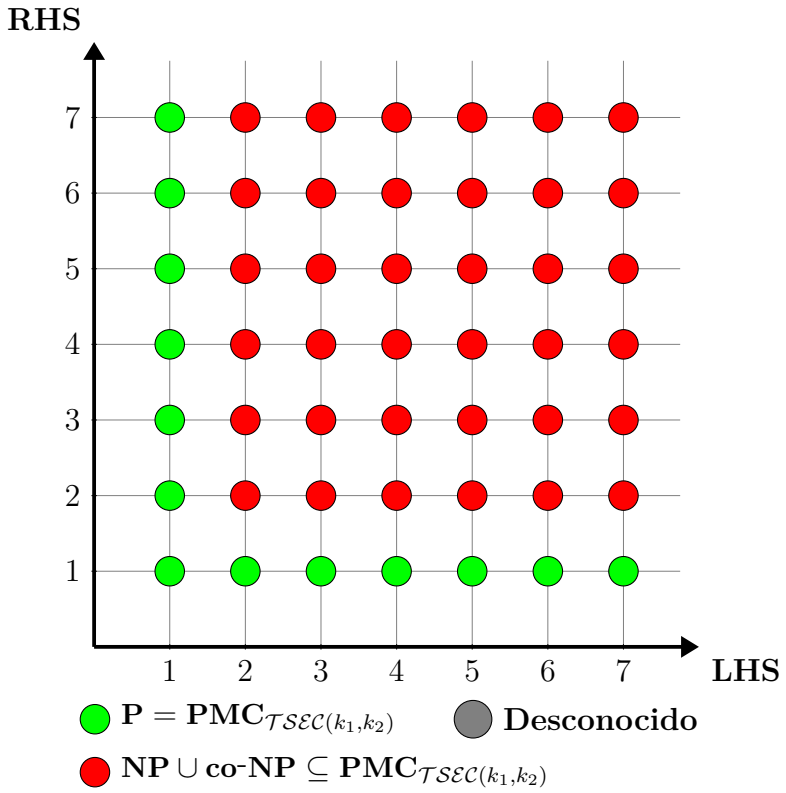


Figura 8.4: Fronteras de la eficiencia en sistemas P de tejidos con reglas de comunicación evolutiva y separación celular (métrica [39])

Capítulo 9

Sistemas P a modo de células con reglas de comunicación evolutiva

En el capítulo anterior se ha analizado, desde el punto de vista de la complejidad computacional, los sistemas de membranas que trabajan a modo de tejidos, con reglas parecidas a las del tipo symport/antiport pero en las que se permite la evolución de los objetos que disparan esas reglas. Específicamente, se han obtenido fronteras entre la no eficiencia y la presumible eficiencia de los sistemas de membranas con las características antes descritas, recordando que cada una de esas fronteras proporciona una herramienta para abordar la resolución del problema **P** *versus* **NP**.

El objetivo de este capítulo consiste en realizar un estudio similar centrado en los sistemas de membranas que trabajan a modo de células. Es conveniente recordar que la estructura de árbol enraizado, subyacente a este tipo de sistemas de membranas, exige imponer una serie de restricciones importantes que afectan tanto a la sintaxis como a la semántica de los modelos de computación objetos de estudio.

9.1. Sistemas P con reglas de comunicación evolutiva

En esta sección se definirá dicho marco, tanto en términos de sintaxis como de semántica, y se presentarán resultados de complejidad computacional que darán una visión de qué clases de complejidad clásicas corresponden a las de

dichos tipos de sistemas de membranas.

Definición 9.1. *Un sistema P con reglas de comunicación evolutiva y reglas de división de grado $q \geq 1$ es una tupla*

$$\Pi = (\Gamma, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{out})$$

donde

- Γ *es un alfabeto finito,* $\mathcal{E} \subseteq \Gamma$ *y* $\mathcal{M}_j, 1 \leq j \leq q$, *son multiconjuntos sobre* Γ*;*
- $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ *de la forma:*
  - *Reglas de comunicación evolutiva de tipo symport hacia dentro:* $[\, u\, [\quad]_j\,]_i \rightarrow [\, [\, u'\,]_j\,]_i$, *siendo* $0 \leq i, j \leq q, i \neq j, u \in M_f^+(\Gamma), u' \in M_f(\Gamma)$*;*
  - *Reglas de comunicación evolutiva de tipo symport hacia fuera:* $[\, [\, u\,]_j\,]_i \rightarrow [\, u'\, [\quad]_j\,]_i$, *siendo* $0 \leq i, j \leq q, i \neq j, u \in M_f^+(\Gamma), u' \in M_f(\Gamma)$*;*
  - *Reglas de comunicación evolutiva de tipo antiport:* $[\, u\, [\, v\,]_j\,]_i \rightarrow [\, v'\, [\, u'\,]_j\,]_i$, *where* $0 \leq i, j \leq q, i \neq j, u, v \in M_f^+(\Gamma)$, $u', v' \in M_f(\Gamma)$*;*
  - *Reglas de división celular:* $[\, a\,]_i \rightarrow [\, b\,]_i\, [\, c\,]_i$, *siendo* $i \in \{1, \ldots, q\}, i \neq i_{out}$ *y* $a, b, c \in \Gamma$*;*
- $i_{out} \in \{0, 1, \ldots, q\}$.

Un sistema P con reglas de comunicación evolutiva y división de grado $q \geq 1$, $\Pi = (\Gamma, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{out})$ puede ser considerado como un conjunto de q membranas etiquetadas por $1, \ldots, q$ organizadas en forma de árbol enraizado definido en μ, cuya raíz es la *membrana piel*, de tal manera que: (a) $\mathcal{E}$ representa el conjunto de objetos colocados inicialmente en el entorno, cada uno de ellos con un número arbitrario (teóricamente infinito) de copias; (b) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ representa los multiconjuntos de objetos colocados inicialmente en las q membranas del sistema; y (c) i_{out} representa una *zona* distinguida que codificará la salida del sistema. Usaremos el término *zona* i ($0 \leq i \leq q$) para referirnos a la membrana i, en el caso $1 \leq i \leq q$, o bien al entorno, en el caso $i_{out} = 0$.

Una *configuración* en un instante t de un tal sistema está descrita por la estructura de membranas del sistema P en el instante t, los multiconjuntos de objetos de Γ colocados en cada membrana en ese instante así como

el multiconjunto de objetos de $\Gamma \setminus \mathcal{E}$ colocados en el entorno, en ese instante. La configuración inicial de $\Pi = (\Gamma, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{out})$ es $(\mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \emptyset)$.

Una regla de comunicación evolutiva de tipo symport hacia dentro $[\, u\, [\quad]_j\,]_i \rightarrow [\, [\, u'\,]_j\,]_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una zona que esté etiquetada por i que tiene, al menos, una membrana hija etiquetada por j y contiene el multiconjunto u. La ejecución de la regla $[\, u\, [\quad]_j\,]_i \rightarrow [\, [\, u'\,]_j\,]_i$ a una tal zona de $\mathcal{C}_t$ produce los siguientes efectos: los objetos del multiconjunto representado por u son consumidos y los objetos del multiconjunto representado por u' son enviados desde la zona i hasta una membrana hija j en $\mathcal{C}_t$. Una regla de comunicación evolutiva de tipo symport hacia fuera $[\, [\, u\,]_j\,]_i \rightarrow [\, u'\, [\quad]_j\,]_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una zona que esté etiquetada por j que tiene, al menos, una membrana padre etiquetada por i y contiene el multiconjunto u. La ejecución de la regla $[\, [\, u\,]_j\,]_i \rightarrow [\, u'\, [\quad]_j\,]_i$ a una tal zona de $\mathcal{C}_t$ produce los siguientes efectos: los objetos del multiconjunto representado por u son consumidos y los objetos del multiconjunto representado por u' son enviados desde la zona j hasta su membrana padre i en $\mathcal{C}_t$. Una regla de comunicación evolutiva de tipo antiport $[\, u\, [\, v\,]_j\,]_i \rightarrow [\, v'\, [\, u'\,]_j\,]_i$ es aplicable a una configuración $\mathcal{C}_t$ en un instante t si en dicha configuración existe, al menos, una zona que esté etiquetada por i que contiene el muticonjunto u y que tiene, al menos, una membrana hija etiquetada por j que contiene el multiconjunto v. La ejecución de la regla $[\, u\, [\quad]_j\,]_i \rightarrow [\, [\, u'\,]_j\,]_i$ a una tal zona de $\mathcal{C}_t$ produce los siguientes efectos: los objetos de los multiconjuntos representados por u y v son consumidos de sus respectivas zonas y los objetos de los multiconjuntos representados por u' y v' son enviados desde la zona i hasta una membrana hija j y desde la zona j hasta la zona i, respectivamente, en $\mathcal{C}_t$.

Por su parte, los conceptos de aplicabilidad de las reglas de división y su ejecución se definen de manera similar al caso de los sistemas P ordinarios.

Al igual que en el marco de los sistemas P de tejidos con reglas de comunicación evolutiva, tendremos dos definiciones de *longitud* (o *tamaño*) de una regla de comunicación evolutiva: por una parte, una posible longitud de una tal regla es $length(r) = |u| + |v| + |u'| + |v'|$; otra posible definición es un par ordenado de números naturales definido como sigue: $length'(r) = (|u| + |v|, |u'| + |v'|)$. Suponemos que si es una regla symport, $|v| = |v'| = 0$.

Se dirá que una configuración $\mathcal{C}_t$ del sistema P reglas de comunicación evolutiva y división celular Π produce una configuración $\mathcal{C}_{t+1}$ en un *paso de transición*, lo denotaremos por $\mathcal{C}_t \Rightarrow_\Pi \mathcal{C}_{t+1}$ y diremos que $\mathcal{C}_{t+1}$ es *una configu-*

ración siguiente de $\mathcal{C}_t$, si se puede pasar de $\mathcal{C}_t$ a $\mathcal{C}_{t+1}$ aplicando las reglas de $\mathcal{R}$ de acuerdo con los siguientes principios:

- A un objeto arbitrario de una membrana cualquiera sólo se le puede aplicar, a lo sumo, una regla (escogida de forma no determinista).
- A cada membrana sólo se le puede aplicar o bien reglas de comunicación o bien una regla de división. En el caso de aplicarse reglas de comunicación a la configuración $\mathcal{C}_t$, éstas se aplicarán de forma *no determinista*, paralela y *maximal*, de acuerdo con lo indicado previamente en los sistemas P básicos. En el caso de aplicarse una regla de división a la configuración $\mathcal{C}_t$, ésta será seleccionada de forma no determinista. Así pues, podemos imaginar que cuando se aplica una regla de división a una membrana de una configuración $\mathcal{C}_t$, la propia membrana queda bloqueada a efectos de poder comunicarse con otras membranas vecinas o con el entorno.

Las nuevas membranas resultantes de la división interactuarán con las otras membranas o con el entorno sólo en el siguiente paso de transición, siempre que no se vuelvan a dividir. Además, esas membranas tendrán las mismas etiquetas que la célula que se divide y, por tanto, proporcionarán nuevos arcos en μ al sistema.

A partir del concepto de paso de transición se define, de manera natural, el concepto de computación del sistema.

Definición 9.2. *Un sistema P con reglas de comunicación evolutiva y reglas de separación de grado $q \geq 1$ es una tupla*

$$\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{out})$$

donde

- Γ *es un alfabeto finito,* $\mathcal{E} \subseteq \Gamma$ *y* $\mathcal{M}_j, 1 \leq j \leq q$*, son multiconjuntos sobre* Γ*;*
- $\{\Gamma_0, \Gamma_1\}$ *es una partición de* Γ*; es decir,* $\Gamma = \Gamma_0 \cup \Gamma_1, \Gamma_0, \Gamma_1 \neq \emptyset, \Gamma_0 \cap \Gamma_1 = \emptyset$*;*
- $\mathcal{R}$ *es un conjunto finito de reglas sobre* Γ *de la forma:*
  - *Reglas de comunicación evolutiva de tipo symport hacia dentro:* $[\, u\, [\quad]_j\,]_i \rightarrow [\, [\, u'\,]_j\,]_i$*, siendo* $0 \leq i, j \leq q, i \neq j, u \in M_f^+(\Gamma), u' \in M_f(\Gamma)$*;*

- *Reglas de comunicación evolutiva de tipo symport hacia fuera:* $[\,[\,u\,]_j\,]_i \to [\,u'\,[\quad]_j\,]_i$, *siendo* $0 \leq i, j \leq q, i \neq j, u \in M_f^+(\Gamma), u' \in M_f(\Gamma)$;
- *Reglas de comunicación evolutiva de tipo antiport:* $[\,u\,[\,v\,]_j\,]_i \to [\,v'\,[\,u'\,]_j\,]_i$, *where* $0 \leq i, j \leq q, i \neq j, u, v \in M_f^+(\Gamma)$, $u', v' \in M_f(\Gamma)$;
- *Reglas de separación celular:* $[\,a\,]_i \to [\,\Gamma_0\,]_i\,[\,\Gamma_1\,]_i$, *siendo* $i \in \{1, \ldots, q\}$, $i \neq i_{out}$ *y* $a \in \Gamma$;

- $i_{out} \in \{0, 1, \ldots, q\}$.

Los conceptos de aplicabilidad de las reglas de comunicación evolutiva de tipo symport y antiport, así como de su ejecución, han sido definidos previamente. Por su parte, los conceptos de aplicabilidad de las reglas de separación y su ejecución se definen de manera similar al caso de los sistemas P ordinarios.

Con respecto a la semántica de esta variante, las reglas serán aplicadas de forma no determinista y maximal con la siguiente restricción: cuando una membrana etiquetada por i es separada, entonces la regla de separación será la única regla de $\mathcal{R}_i$ que se aplicará a esa membrana en ese paso de computación. Las nuevas membranas resultantes de la separación interactuarán con otras las otras membranas o con el entorno sólo a partir del siguiente paso de transición, siempre y cuando no se separen. Además, esas membranas tendrán las mismas etiquetas que la membrana que se separa y, por tanto, proporcionarán nuevos elementos en μ.

En primer lugar, las mismas relaciones que en los sistemas P de tejidos con comunicación evolutiva pueden ser observadas. Si $\mathcal{X} \in \{\mathcal{D}, \mathcal{S}\}$, entonces se tiene que $\mathcal{CXC} \subseteq \mathcal{CXEC}$ ya que, por una parte, toda regla symport hacia dentro $(u, in) \in \mathcal{R}_i$ (respectivamente, hacia fuera $(u, out) \in \mathcal{R}_i$) puede ser considerada como una regla de comunicación evolutiva symport hacia dentro del tipo $[\,u\,[\quad]_j\,]_i \to [\,[\,u\,]_j\,]_i$ (respectivamente, hacia fuera del tipo $[\,[\,u\,]_j\,]_i \to [\,u\,[\quad]_j\,]_i$); y, por otra, toda regla antiport $(u, out; v, in) \in \mathcal{R}_i$ puede ser considerada como una regla de comunicación evolutiva antiport del tipo $[\,u\,[\,v\,]_j\,]_i \to [\,v\,[\,u\,]_j\,]_i$.

Además, se verifican las relaciones siguientes:

- $\mathbf{PMC}_{\mathcal{CXC}(k)} \subseteq \mathbf{PMC}_{\mathcal{CXEC}(k,k)} \subseteq \mathbf{PMC}_{\mathcal{CXEC}(2k)}$.
- $\mathbf{PMC}_{\mathcal{CXEC}(k_1,k_2)} \subseteq \mathbf{PMC}_{\mathcal{CXEC}(k_1+k_2)}$.
- Si $k \geq 2$ entonces $\mathbf{PMC}_{\mathcal{CXEC}(k)} \subseteq \mathbf{PMC}_{\mathcal{CXEC}(k,k-1)}$.

9.1.1. No eficiencia de los sistemas P con comunicación evolutiva

El Teorema 7 de [39] declara que, usando la técnica del grafo de dependencias se puede comprobar la no eficiencia de los sistemas P de tejidos con comunicación evolutiva y separación cuando la parte izquierda de las reglas symport/antiport están restringidas a longitud 1, dado que son sistemas *no cooperativos*. Por supuesto, el mismo argumento puede ser usado para sus análogos con división en lugar de separación. Aquí, daremos una demostración exhaustiva del resultado en sistemas P a modo de células. Además, en el mismo artículo se demuestra mediante la técnica algorítmica que los sistemas de $\mathcal{TSEC}(k,1), k \geq 1$ tampoco pueden resolver problemas presuntamente intratables en tiempo polinomial. También, basándonos en dicha demostración podremos demostrar que los sistemas P a modo de células también tienen dicha restricción.

9.1.1.1. No eficiencia de los sistemas no cooperativos

Sea $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ un sistema P reconocedor de $\mathcal{CDEC}(1,k)$ o de $\mathcal{CSEC}(1,k)$, siendo $k \geq 1$. Denotamos por $\mathcal{M}_j^*$ el multiconjunto sobre $\Gamma \times \{j\}$ obtenido de $\mathcal{M}_j$ reemplazando $a \in \Gamma$ por (a,j), y para cada multiconjunto finito w sobre Σ por (a, i_{in}).

Las reglas de $\mathcal{R}_1 \cup \cdots \cup \mathcal{R}_q$ son de las siguientes formas: $[\,[\,a\,]_i\,]_j \to [\,u'\,[\quad]_i\,]_j$, $[\,a\,[\quad]_i\,]_j \to [\,[\,u'\,]_i\,]_j$, $[\,a\,]_h \to [\,b\,]_h\,[\,c\,]_h$ and $[\,a\,]_h \to [\,\Gamma_0\,]_h\,[\,\Gamma_1\,]_h$. Estas reglas pueden ser consideradas, en cierta manera, como una *dependencia* entre el objeto que dispara la regla y los objetos producidos por su aplicación.

- Las reglas en $\mathcal{R}_i$ del tipo $[\,[\,a\,]_i\,]_j \to [\,u'\,[\quad]_i\,]_j$ pueden ser descritas como el par (a,i) que produce los pares (b,j), para $b \in u'$.
- Las reglas en $\mathcal{R}_i$ del tipo $[\,a\,[\quad]_i\,]_j \to [\,[\,u'\,]_i\,]_j$ pueden ser descritas como el par (a,j) que produce los pares (b,i), para $b \in u'$.
- Las reglas en $\mathcal{R}_i$ del tipo $[\,a\,]_i \to [\,b\,]_i\,[\,c\,]_i$ pueden ser descritas como el par (a,i) que produce los pares (b,i) and (c,i).
- Las reglas en $\mathcal{R}_i$ del tipo $[\,a\,]_i \to [\,\Gamma_0\,]_i\,[\,\Gamma_1\,]_i$ pueden ser descritas como el par (a,i) que no produce ningún nuevo par.

Formalizamos estas ideas en la siguiente definición.

Definición 9.3. *Sea* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ *un sistema P reconocedor de* $\mathcal{CDEC}(1,k)$ *o* $\mathcal{CSEC}(1,k)$*. El grafo de dependencias asociado a* Π *es el grafo dirigido* $G_\Pi = (V_\Pi, E_\Pi)$ *definido como sigue:*

- *El conjunto de vértices es* $V_\Pi = \{s\} \cup VL_\Pi \cup VR_\Pi$*, donde:*

$$\begin{aligned} VL_\Pi &= \{(a,i) \in \Gamma \times \{1,\ldots,q\} \mid [[\,[\,a\,]_i\,]_j \to [\,u'\,[\quad]_i\,]_j \in \mathcal{R}_i] \vee \\ &\quad [\exists j \in ch(i)([\,a\,[\quad]_i\,]_j \to [\,[\,u'\,]_i\,]_j \in \mathcal{R}_j] \vee \\ &\quad [\exists b,c \in \Gamma([\,a\,]_i \to [\,b\,]_i\,[\,c\,]_i \in \mathcal{R}_i)] \vee \\ &\quad [([\,a\,]_i \to [\,\Gamma_0\,]_i\,[\,\Gamma_1\,]_i \in \mathcal{R}_i)]\}; \\ VR_\Pi &= \{(b,i) \in \Gamma \times \{1,\ldots,q\} \mid [\,a\,[\quad]_i\,]_j \to [\,[\,u'\,]_i\,]_j \in \mathcal{R}_i, \\ &\quad b \in u'] \vee [\exists j \in ch(i)([\,[\,a\,]_i\,]_j \to [\,u'\,[\quad]_i\,]_j \in \mathcal{R}_j, b \in u'] \vee \\ &\quad [\exists a,c \in \Gamma([\,a\,]_i \to [\,b\,]_i\,[\,c\,]_i \in \mathcal{R}_i)]. \end{aligned}$$

- *El conjunto de aristas es:*

$$\begin{aligned} E_\Pi &= \{(s,(a,j)) \mid 1 \le j \le q \wedge (a,j) \in \mathcal{M}_j^*\} \cup \\ &\quad \{((a,i),(b,j)) \in V_\Pi \times V_\Pi \mid [[\,[\,a\,]_i\,]_j \to [\,u'\,[\quad]_i\,]_j \in \mathcal{R}_i, \\ &\quad j = p(i), b \in u'] \vee [\exists j \in ch(i)[\,a\,[\quad]_i\,]_j \to [\,[\,u'\,]_i\,]_j \in \mathcal{R}_j, \\ &\quad b \in u'] \vee [i = j] \wedge [\exists c \in \Gamma([\,a\,]_i \to [\,b\,]_i\,[\,c\,]_i] \in \mathcal{R}_i\}. \end{aligned}$$

A continuación, se muestra como el grafo de dependencia asociado al sistema P de $\mathcal{CDEC}(1,k)$ o de $\mathcal{CSEC}(1,k)$, puede ser construido por una máquina de Turing en tiempo polinomial

Proposición 9.1. *Sea* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ *un sistema P reconocedor de* $\mathcal{CFEC}(1,k)$*, siendo* $\mathcal{F} \in \{\mathcal{D}, \mathcal{S}\}$ *y* $k \ge 1$*. Existe una máquina de Turing que construye el grafo de dependencias,* G_Π*, asociado con* Π*, en tiempo polinomial (es decir, en un tiempo acotado por una función polinómica que depende del número total de reglas y la longitud máxima de las reglas).*

Demostración. Un algoritmo determinista que, dado un sistema P reconocedor Π de $\mathcal{CFEC}(1,k)$, cuyo conjunto de reglas es $\mathcal{R} = \mathcal{R}_1 \cup \cdots \cup \mathcal{R}_q$, construye el siguiente grafo de dependencias, es el Algoritmo 9.1.1.

El tiempo de ejecución de este algoritmo está acotado por $O(|R|) \subset O(q \cdot |\Gamma|^3)$.

□

Proposición 9.2. *Sea* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ *un sistema P reconocedor confluente de* $\mathcal{CFEC}(1,k)$*. Las siguientes afirmaciones son equivalentes:*

Algoritmo 9.1.1 Algoritmo $\mathcal{A}$

Entrada: El sistema P reconocedor Π

$V_\Pi \leftarrow \{s\}$; $E_\Pi \leftarrow \emptyset$
para $j = 1$ hasta q **hacer**
  para cada par $(a, j) \in \mathcal{M}_j^*$ **hacer**
    $E_\Pi \leftarrow E_\Pi \cup \{(s, (a, j))\}$
  fin para
fin para
para cada $r \in \mathcal{R}$ de Π **hacer**
  si $r = [\, a\, [\quad]_i\,]_j \rightarrow [\, [\, u'\,]_i\,]_j \in \mathcal{R}_i$ **entonces**
    $V_\Pi \leftarrow V_\Pi \cup \{(a, j), (b, i) \mid b \in u'\}$;
    $E_\Pi \leftarrow E_\Pi \cup \{((a, j), (b, i)) \mid b \in u'\}$
  fin si
  si $r = [\, [\, a\,]_i\,]_j \rightarrow [\, u'\, [\quad]_i\,]_j \in \mathcal{R}_i$ **entonces**
    $V_\Pi \leftarrow V_\Pi \cup \{(a, i), (b, j) \mid b \in u'\}$
    $E_\Pi \leftarrow E_\Pi \cup \{((a, i), (b, j)) \mid b \in u'\}$
  fin si
  si $r = [\, a\,]_i \rightarrow [\, b\,]_i\, [\, c\,]_i \in \mathcal{R}_i$ **entonces**
    $V_\Pi \leftarrow V_\Pi \cup \{(a, i), (b, i), (c, i)\}$
    $E_\Pi \leftarrow E_\Pi \cup \{((a, i)), (b, i))\} \cup \{((a, i), (c, i))\}$
  fin si
  si $r = [\, a\,]_i \rightarrow [\, \Gamma_0\,]_i\, [\, \Gamma_1\,]_i \in \mathcal{R}_i$ **entonces**
    $V_\Pi \leftarrow V_\Pi \cup \{(a, i)\}$
  fin si
fin para

(1) *Existe una computación de aceptación de* Π.

(2) *Existe un camino (con longitud mayor que o igual a 2) desde* s *hasta* $(\texttt{yes}, 0)$ *en el grafo de dependencias asociado a* Π.

Demostración. (1) $\Rightarrow$ (2). Primero, demostramos que por cada computación de aceptación $\mathcal{C}$ de Π existe un camino desde s hasta $(\texttt{yes}, 0)$ en el grafo de dependencias asociado a Π. Por inducción sobre la longitud n de $\mathcal{C}$.

Sea $n = 1$ y $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1)$ una computación de aceptación de Π con longitud 1. Entonces una regla de la forma $[\,[\,a\,]_{i_{skin}}\,]_{env} \rightarrow [\,u'\,[\quad]_{i_{skin}}\,]_{env} \in \mathcal{R}_{i_{skin}}$, donde $a \in \Gamma$ y $\texttt{yes} \in u'$, ha sido aplicada a la configuración inicial $\mathcal{C}_0$. Entonces, $a \in \mathcal{C}_0(i_{skin})$, así que $(a, i_{skin}) \in \mathcal{M}^*_{i_{skin}}$. Por tanto, $(s, (a, i_{skin}), (\texttt{yes}, 0))$ es un camino desde s hasta $(\texttt{yes}, 0)$ en el grafo de dependencias asociado a Π.

Supongamos que el resultado es válido para n. Sea $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_n, \mathcal{C}_{n+1})$ una computación de aceptación de Π de longitud $n + 1$. En esta situación, $\mathcal{C}' = (\mathcal{C}_1, \ldots, \mathcal{C}_n, \mathcal{C}_{n+1})$ es una computación de aceptación del sistema $\Pi' = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mu, \mathcal{M}'_1, \ldots, \mathcal{M}'_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$, siendo $\mathcal{M}'_j = \{a \mid a \in \mathcal{C}_1(j)\}$ el «contenido» de la membrana j en la configuración $\mathcal{C}_1$, para cada $1 \leq j \leq q$. Por hipótesis de induccción existe un camino $\gamma_{\mathcal{C}'} = (s, (b_1, i_1), \ldots, (\texttt{yes}, 0))$ desde s to $(\texttt{yes}, 0)$ en el grafo de dependencias asociado a Π' (con longitud mayor que o igual a 2). Distinguimos dos casos. Si $b_1 \in \mathcal{C}_0(i_1)$ (es decir, si $b_1 \in \mathcal{M}_{i_1}$), entonces una regla de división del tipo $[\,b_0\,]_{i_1} \rightarrow [\,b_2\,]_{i_1}\,[\,b_3\,]_{i_1}$, donde $b_0 \neq b_1$, $b_2 \neq b_1$ y $b_3 \neq b_1$, ha sido aplicada a la membrana i_1. Entonces, $\gamma_{\mathcal{C}} = (s, (b_1, i_1), (b_1, i_1), \ldots, (\texttt{yes}, 0))$ es un camino desde s hasta $(\texttt{yes}, 0)$ en el grafo de dependencias asociado a Π, y el resultado es válido. De otra forma, hay tres opciones:

- $\exists b_0 \in \Gamma, \exists j \in ch(i_1) \mid [\,[\,b_0\,]_j\,]_{i_1} \rightarrow [\,u'\,[\quad]_j\,]_{i_1} \in \mathcal{R}_j, b_1 \in u'$;
- $\exists b_0 \in \Gamma \mid [\,b_0\,[\quad]_{i_1}\,]_j \rightarrow [\,[\,u'\,]_{i_1}\,]_j \in \mathcal{R}_{i_1}, b_1 \in u'$; o
- $\exists b_0, b_2 \in \Gamma \mid [\,b_0\,]_{i_1} \rightarrow [\,b_1\,]_{i_1}\,[\,b_2\,]_{i_1}$.

En cualquier caso, esto lleva a la producción de (b_1, i_1) en el primer paso de la computación $\mathcal{C}$. Por tanto, $\gamma_{\mathcal{C}} = (s, (b_0, i_0), (b_1, i_1), \ldots, (\texttt{yes}, 0))$ es un camino desde s hasta $(\texttt{yes}, 0)$ en el grafo de dependencias asociado a Π.

(2) $\Rightarrow$ (1). Veamos que para cada camino desde s hasta $(\texttt{yes}, 0)$ en el grafo de dependencias asociado a Π, con longitud $k \geq 2$, existe una computación de aceptación de Π. Por inducción sobre la longitud k del camino.

Sea $k = 2$ y $\gamma_{\mathcal{C}} = (s, (a_0, i_0), (\texttt{yes}, 0))$. Entonces, $i_0 = i_{skin}$ es la etiqueta de la membrana piel, $[\,[\,a_0\,]_{i_{skin}}\,]_{env} \rightarrow [\,u'\,[\quad]_{i_{skin}}\,]_{env} \in \mathcal{R}_{i_{skin}}$, donde $a_0 \in \Gamma$

y $\texttt{yes} \in u'$, y la computación $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1)$ donde la regla $(a_0, out; u') \in \mathcal{R}_{i_{skin}}$ pertenece al conjunto de reglas que da lugar a la configuración $\mathcal{C}_1$ desde la configuración $\mathcal{C}_0$, es una computación de aceptación de Π.

Supongamos que el resultado es válido para $k \geq 2$. Sea entonces $(s, (a_0, i_0), (a_1, i_1), \ldots, (a_{k-1}, i_{k-1}), (\texttt{yes}, 0))$ un camino desde s hasta $(\texttt{yes}, 0)$ en el grafo de dependencias de longitud $k+1$. Si $(a_0, i_0) = (a_1, i_1)$, entonces el resultado valida la hipótesis de inducción. En cualquier otro caso, sea $\mathcal{C}_1$ una configuración de Π obtenida de $\mathcal{C}_0$ por la aplicación de un conjunto de reglas que contienen a la regla que da lugar a (a_1, i_0) desde (a_0, i_0). Entonces $(s, (a_1, i_1), \ldots, (a_{k-1}, i_{k-1}), (\texttt{yes}, 0))$ es un camion desde s hasta $(\texttt{yes}, 0)$ de longitud k, en el grafo de dependencias asociado al sistema $\Pi' = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mu, \mathcal{M}'_1, \ldots, \mathcal{M}'_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$, donde $\mathcal{M}'_j = \{a \mid a \in \mathcal{C}_1(j)\}$ es el contenido de la membrana j en la configuración $\mathcal{C}_1$, para cada $1 \leq j \leq q$. Por hipótesis de inducción, existe una computación de acpetación $\mathcal{C}' = (\mathcal{C}_1, \ldots, \mathcal{C}_k)$ de Π'. Por lo tanto $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_k)$ es una computación de aceptación de Π.

□

Corolario 9.1. *Sea $X = (I_X, \theta_X)$ un problema de decisión. Sea $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ una familia de sistemas P reconocedores de $\mathcal{CFEC}(1,k)$ que resuelven X, de acuerdo con la definición dada en la Sección 4.4. Sea (cod, s) una codificación polinomial asoociada a esa solución. Entonces, para cada instancia w del problema X las siguientes afirmaciones son equivalentes following assertions are equivalent:*

1. *$\theta_X(w) = 1$ (es decir, la respuesta al problema es* yes *para w).*
2. *Existe un camino desde s hasta $(\texttt{yes}, 0)$ en el grafo de dependencias asociado al sistema $\Pi(s(w))$ con multiconjunto de entrada $cod(w)$.*

Demostración. Sea $w \in I_X$. Entonces, $\theta_X(w) = 1$ si y sólo si existe una computación de aceptación del sistema $\Pi(s(w)) + cod(w)$. Teniendo en cuenta que $\Pi(s(w)) + cod(w)$ es un sistema *confluente*, por la Proposición 9.2 deducimos que $\theta_X(w) = 1$ si y sólo si existe un camino desde s hasta $(\texttt{yes}, 0)$ en el grafo de dependencias asociado al sistema $\Pi(s(w)) + cod(w)$.

□

Teorema 9.1. $\mathbf{P} = \mathbf{PMC}_{\mathcal{CDEC}(1,k)} = \mathbf{PMC}_{\mathcal{CSEC}(1,k)}$, *para $k \geq 1$.*

Demostración. Tenemos que $\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{CDEC}(1,k)} \cap \mathbf{PMC}_{\mathcal{CSEC}(1,k)}$ con $k \geq 1$ dado que $\mathbf{PMC}_{\mathcal{CDEC}(1,k)}$ y $\mathbf{PMC}_{\mathcal{CSEC}(1,k)}$ son clases no vacías cerradas bajo reducibilidad en tiempo polinomial. A continuación, se muestra que $\mathbf{PMC}_{\mathcal{CDEC}(1,k)} \cup$

$\mathbf{PMC}_{\mathcal{CSEC}(1,k)} \subseteq \mathbf{P}$. Para ello, sea $X \in \mathbf{PMC}_{\mathcal{CDEC}(1,k)} \cap \mathbf{PMC}_{\mathcal{CSEC}(1,k)}$ y $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ una familia de sistemas P reconocedores de $\mathcal{CDEC}(1,k)$ o $\mathcal{CSEC}(1,k)$ que resuelvan X, de acuerdo con la definición dada en la Sección 4.4. Sea (cod, s) la codificación polinomial asociada a dicha solución. Consideramos entonces el algoritmo 9.1.2

Algoritmo 9.1.2 Algoritmo $\mathcal{A}$

Entrada: Una instancia w de X

Construir el sistema $\Pi(s(w)) + cod(w)$

Construir el grafo de dependencias $G_{\Pi(s(w))+cod(w)}$ asociado a $\Pi(s(w)) + cod(w)$

$\texttt{Reachability}(G_{\Pi(s(w))+cod(w)}, s, (\texttt{yes}, 0)$

Obviamente, este algoritmo es polinomial respecto al tamaño $|w|$ de la entrada. □

9.1.1.2. No eficiencia de los sistemas con separación y con longitud de la parte derecha acotada a 1

Sea $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ un sistema P reconocedor de $\mathcal{CSEC}(k,1)$, siendo $k \geq 1$. Denotamos como $\mathcal{R}_C$ (respectivamente, $\mathcal{R}_S$) el conjunto de reglas de comunicación (respectivamente, reglas separación) de Π. Fijaremos un orden total en $\mathcal{R}_C$ y un orden total en $\mathcal{R}_S$. Ya que varias membranas con la misma etiqueta serán generadas mediante el uso de reglas de separación, para identificar las diferentes membranas con la misma etiqueta, se usará la siguiente definición recursiva para modificar las etiquetas de las membranas que se vayan generando:

- Denotamos la etiqueta de una membrana como un par (i, σ), donde $1 \leq i \leq q$ y $\sigma \in \{0,1\}^*$ es una cadena binaria.
- Si se aplica una regla de separación a una membrana etiquetada con (i, σ), entonces las nuevas membranas creadas serán etiquetadas con $(i, \sigma 0)$ y $(i, \sigma 1)$, respectivamente. Se considera el orden lexicográfico sobre el conjunto de etiquetas de las membranas durante toda la computación del sistema.

Nótese que si se aplica una regla de comunicación entre dos membranas, las etiquetas no cambian.

Una configuración en un instante t de un sistema P de $\mathcal{CSEC}(k,1)$ es descrito por los multiconjuntos de objetos sobre Γ contenidos en cada membrana y el

multiconjunto de objetos sobre $\Gamma \setminus \mathcal{E}$ en el entorno. Por tanto, una configuración de Π puede ser descrita como sigue:

$$\{(a, i, \sigma) \mid a \in \Gamma \cup \{\lambda\}, 1 \leq i \leq q, \sigma \in \{0,1\}^*\} \cup \{(a, 0) \mid a \in \Gamma \setminus \mathcal{E}\}$$

Usamos LHS y RHS para referirnos a la parte izquierda y a la parte derecha de una regla. Son definidas de manera natural de acuerdo a la definición de una regla:

- $r \equiv [\,[\,u\,]_i\,]_j \to [\,b\,[\quad]_i\,]_j \in \mathcal{R}_i$, siendo u el multiconjunto $\{a_1, \ldots, a_k\} \in M_f(\Gamma)$ y $b \in \Gamma \cup \{\lambda\}$. Entonces, denotamos con $n \cdot LHS(r, (i, \sigma_i)) = (a_1, i, \sigma_i)^n \ldots (a_k, i, \sigma_i)^n$, para $a_l \in \Gamma$ $(1 \leq l \leq k)$, y por $n \cdot RHS(r, (i, \sigma_i)) = (b, j, \sigma_j)^n$.

- $r \equiv [\,u\,[\quad]_j\,]_i \to [\,[\,b\,]_j\,]_i \in \mathcal{R}_i$, siendo u el multiconjunto $\{a_1, \ldots, a_k\} \in M_f(\Gamma)$, $b \in \Gamma \cup \{\lambda\}$ y $i \neq i_{skin}$ (respectivamente, $i = i_{skin}$). Entonces, denotamos con $n \cdot LHS(r, (i, \sigma_i)) = (a_1, j, \sigma_j)^n \ldots (a_k, j, \sigma_j)^n$, con $a_l \in \Gamma$ (respectivamente, $a_l \in \Gamma \setminus \mathcal{E}$) $(1 \leq l \leq k)$ y con $n \cdot RHS(r, (i, \sigma_i)) = (b, i, \sigma_i)^n$.

- $r \equiv [\,v\,[\,u\,]_j\,]_i \to [\,[\,c\,]_j\,]_i \in \mathcal{R}_i$, siendo u el multiconjunto $\{a_1, \ldots, a_{k_1}\} \in M_f(\Gamma)$, v el multiconjunto $\{b_1, \ldots, b_{k_2}\} \in M_f(\Gamma)$, $c \in \Gamma \cup \{\lambda\}$ y $i \neq i_{skin}$ (respectivamente, $i = i_{skin}$). Entonces, denotamos con $n \cdot LHS(r, (i, \sigma_i)) = (a_1, i, \sigma_i)^n \ldots (a_{k_1}, i, \sigma_i)^n (b_1, j, \sigma_j)^n \ldots (b_{k_2}, j, \sigma_j)^n$ con $a_{l_1}, b_{l_2} \in \Gamma$ (respectivamente, $a_{l_1} \in \Gamma$ and $b_{l_2} \in \Gamma \setminus \mathcal{E}$) $(1 \leq l_1 \leq k_1, 1 \leq l_2 \leq k_2)$ y con $n \cdot RHS(r, (i, \sigma_i)) = (c, i, \sigma_i)^n$.

- $r \equiv [\,v\,[\,u\,]_j\,]_i \to [\,c\,[\quad]_j\,]_i \in \mathcal{R}_i$, siendo u el multiconjunto $\{a_1, \ldots, a_{k_1}\} \in M_f(\Gamma)$, v $\{b_1, \ldots, b_{k_2}\} \in M_f(\Gamma)$, $c \in \Gamma \cup \{\lambda\}$ y $i \neq i_{skin}$ (respectivamente, $i = i_{skin}$). Entonces, denotamos con $n \cdot LHS(r, (i, \sigma_i)) = (a_1, i, \sigma_i)^n \ldots (a_{k_1}, i, \sigma_i)^n (b_1, j, \sigma_j)^n \ldots (b_{k_2}, j, \sigma_j)^n$ con $a_{l_1}, b_{l_2} \in \Gamma$ (respectivamente, $a_{l_1} \in \Gamma$ y $b_{l_2} \in \Gamma \setminus \mathcal{E}$) $(1 \leq l_1 \leq k_1, 1 \leq l_2 \leq k_2)$ y con $n \cdot RHS(r, (i, \sigma_i)) = (c, j, \sigma_j)^n$.

Hay que remarcar que los objetos del tipo (λ, i, σ_i) pueden ser omitidos, dado que significa que la RHS está vacía.

Si $\mathcal{C}_t$ es una configuración de Π, entonces el multiconjunto obtenido al reemplazar en $\mathcal{C}_t$ todas las apariciones de (x, i, σ) por (x, i, σ') es denotado por $\mathcal{C}_t + \{(x, i, \sigma)/\sigma'\}$. Además, denotamos con $\mathcal{C}_t + m$ (respectivamente, $\mathcal{C}_t \setminus$

m) un multiconjunto m de objetos etiquetados añadidos a (respectivamente, eliminados de) la configuración $\mathcal{C}_t$.

A continuación, demostraremos que los sistemas P de $\mathcal{CSEC}(k,1)$ pueden resolver sólo problemas tratables.

Si $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_n)$ es una computación de parada, denotamos con $|\mathcal{C}| = n$ la longitud de $\mathcal{C}$. Para cada i $(1 \leq i \leq q)$, el multiconjunto de objetos sobre Γ contenido en todas las membranas etiquetadas por i en la configuración $\mathcal{C}_t$ es denotado por $\mathcal{C}_t(i)$. Denotamos por $\mathcal{C}_t(0)$ el multiconjunto de objetos sobre $\Gamma \backslash \mathcal{E}$ contenido en el entorno en la configuración $\mathcal{C}_t$. Finalmente, el multiconjunto finito $\mathcal{C}_t(0) + \mathcal{C}_t(1) + \cdots + \mathcal{C}_t(q)$ es denotado por $\mathcal{C}_t^*$.

Lema 9.1. *Sea* $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ *un sistema P reconocedor de* $\mathcal{CSEC}(k,1)$*, siendo* $k \geq 1$*. Sea* $M = |\mathcal{M}_1 + \cdots + \mathcal{M}_q|$ *y* $\mathcal{C} = (\mathcal{C}_0, \ldots, \mathcal{C}_n)$ *una computación de* Π*. Entonces, tenemos*

1. $|\mathcal{C}_0^*| = M$*, y para cada* $t, 0 \leq t \leq n-1, |\mathcal{C}_{t+1}^*| \leq |\mathcal{C}_t^*|$*;*
2. *para cada* $t, 0 \leq t \leq n, |\mathcal{C}_t^*| \leq M$*; y*
3. *el número de membranas creadas a lo largo de la computación* $\mathcal{C}$ *por la aplicación de reglas de separación está acotada por* $2M$*.*

Demostración. (1) Observemos que $|\mathcal{C}_0^*| = |\mathcal{C}_0(0) + \mathcal{C}_0(1) + \cdots + \mathcal{C}_0(q)| = |\mathcal{M}_1 + \cdots + \mathcal{M}_q| = M$. Sea Π un sistema P reconocedor de $\mathcal{CSEC}(k,1)$ y $\mathcal{R}_1, \ldots, \mathcal{R}_q$ los conjuntos de reglas asociados a Π, que contienen los siguientes tipos de reglas de comunicación evolutiva:

- $[\,[\,u\,]_i\,]_j \to [\,a\,[\quad]_i\,]_j \in \mathcal{R}_i$, $1 \leq i \leq q$, $u \in M_f(\Gamma)$ y $a \in \Gamma \cup \{\lambda\}$.
- $[\,u\,[\quad]_j\,]_i \to [\,[\,a\,]_j\,]_i \in \mathcal{R}_i$, $i \neq i_{skin}$ (respectivamente, $i = i_{skin}$), $u \in M_f(\Gamma)$ (respectivamente, $u \cap \Gamma \setminus \mathcal{E} \neq \emptyset$) y $a \in \Gamma \cup \{\lambda\}$.
- $[\,u\,[\,v\,]_j\,]_i \to [\,[\,a\,]_j\,]_i$, $i \neq i_{skin}$ (respectivamente, $i = i_{skin}$), $u, v \in M_f(\Gamma)$ (respectivamente, $u \in M_f(\Gamma)$, $v \cap \Gamma \setminus \mathcal{E} \neq \emptyset$) y $a \in \Gamma \cup \{\lambda\}$.
- $[\,u\,[\,v\,]_j\,]_i \to [\,a\,[\quad]_j\,]_i$, $i \neq i_{skin}$ (respectivamente, $i = i_{skin}$), $u, v \in M_f(\Gamma)$ (respectivamente, $u \in M_f(\Gamma)$, $v \cap \Gamma \setminus \mathcal{E} \neq \emptyset$) y $a \in \Gamma \cup \{\lambda\}$.

Para cada $t, 0 \leq t \leq n-1$, en la transición desde la configuración $\mathcal{C}_t$ a la configuración $\mathcal{C}_{t+1}$, mediante el uso de cualquier regla, al menos un objeto de $\mathcal{C}_t$ es consumido y a lo sumo un objeto es producido en $\mathcal{C}_{t+1}$. Por lo tanto, en cualquier paso de transición el número de objetos no es incrementado.

(2) Por inducción sobre t. Empecemos analizando el caso base $t = 0$. El resultado es trivial dado que $|\mathcal{C}_0^*| = M$. Por hipótesis de inducción, supongamos que el resultado es válido para $t, 0 \leq t \leq n-1$. Entonces, $|\mathcal{C}_{t+1}^*| \geq |\mathcal{C}_t^*|$, que es verdadero debido a (1), y por hipótesis de inducción sabemos que $|\mathcal{C}_t^*| \leq M$. Por tanto, el resultado también es verdadero para $t+1$.

(3) De acuerdo al hecho que la aplicación de una regla de separación consume un objeto y produce dos nuevas membranas, el resultado (3) puede ser obtenido de (2) fácilmente, dado que el número máximo de reglas de separación que pueden ser ejecutadas a lo largo de la computación en este tipo de sistemas viene acotada por la cantidad de objetos en los multiconjuntos iniciales.

□

El Algoritmo 9.1.3 $\mathcal{A}$ es un algoritmo determinista acotado polinomialmente respecto al tiempo. Dicho algoritmo recibe como entrada un sistema P Π de $\mathcal{CSEC}(k,1)$ y un multiconjunto m de Π, de tal manera que el algoritmo $\mathcal{A}$ reproduce el comportamiento de **una** computación de $\Pi + m$. Si el sistema Π es confluente, entonces el algoritmo $\mathcal{A}$ devolverá la misma respuesta que Π.

Algoritmo 9.1.3 Algoritmo $\mathcal{A}$

Entrada: Un sistema P Π de $\mathcal{CSEC}(2,1)$ y un multiconjunto de entrada m
Salida: *Sí* si $\Pi + m$ tiene una computación de aceptación, *No* en caso contrario

```
Fase de inicialización: C_0 es la configuración inicial de Π + m
t ← 0
mientras C_t no es una configuración de parada hacer
    Fase de selección: Entrada C_t, salida (C'_t, A)
    Fase de ejecución: Entrada (C'_t, A), salida C_{t+1}
    t ← t + 1
fin mientras
```

El algoritmo $\mathcal{A}$ recibe un sistema P reconocedor $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ de $\mathbf{CSEC}(k,1)$, siendo $k \geq 1$ y m un multiconjunto de entrada para este sistema. Sea $M = |\mathcal{M}_1 + \cdots + \mathcal{M}_q|$. Supongamos que cualquier computación de Π realiza a lo sumo p pasos de transición, $p \in \mathbb{N}^+$. Entonces, del Lema 9.1, el número de membranas en el sistema a lo largo de cualquier computación está acotado por $2M + q$.

Una transición de un sistema P reconocedor $\Pi + m$ es realizada en dos fases: la fase de selección y la fase de ejecución, mostradas en el Algoritmo 9.1.4 y el Algoritmo 9.1.5.

Algoritmo 9.1.4 Fase de selección

Entrada: Una configuración $\mathcal{C}_t$ de $\Pi + m$ en el instante t

$\mathcal{C}'_t \leftarrow \mathcal{C}_t$; $A \leftarrow \emptyset$; $B \leftarrow \emptyset$

para $r \in \mathcal{R}_i \wedge r \in \mathcal{R}_C$, de acuerdo al orden escogido **hacer**

  para cada membrana (i, σ_i) de $\mathcal{C}'_t$ de acuerdo con el orden lexicográfico **hacer**

    $n_r \leftarrow$ es el número de veces máximo que r es aplicable a (i, σ_i)

    si $n_r > 0$ **entonces**

      $\mathcal{C}'_t \leftarrow \mathcal{C}'_t - n_r \cdot LHS(r, (i, \sigma_i))$

      $A \leftarrow A \cup \{(r, n_r, (i, \sigma_i))\}$

      $B \leftarrow B \cup \{(i, \sigma_i)\}$

    fin si

  fin para

fin para

para $r \in \mathcal{R}_i \wedge r \in \mathcal{R}_S$ de acuerdo al orden escogido **hacer**

  para cada (a, i, σ_i) de acuerdo al orden lexicográfico, y tal que $(i, \sigma_i) \notin B$ **hacer**

    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t \setminus \{(a, i, \sigma_i)\}$

    $A \leftarrow A \cup \{(r, 1, (i, \sigma_i))\}$

    $B \leftarrow B \cup \{(i, \sigma_i)\}$

  fin para

fin para

Es fácil darse cuenta que este algoritmo es determinista y que el tiempo de ejecución es polinomial en el tamaño de Π dado que el número de ciclos del primer bucle **para** es de orden $O(|\mathcal{R}| \cdot M^2 \cdot q^2)$; y el número de ciclos del segundo bucle **para** es de orden $O(|\mathcal{R}| \cdot M \cdot q \cdot |\Gamma|)$.

Algoritmo 9.1.5 Fase de ejecución

Entrada: La salida $(\mathcal{C}'_t, A)$ de la fase de selección
  para cada $(r, n_r, (i, \sigma_i)) \in A, r \in \mathcal{R}_C$ **hacer**
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + n_r \cdot RHS(r, (r, \sigma_i))$
  fin para
  para cada $(r, 1, (i, \sigma_i)) \in A, r \in \mathcal{R}_S$ **hacer**
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(\lambda, i, \sigma_i)/\sigma_i 0\}$
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(\lambda, i, \sigma_i 1)\}$
    para cada $(x, i, \sigma_i) \in \mathcal{C}'_t$ de acuerdo con el orden lexicográfico **hacer**
      si $x \in \Gamma_0$ **entonces**
        $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x, i, \sigma_i)/\sigma_i 0\}$
      si no
        $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x, i, \sigma_i)/\sigma_i 1\}$
      fin si
    fin para
  fin para
  $\mathcal{C}_{t+1} \leftarrow \mathcal{C}'_t$

Este algoritmo es determinista y se ejecuta en tiempo polinomial en el tamaño de Π dado que el número de ciclos del primer bucle **para** es de orden $O(|\mathcal{R}| \cdot M^2 \cdot q^2)$; y el número de ciclos del segundo bucle **para** es de orden $O(|\mathcal{R} \cdot M \cdot q \cdot |\Gamma|)$.

Teorema 9.2. $\mathbf{P} = \mathbf{PMC}_{\mathcal{CSEC}(k,1)}$, *para* $k \geq 1$.

Demostración. Dado que $\mathbf{PMC}_{\mathcal{CSEC}(k,1)}$ es cerrada bajo reducibilidad en tiempo polinomial y es una clase no vacía, $\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{CSEC}(k,1)}$. En lo que sigue, demostraremos que $\mathbf{PMC}_{\mathcal{CSEC}(k,1)} \subseteq \mathbf{P}$. Sea $X \in \mathbf{PMC}_{\mathcal{CSEC}(k,1)}$ y $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ una familia de sistemas P reconocedores de $\mathcal{CSEC}(k, 1)$ que resuelve X de acuerdo a la definición dada en la Sección 4.4. Sea (cod, s) una codificación polinomial asociada a dicha solución. Si $u \in I_X$ es una instancia del problema X, entonces u será procesada por el sistema $\Pi(s(u)) + cod(u)$. Consideramos el Algoritmo 9.1.6 determinista $\mathcal{A}'$.

El algoritmo $\mathcal{A}'$ recibe una instancia u del problema de decisión $X = (I_X, \theta_X)$, y se ejecuta en tiempo polinomial. Las siguientes afirmaciones son equivalentes:

Algoritmo 9.1.6 Algoritmo $\mathcal{A}'$

Entrada: Una instancia u del problema X

Salida: *Sí* si $\Pi(s(u)) + cod(u)$ tiene una computación de aceptación, *No* en caso contrario

Construir el sistema $\Pi(s(u)) + cod(u)$

Ejecutar el algoritmo $\mathcal{A}$ con entrada $\Pi(s(u)) + cod(u)$

- $\theta_X(u) = 1$; es decir, la respuesta a la instancia u del problema X es afirmativa.
- Toda computación de $\Pi(s(u))+cod(u)$ es una computación de aceptación.
- La salida del algoritmo $\mathcal{A}'$ con entrada u es *Sí*.

Por lo tanto, $X \in \mathbf{P}$.

□

9.1.2. Presumible eficiencia de los sistemas P con comunicación evolutiva

Este apartado estará destinado, al contrario que el anterior, a demostrar que imponiendo una longitud máxima menos restrictiva a las reglas de comunicación evolutiva, podemos obtener soluciones eficientes a problemas presuntamente intratables.

9.1.2.1. Presumible eficiencia de los sistemas de $\mathcal{CDEC}(2,1)$

Para cada $n, p \in \mathbb{N}$, consideramos el sistema P reconocedor

$$\Pi(\langle n, p\rangle) = (\Gamma, \Sigma, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{M}_q, i_{in}, i_{out})$$

de $\mathcal{CDEC}(2,1)$ definido como sigue:

1. Alfabeto de trabajo Γ:
   $\{\texttt{yes}, \texttt{no}, y_1, y_2, n_1, n_2, \#\} \cup$
   $\{a_{i,j}, T_{i,j}, F_{i,j} \mid 1 \leq i \leq n, 0 \leq j \leq p\} \cup$
   $\{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 1 \leq i \leq n, 1 \leq j \leq p, 0 \leq k \leq 2np\} \cup$
   $\{\alpha_j \mid 1 \leq j \leq p+1\} \cup$
   $\{\delta_k \mid 0 \leq k \leq 2np+3\} \cup$
   $\{\delta'_k \mid 0 \leq k \leq 2np+1\}$.

2. Alfabeto de entrada Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,k,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.

3. Alfabeto del entorno $\mathcal{E}$: $\emptyset$.

4. $\mu = [\,[\quad]_1[\quad]_2 \ldots [\quad]_{np}[\quad]_{np+2}[\quad]_{np+3}[\quad]_{np+4}\,]_{np+1}$

5. $\mathcal{M}_k = \emptyset, 1 \leq k \leq np$,
$\mathcal{M}_{np+1} = \{\delta_0, \gamma^{n^2p^2-1}\}$,
$\mathcal{M}_{np+2} = \{a_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq p\} \cup \{\alpha_j \mid 1 \leq j \leq p+1\}$,
$\mathcal{M}_{np+3} = \{\delta'_0\}, \mathcal{M}_{np+4} = \{\gamma^{4np+4}\}$

6. El conjunto $\mathcal{R} = \mathcal{R}_1 \cup \cdots \cup \mathcal{R}_{np+4}$ consiste en las siguientes reglas:

   1.1 Reglas para generar p copias de las 2^n posibles valoraciones de verdad. Para ello, generaremos 2^{np} valoraciones de verdad parciales.

$$\left.\begin{array}{l} [\,a_{i,j}\,]_{np+2} \to [\,T_{i,j}\,]_{np+2}\,[\,F_{i,j}\,]_{np+2} \\ [\,[\,[\,T_{i,j}F_{i,j}\,]_{np+1}\,]_{np+2} \to [\,\#\,[\quad]_{np+1}\,]_{np+2} \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j, j' \leq p \end{array}$$

   1.2 Reglas para generar 2^{np} copias de $cod(\varphi)$.

$$\left.\begin{array}{l} [\,x_{i,j,0}\,[\,]_{i\perp j}\,]_{np+1} \to [\,[\,x_{i,j,1}\,]_{i\perp j}\,]_{np+1} \\ [\,\overline{x}_{i,j,0}\,[\quad]_{i\perp j}\,]_{np+1} \to [\,[\,\overline{x}_{i,j,1}\,]_{i\perp j}\,]_{np+1} \\ [\,x^*_{i,j,0}\,[\quad]_{i\perp j}\,]_{np+1} \to [\,[\,x^*_{i,j,1}\,]_{i\perp j}\,]_{np+1} \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$$

$$\left.\begin{array}{l} [\,\gamma\,[\,x_{i,j,k}\,]_{i\perp j}\,]_{np+1} \to [\,[\,x_{i,j,k+1}\,]_{i\perp j}\,]_{np+1} \\ [\,\gamma\,[\,\overline{x}_{i,j,k}\,]_{i\perp j}\,]_{np+1} \to [\,[\,\overline{x}_{i,j,k+1}\,]_{i\perp j}\,]_{np+1} \\ [\,\gamma\,[\,x^*_{i,j,k}\,]_{i\perp j}\,]_{np+1} \to [\,[\,x^*_{i,j,k+1}\,]_{i\perp j}\,]_{np+1} \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 1 \leq k \leq np-1 \end{array}$$

$$\left.\begin{array}{l} [\,x_{i,j,k}\,]_{i\perp j} \to [\,x_{i,j,k+1}\,]_{i\perp j}\,[\,x_{i,j,k+1}\,]_{i\perp j} \\ [\,\overline{x}_{i,j,k}\,]_{i\perp j} \to [\,\overline{x}_{i,j,k+1}\,]_{i\perp j}\,[\,\overline{x}_{i,j,k+1}\,]_{i\perp j} \\ [\,x^*_{i,j,k}\,]_{i\perp j} \to [\,x^*_{i,j,k+1}\,]_{i\perp j}\,[\,x^*_{i,j,k+1}\,]_{i\perp j} \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ np \leq k \leq 2np-1 \end{array}$$

$$\left.\begin{array}{l} [\,[\,x_{i,j,2np}\,]_{i\perp j}\,]_{np+1} \to [\,x_{i,j,2np}\,[\quad]_{i\perp j}\,]_{np+1} \\ [\,[\,\overline{x}_{i,j,2np}\,]_{i\perp j}\,]_{np+1} \to [\,\overline{x}_{i,j,2np}\,[\quad]_{i\perp j}\,]_{np+1} \\ [\,[\,x^*_{i,j,2np}\,]_{i\perp j}\,]_{np+1} \to [\,x^*_{i,j,2np}\,[\quad]_{i\perp j}\,]_{np+1} \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$$

$$[\,\gamma\,[\,\delta'_k\,]_{np+3}\,]_{np+1} \to [\,[\,\delta'_{k+1}\,]_{np+3}\,]_{np+1} \quad \text{, para } 0 \leq k \leq np$$

$$[\,\delta'_k\,]_{np+3} \to [\,\delta'_{k+1}\,]_{np+3}\,[\,\delta'_{k+1}\,]_{np+3} \quad \text{, para } np+1 \leq k \leq 2np$$

$$[\,[\,\delta'_{2np+1}\,]_{np+3}\,]_{np+1} \to [\,\delta'_{2np+1}\,[\quad]_{np+3}\,]_{np+1}$$

   2.1 Reglas para chequear las cláusulas satisfechas.

$$\left.\begin{array}{l} [\,x_{i,j,2np}\,[\,T_{i,j}\,]_{np+2}\,]_{np+1} \to [\,[\,c_j\,]_{np+2}\,]_{np+1} \\ [\,\overline{x}_{i,j,2np}\,[\,T_{i,j}\,]_{np+2}\,]_{np+1} \to [\,[\,\#\,]_{np+2}\,]_{np+1} \\ [\,x^*_{i,j,2np}\,[\,T_{i,j}\,]_{np+2}\,]_{np+1} \to [\,[\,\#\,]_{np+2}\,]_{np+1} \\ [\,x_{i,j,2np}\,[\,F_{i,j}\,]_{np+2}\,]_{np+1} \to [\,[\,\#\,]_{np+2}\,]_{np+1} \\ [\,\overline{x}_{i,j,2np}\,[\,F_{i,j}\,]_{np+2}\,]_{np+1} \to [\,[\,c_j\,]_{np+2}\,]_{np+1} \\ [\,x^*_{i,j,2np}\,[\,F_{i,j}\,]_{np+2}\,]_{np+1} \to [\,[\,\#\,]_{np+2}\,]_{np+1} \end{array}\right\} \text{para } \begin{array}{l} 1 \leq i \leq n, \\ 1 \leq j \leq p \end{array}$$

3.1 Reglas para chequear si todas las cláusulas son satisfechas por una valoración de verdad.

$$[\,\delta'_{2np+1}\,[\,\alpha_{p+1}\,]_{np+2}\,]_{np+1} \to [\,[\,\alpha'_{p+1}\,]_{np+2}\,]_{np+1}$$
$$[\,c_j\,[\,\alpha_j\,]_{np+2}\,]_{np+1} \to [\,\#\,[\quad]_{np+2}\,]_{np+1} \quad \text{, para } 1 \leq j \leq p$$

4.1 Contador general.

$$[\,\delta_k\,[\,\gamma\,]_{np+1}\,]_{np+4} \to [\,\delta_{k+1}\,[\quad]_{np+1}\,]_{np+4} \quad \text{, para } 0 \leq k \leq 2np+2$$

4.2 Reglas para devolver una respuesta negativa.

$$[\,[\,\alpha_j\alpha'_{p+1}\,]_{np+2}\,]_{np+1} \to [\,n_1\,[\quad]_{np+2}\,]_{np+1} \quad \text{, para } 1 \leq j \leq p$$
$$[\,n_1\,[\quad]_{np+2}\,]_{np+1} \to [\,[\,n_1\,]_{np+2}\,]_{np+1}$$
$$[\,\delta_{2np+3}\,[\,n_1\,]_{np+2}\,]_{np+1} \to [\,n_2\,[\quad]_{np+2}\,]_{np+1}$$
$$[\,[\,n_2\,]_{np+1}\,]_0 \to [\,\texttt{no}\,[\quad]_{np+1}\,]_0$$

4.3 Reglas para devolver una respuesta afirmativa.

$$[\,\delta_{2np+3}\,[\,\alpha'_{p+1}\,]_{np+2}\,]_{np+1} \to [\,[\,y_1\,]_{np+2}\,]_{np+1}$$
$$[\,\gamma\,[\,y_1\,]_{np+2}\,]_{np+1} \to [\,y_2\,[\quad]_{np+2}\,]_{np+1}$$
$$[\,[\,y_2\,]_{np+1}\,]_0 \to [\,\texttt{yes}\,[\quad]_{np+1}\,]_0$$

7. La membrana de entrada es la membrana etiquetada por 1 ($i_{in} = np+1$) y la región de salida es el entorno ($i_{out} = env$).

9.1.3. Descripción de la computación

Consideramos la codificación de la fórmula φ como se sigue. Sea $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_p, C_j = l_{j_1} \vee \cdots \vee l_{j_{r_j}}, 1 \leq j \leq p, l_{j_k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$ una instancia del problema SAT. Entonces la fórmula Booleana φ será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$, en donde $s(\varphi) = \langle n, p\rangle$ y $cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$.

Denotamos con $cod_k(\varphi)$ como el conjunto $cod(\varphi)$ con el tercer subíndice igual a k.

La solución propuesta sigue un algoritmo de fuerza bruta en el marco de sistemas P reconocedores con reglas de comunicación evolutiva y con reglas de división, y consiste en las siguientes fases:

- *Fase de generación*: Mediante el uso de reglas de 1.1, p copias de cada 2^n valoración de verdad asociada a $\{x_1, \ldots, x_n\}$ son generadas en las membranas etiquetadas por $np+2$. Para ello, $2^{np} - 2^n$ valoraciones de verdad parciales son producidas eliminando objetos cuando dos valores de verdad distintos sean asignados a la misma variable.

En paralelo, 2^{np} copias de cada objeto $x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k}$ que aparece en $cod(\varphi)$ son generadas aplicando las reglas de 1.2. Para sincronizar el comienzo de la primera fase de chequeo, el tercer subíndice k evolucionará hasta alcanzar el valor $2np$, y entonces los objetos $x_{i,j,2np}, \overline{x}_{i,j,2np}, x^*_{i,j,2np}$ serán enviados a la membrana $np+1$ para interactuar con los objetos $T_{i,j}$ y $F_{i,j}$. Esta fase tarda exactamente $2np+1$ pasos de computación.

- *Primera fase de chequeo*: Las p copias de cada una de las 2^n valoraciones de verdad asociadas a $\{x_1, \ldots, x_n\}$ generadas en las membranas etiquetadas por $np+2$ cooperan con las 2^n copias de los objetos $x_{i,j,2np}, \overline{x}_{i,j,2np}, x^*_{i,j,2np}$ usando las reglas de 2.1 para obtener qué cláusulas de la fórmula φ han sido satisfechas por una valoración de verdad. Vale la pena señalar que en esta fase, las $2^{np} - 2^n$ valoraciones de verdad parciales no podrán producir los objetos correspondientes a todas las cláusulas $C_1, \ldots, C_p$ en el caso que φ no sea satisfactible. Esta fase toma 1 paso de computación.

- *Segunda fase de chequeo*: Usando las reglas de 3.1, el α_j es eliminado de una membrana etiquetada por $np+2$ si y sólo si la valoración de verdad asociada a dicha membrana hace verdadera la cláusula C_j. Por lo tanto, una membrana etiquetada por $np+2$ codificará una valoración de verdad que hace verdadera la fórmula φ si y sólo si no queda ningún objeto α_j después de la ejecución de esta fase, que tarda exactamente 1 paso.

- *Fase de salida*: Con las reglas de 4.2 y 4.3, el sistema puede dar una respuesta afirmativa o negativa dependiendo de si la fórmula φ es o no satisfactible. Esta fase tarda exactamente 4 pasos de computación, independientemente de ser o no satisfactible la fórmula φ.

9.1.4. Resultado

Teorema 9.3. $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{CDEC}(2,1)}$

Demostración. La familia de sistemas P construida previamente verifica lo siguiente:

- Todo sistema de la familia $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ es un sistema P reconocedor de $\mathcal{TDEC}(2,1)$.

- La familia $\mathbf{\Pi}$ es polinomialmente uniforme por máquinas de Turing dado que para cada $n, p \in \mathbb{N}$, las reglas de $\Pi(\langle n, p\rangle)$ de la familia son recursivamente definidas por $n, p \in \mathbb{N}$, y la cantidad de recursos necesitados para

construir un elemento de la familia es de orden polinomial con respecto a n y a p, como se muestra debajo:

- Tamaño del alfabeto: $\Theta(n^2p^2)$.
- Número inicial de células: $4 + np \in \Theta(np)$.
- Número inicial de objetos en células: $n^2p^2 + 5np + p + 4 \in \Theta(n^2p^2)$.
- Número de reglas: $\Theta(n^2p^2)$.
- Número máximo de objetos involucrados en cualquier regla: $3 \in \Theta(1)$.

- El par de funciones computables en tiempo polinomial (cod, s) definido cumple lo siguiente: para cada fórmula φ del problema `SAT`, $s(\varphi)$ es un número natural, $cod(\varphi)$ es un multiconjunto de entrada del sistema $\Pi(s(\varphi))$ y para cada $t \in \mathbb{N}$, $s^{-1}(t)$ es un conjunto finito.

- La familia $\mathbf{\Pi}$ está polinomialmente acotada en tiempo: de hecho, para cada fórmula φ del problema `SAT`, el sistema P reconocedor $\Pi(s(\varphi)) + cod(\varphi)$ tarda exactamente $\lfloor \frac{3np}{2} \rfloor + np + 6$ pasos de computación en devolver una respuesta positiva o negativa, siendo n el número de variables y p el número de cláusulas de φ.

- La familia $\mathbf{\Pi}$ es adecuada con respecto a (X, cod, s): de hecho, para cada fórmula φ, si las computaciones de $\Pi(s(\varphi)) + cod(\varphi)$ son de *aceptación*, entonces φ es satisfactible.

- La familia $\mathbf{\Pi}$ es completa con respecto a (X, cod, s): de hecho, para cada fórmula φ que sea satisfactible, todas las computaciones de $\Pi(s(\varphi)) + cod(\varphi)$ son de *aceptación*.

□

Corolario 9.2. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{CDEC}(2,1)}$

Demostración. Basta con observar que `SAT` es un problema **NP**-completo, $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{TDEC}(2,1)}$ y la clase de complejidad $\mathbf{PMC}_{\mathcal{TDEC}(2,1)}$ está cerrada bajo reducibilidad en tiempo polinomial y bajo complementario. □

De hecho, esta familia de sistemas P con reglas symport/antiport evolutivas y reglas de división no hace uso del entorno, por lo tanto:

Corolario 9.3. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\widehat{\mathcal{CDEC}}(2,1)}$

9.1.5. Presumible eficiencia de los sistemas de $\mathcal{CSEC}(2,2)$

Para cada $n, p \in \mathbb{N}$, consideramos el sistema P reconocedor

$$\Pi(\langle n, p\rangle) = (\Gamma, \Gamma_0, \Gamma_1, \Sigma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$$

de $\mathcal{CSEC}(2,2)$ definido como sigue:

1. Alfabeto de trabajo Γ:
   $\{\texttt{yes}, \texttt{no}, y_1, y_2, n_1, n_2, \#\} \cup$
   $\{a_{i,j} \mid 1 \leq i \leq n, 0 \leq j \leq i\} \cup$
   $\{a'_{i,j} \mid 2 \leq i \leq n, 0 \leq j \leq i-1\} \cup$
   $\{a^L_{i,j}, a^R_{i,j} \mid 2 \leq i \leq n, 1 \leq j \leq i-1\} \cup$
   $\{\alpha_j, \alpha'_j, \alpha^L_j, \alpha^R_j \mid 1 \leq j \leq p+1\} \cup$
   $\{t_i, f_i, t'_i, t''_i f''_i, t^L_i, t^R_i, f^L_i, f^R_i \mid 1 \leq i \leq n\} \cup$
   $\{\beta_{l,k}, \beta'_{l,k}, \beta^L_{l,k}, \beta^R_{l,k} \mid 0 \leq k \leq n, 1 \leq l \leq n\} \cup$
   $\{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n+j-1\} \cup$
   $\{x'_{i,j,k}, \overline{x}'_{i,j,k}, {x^*}'_{i,j,k}, x''_{i,j,k}, \overline{x}''_{i,j,k}, {x^*}''_{i,j,k}, x'''_{i,j,k}, \overline{x}'''_{i,j,k}, {x^*}'''_{i,j,k}, |$
   $0 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n\} \cup$
   $\{c_{j,k} \mid 1 \leq j \leq p, j \leq k \leq p\} \cup \{\delta_i \mid 0 \leq i \leq 4n+p+2\} \cup$
   $\{\delta'_i \mid 0 \leq i \leq 4n+p\} \cup \{\gamma_k \mid 0 \leq k \leq n+1\}$.

2. $\Gamma_1 = \Gamma \setminus \Gamma_0$, $\Gamma_0 = \{a^L_{i,j} \mid 2 \leq i \leq n, 1 \leq j \leq i-1\} \cup$
   $\{\alpha^L_j \mid 1 \leq j \leq p+1\} \cup \{t^L_i, f^L_i \mid 1 \leq i \leq n\} \cup$
   $\{\beta^L_{l,k} \mid 0 \leq k \leq n, k+1 \leq l \leq n\}$

3. Alfabeto de entrada Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,k,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.

4. Alfabeto del entorno $\mathcal{E}$: $\emptyset$.

5. $\mu = [\,[\quad]_2\,[\quad]_3\,]_1$.

6. $\mathcal{M}_1 = \{\delta_0, \delta'_0, \gamma_0^{2n^2+6np+4n+2p+6}\}$,
   $\mathcal{M}_2 = \{a_{i,0} \mid 1 \leq i \leq n\} \cup \{\alpha_j \mid 1 \leq j \leq p+1\}$,
   $\mathcal{M}_3 = \{\gamma_0^{2n^2+6np+4n+2p+6}\} \cup \{\beta_{l,0}^{n+p+1} \mid 1 \leq l \leq n\}$.

7. El conjunto $\mathcal{R}$ consiste en las siguientes reglas:

   0.1 Reglas para generar los objetos γ para simular el entorno

   $$\left.\begin{array}{l} [\,[\,\gamma_k\,]_3\,]_1 \to [\,\gamma^2_{k+1}\,[\quad]_3\,]_1 \\ [\,\gamma_k\,[\quad]_3\,]_1 \to [\,[\,\gamma^2_{k+1}\,]_3\,]_1 \end{array}\right\} \text{para } 1 \leq k \leq n$$

   $$[\,[\,\gamma_{n+1}\,]_3\,]_1 \to [\,\gamma\,[\quad]_3\,]_1$$
   $$[\,\gamma_{n+1}\,[\quad]_3\,]_1 \to [\,[\,\gamma\,]_3\,]_1$$

1.1 Reglas para los pasos $(4k+1)$.

$$[\,\gamma\,[\,a_{i,i-1}\,]_2\,]_1 \to [\,[\,a'_{i,i-1}t'_i\,]_2\,]_1 \quad \text{, para } 1 \le i \le n$$

$$\left.\begin{array}{l}[\,\gamma\,[\,t_i\,]_2\,]_1 \to [\,[\,t''_i\,]_2\,]_1 \\ [\,\gamma\,[\,f_i\,]_2\,]_1 \to [\,[\,f''_i\,]_2\,]_1\end{array}\right\} \text{para } 1 \le i \le n$$

$$[\,\gamma\,[\,a_{i,j}\,]_2\,]_1 \to [\,[\,a'_{i,j}\,]_2\,]_1 \quad \text{, para } 2 \le i \le n, 0 \le j \le i-2$$

$$[\,\gamma\,[\,\alpha_j\,]_2\,]_1 \to [\,[\,\alpha'_j\,]_2\,]_1 \quad \text{, para } 1 \le j \le p+1$$

$$\left.[\,\gamma\,[\,\beta_{l,k}\,]_3\,]_1 \to [\,[\,\beta'_{l,k}\,]_3\,]_1\ \right\} \text{para } \begin{array}{l}0 \le k \le n, \\ k+1 \le l \le n\end{array}$$

$$\left.\begin{array}{l}[\,x_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,x'_{i,j,k}\,[\quad]_3\,]_1 \\ [\,\overline{x}_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,\overline{x}'_{i,j,k}\,[\quad]_3\,]_1 \\ [\,x^*_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,{x^*}'_{i,j,k}\,[\quad]_3\,]_1\end{array}\right\} \text{para } \begin{array}{r}1 \le i \le n, \\ 1 \le j \le p, \\ 0 \le k \le n-1\end{array}$$

1.2 Reglas para los pasos $(4k+2)$.

$$\left.\begin{array}{l}[\,\gamma\,[\,a'_{i,i-1}\,]_2\,]_1 \to [\,[\,a_{i,i}f_i^R\,]_2\,]_1 \\ [\,\gamma\,[\,t'_i\,]_2\,]_1 \to [\,[\,t_i^L\,]_2\,]_1\end{array}\right\} \text{para } 1 \le i \le n$$

$$\left.\begin{array}{l}[\,\gamma\,[\,t''_i\,]_2\,]_1 \to [\,[\,t_i^L\,t_i^R\,]_2\,]_1 \\ [\,\gamma\,[\,f''_i\,]_2\,]_1 \to [\,[\,f_i^Lf_i^R\,]_2\,]_1\end{array}\right\} \text{para } 1 \le i \le n$$

$$[\,\gamma\,[\,a'_{i,j}\,]_2\,]_1 \to [\,[\,a^L_{i,j+1}a^R_{i,j+1}\,]_2\,]_1 \quad \text{, para } \begin{array}{l}2 \le i \le n, \\ 0 \le j \le i-1\end{array}$$

$$[\,\gamma\,[\,\alpha'_j\,]_2\,]_1 \to [\,[\,\alpha_j^L\alpha_j^R\,]_2\,]_1 \quad \text{, para } 1 \le j \le p+1$$

$$[\,\gamma\,[\,\beta'_{l,k}\,]_3\,]_1 \to [\,[\,\beta^L_{l,k+1}\beta^R_{l,k+1}\,]_3\,]_1 \quad \text{, para } \begin{array}{l}0 \le k \le n, \\ k+1 \le l \le n\end{array}$$

$$\left.\begin{array}{l}[\,x'_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,x''^2_{i,j,k+1}\,[\quad]_3\,]_1 \\ [\,\overline{x}'_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,\overline{x}''^2_{i,j,k+1}\,[\quad]_3\,]_1 \\ [\,{x^*}'_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,{x^*}''^2_{i,j,k+1}\,[\quad]_3\,]_1\end{array}\right\} \begin{array}{l}1 \le i \le n, \\ 1 \le j \le p, \\ 0 \le k \le n-1\end{array}$$

1.3 Reglas para los pasos $(4k+3)$.

$$[\,a_{i,i}\,]_2 \to [\,\Gamma_0\,]_2\,[\,\Gamma_1\,]_2 \quad \text{, para } 1 \le i \le n$$

$$\left.\begin{array}{l}[\,[\,\beta^O_{k,k}\,]_3\,]_1 \to [\,\beta^O_{k,k}\,[\quad]_3\,]_1 \\ [\,[\,\beta^O_{l,k}\,]_3\,]_1 \to [\,\beta_{l,k}\,[\quad]_3\,]_1\end{array}\right\} \text{para } \begin{array}{l}O \in \{L,R\}, \\ 1 \le k \le n, \\ k+1 \le l \le n\end{array}$$

$$\left.\begin{array}{l}[\,x''_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,x'''_{i,j,k}\,[\quad]_3\,]_1 \\ [\,\overline{x}''_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,\overline{x}'''_{i,j,k}\,[\quad]_3\,]_1 \\ [\,{x^*}''_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,{x^*}'''_{i,j,k}\,[\quad]_3\,]_1\end{array}\right\} \text{para } \begin{array}{r}1 \le i \le n, \\ 1 \le j \le p, \\ 1 \le k \le n\end{array}$$

1.4 Reglas para los pasos $(4k)$.

$$\left.\begin{array}{l}[\,\beta^O_{k,k}\,[\,a^O_{i,j}\,]_2\,]_1 \to [\,[\,a_{i,j}\,]_2\,]_1\\ [\,\beta^O_{k,k}\,[\,r^O_i\,]_2\,]_1 \to [\,[\,r_i\,]_2\,]_1\end{array}\right\} \text{para } \begin{array}{l} O \in \{L,R\},\\ r \in \{t,f\},\\ 1 \le i \le n,\\ 1 \le j \le n,\\ 1 \le k \le n\end{array}$$

$$[\,\beta^O_{k,k}\,[\,\alpha^O_j\,]_2\,]_1 \to [\,[\,\alpha_j\,]_2\,]_1 \quad \text{, para } \begin{array}{l} O \in \{L,R\},\\ 1 \le j \le p+1,\\ 0 \le k \le n\end{array}$$

$$\left.\begin{array}{l}[\,x'''_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,x_{i,j,k}\,[\quad]_3\,]_1\\ [\,\overline{x}'''_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,\overline{x}_{i,j,k}\,[\quad]_3\,]_1\\ [\,x^{*'''}_{i,j,k}\,[\,\gamma\,]_3\,]_1 \to [\,x^*_{i,j,k}\,[\quad]_3\,]_1\end{array}\right\} \text{para } \begin{array}{l} 1 \le i \le n,\\ 1 \le j \le p,\\ 0 \le k \le n\end{array}$$

$$[\,\beta_{l,k}\,[\quad]_3\,]_1 \to [\,[\,\beta_{l,k}\,]_3\,]_1 \quad \text{, para } 0 \le k \le n, k+1 \le l \le n$$

2.1 Reglas para chequear las cláusulas satisfechas.

$$\left.\begin{array}{l}[\,x_{i,j,n+j-1}\,[\,t_i\,]_2\,]_1 \to [\,[\,c_{j,j}t_i\,]_2\,]_1\\ [\,\overline{x}_{i,j,n+j-1}\,[\,t_i\,]_2\,]_1 \to [\,[\,t_i\,]_2\,]_1\\ [\,x^*_{i,j,n+j-1}\,[\,t_i\,]_2\,]_1 \to [\,[\,t_i\,]_2\,]_1\\ [\,x_{i,j,n+j-1}\,[\,f_i\,]_2\,]_1 \to [\,[\,f_i\,]_2\,]_1\\ [\,\overline{x}_{i,j,n+j-1}\,[\,f_i\,]_2\,]_1 \to [\,[\,c_{j,j}f_i\,]_2\,]_1\\ [\,x^*_{i,j,n+j-1}\,[\,f_i\,]_2\,]_1 \to [\,[\,f_i\,]_2\,]_1\end{array}\right\} \text{para } 1 \le i \le n, 1 \le j \le p$$

$$\left.\begin{array}{l}[\,x_{i,j,n+k}\,[\,\gamma\,]_3\,]_1 \to [\,x_{i,j,n+k+1}\,[\quad]_3\,]_1\\ [\,\overline{x}_{i,j,n+k}\,[\,\gamma\,]_3\,]_1 \to [\,\overline{x}_{i,j,n+k+1}\,[\quad]_3\,]_1\\ [\,x^*_{i,j,n+k}\,[\,\gamma\,]_3\,]_1 \to [\,x^*_{i,j,n+k+1}\,[\quad]_3\,]_1\end{array}\right\} \text{para } \begin{array}{l} 1 \le i \le n,\\ 1 \le j \le p,\\ 0 \le k \le j-2\end{array}$$

$$[\,c_{j,k}\,[\,\gamma\,]_2\,]_1 \to [\,c_{j,k+1}\,[\quad]_2\,]_1 \quad \text{, para } 1 \le j \le p, j \le k \le p-1$$

3.1 Reglas para chequear si todas las cláusulas son satisfechas por una valoración de verdad.

$$[\,\delta'_{4n+p}\,[\,\alpha_{p+1}\,]_2\,]_1 \to [\,[\,\alpha'_{p+1}\,]_2\,]_1$$

$$[\,[\,\alpha_j\,c_{j,p}\,]_2\,]_1 \to [\,\#\,[\quad]_2\,]_1 \quad \text{, para } 1 \le j \le p$$

4.1 Contadores generales.

$$[\,\delta_i\,[\,\gamma\,]_3\,]_1 \to [\,\delta_{i+1}\,[\quad]_3\,]_1 \quad \text{, para } 0 \le i \le 4n+p+1$$

$$[\,\delta'_{4i+1}\,[\,\gamma\,]_3\,]_1 \to [\,{\delta'}^2_{4i+2}\,[\quad]_3\,]_1 \quad \text{, para } 0 \le i \le n-1$$

$$[\,\delta'_{4i+k}\,[\,\gamma\,]_3\,]_1 \to [\,\delta'_{4i+k+1}\,[\quad]_3\,]_1 \quad \text{, para } 0 \le i \le n-1, k \in \{0,2,3\}$$

$$[\,\delta'_{4n+i}\,[\,\gamma\,]_3\,]_1 \to [\,\delta'_{4n+i+1}\,[\quad]_3\,]_1 \quad \text{, para } 0 \le i \le p-1$$

4.2 Reglas para devolver una respuesta negativa.

$$[\,[\,\alpha_j\alpha'_{p+1}\,]_2\,]_0 \to [\,n_1\,[\quad]_2\,]_0 \quad \text{, para } 1 \le j \le p$$

$[\, n_1 [\quad]_2]_1 \to [\, [\, n_1]_2]_1$
$[\, \delta_{4n+p+2} [\, n_1]_2]_1 \to [\, n_2 [\quad]_2]_1$
$[\, [\, n_2]_1]_0 \to [\, \texttt{no} [\quad]_1]_0$

4.3 Reglas para devolver una respuesta afirmativa.

$[\, \delta_{4n+p+2} [\, \alpha'_{p+1}]_2]_1 \to [\, [\, y_1]_2]_1$
$[\, [\, y_1]_2]_1 \to [\, y_2 [\quad]_2]_1$
$[\, [\, y_2]_1]_0 \to [\, \texttt{yes} [\quad]_1]_0$

8. La membrana de entrada es la membrana etiquetada por 1 ($i_{in} = 1$) y la región de salida es el entorno ($i_{out} = env$).

Sea $\varphi = C_1 \wedge \cdots \wedge C_p$ una instancia del problema SAT que tiene p cláusulas $C_j = l_{j,1} \vee \cdots \vee l_{j,r_j}, 1 \leq j \leq p$, donde $Var(\varphi) = \{x_1, \ldots, x_n\}$, y $l_{j,k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}, 1 \leq j \leq p, 1 \leq k \leq r_j$. Asumimos que el número de variables, n, y el número de cláusulas, p, de φ, son mayores o iguales que 2. Entonces la instancia φ será procesada por el sistema $\Pi(s(\varphi)) + cod(\varphi)$, en donde $s(\varphi) = \langle n, p \rangle$ y $cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$.

Denotamos con $cod_k(\varphi)$ el conjunto de elementos de $cod(\varphi)$ con el tercer subíndice igual a k. De la misma forma, denotamos con $cod'_k(\varphi)$, $cod''_k(\varphi)$ y $cod'''_k(\varphi)$ como los conjuntos de elementos de $cod_k(\varphi)$ cuyos elementos tienen prima, doble prima o triple prima, respectivamente. Por conveniencia de notación, denotamos con $cod^j_k(\varphi)$ el subconjunto de elementos de $cod_k(\varphi)$ con los elementos de las cláusulas $C_j, \ldots, C_p$.

A continuación, damos una descripción informal de como funciona el sistema.

La solución propuesta sigue un algoritmo de fuerza bruta en el marco de los sistemas P reconocedores con reglas symport/antiport evolutivas y reglas de separación, y consiste en las siguientes fases:

- *Fase de pregeneración*: Para simular los objetos γ del entorno, generamos previamente al inicio del resto de la ejecución del sistema completo un número exponencial de objetos γ, que son suficientes para ser usados a lo largo de toda la computación como si el entorno jugara un papel activo en la misma. Esta fase toma exactamente $n + 1$ pasos.

- *Fase de generación*: Usando reglas de separación cada 4 pasos, producimos 2^n membranas etiquetadas por 2 que contendrán todas las posibles valoraciones de verdad. Al mismo tiempo, generamos 2^n copias de $cod_n(\varphi)$. Esta fase tarda n pasos de computación, siendo n el número de variables de φ.

- *Primera fase de chequeo*: Con las reglas de 2.1, podemos chequear qué cláusulas de la fórmula φ han sido satisfechas por cada una de las valoraciones de verdad. Esta fase tarda exactamente p pasos, siendo p el número de cláusulas de la fórmula φ.

- *Segunda fase de chequeo*: Con las reglas de 3.1, eliminamos los objetos α_j tales que son eliminados de una membrana si y sólo si la valoración de verdad asociada a dicha membrana hace verdadera la cláusula C_j. Esta fase toma exactamente 1 paso de computación.

- *Fase de salida*: Con las reglas de 4.2 y 4.3, devolvemos una respuesta afirmativa o negativa dependiendo de si la fórmula φ es o no satisfactible. Esta fase tarda exactamente 4 pasos de computación, independientemente de si la fórmula es satisfactible o no.

Teorema 9.4. $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{CSEC}(2,2)}$

El diseño introducido aquí es el análogo al presentado en la Sección 8.2 pero con sistemas P a modo de células en lugar de a modo de tejidos, presentando un comportamiento similar al anterior, con lo que la verificación sigue un camino idéntico al marcado en el capítulo anterior.

Corolario 9.4. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{CSEC}(2,2)}$

Demostración. Basta con observar que SAT es un problema **NP**-completo, $\texttt{SAT} \in \mathbf{PMC}_{\mathcal{CSEC}(2,2)}$ y la clase de complejidad $\mathbf{PMC}_{\mathcal{CSEC}(2,2)}$ está cerrada bajo reducibilidad en tiempo polinomial y bajo complementario. □

De hecho, como pasaba en la sección anterior, la fase de pregeneración hace que los objetos γ simulen el comportamiento del entorno, haciendo que este no sea necesario para alcanzar la presumible eficiencia, por lo tanto:

Corolario 9.5. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\widehat{\mathcal{CSEC}}(2,2)}$.

9.2. Nuevas fronteras obtenidas

En este capítulo se ha introducido el marco de computación de sistemas P a modo de células con reglas de comunicación evolutiva y con reglas de división y separación. Para ello, la sintaxis usada es similar a la usada en los sistemas análogos a modo de tejidos, pero teniendo en cuenta la jerarquía de membranas. Aparte de eso, la semántica tiene un funcionamiento similar a los

anteriores, basándose en las reglas symport/antiport clásicas pero dándole a los objetos que disparan la regla la capacidad de evolucionar gracias a la ejecución de la misma. Los resultados, al igual que ocurrían con los sistemas P con reglas symport/antiport clásicas, que no diferían con respecto a los problemas que podían resolver eficientemente con respecto a los sistemas P de tejidos con reglas symport/antiport, en este caso ocurre algo similar, teniendo los resultados presentados previamente, y que comentaremos a continuación.

Respecto al marco de los sistemas P con reglas de comunicación evolutiva y reglas de división, hemos comprobado que con reglas de longitud a lo sumo 2 sólo se pueden resolver de manera eficiente problemas de la clase **P**, mientras que si aumentamos la longitud a 3 podemos obtener soluciones eficientes a problemas **NP**-completos, como la que se muestra en la la sección anterior. Por tanto, hemos encontrado una frontera al pasar de reglas de comunicación de tamaño 2 a tamaño 3.

De la misma forma, sabemos que los sistemas P de $\mathcal{CDEC}(1,k)$ no son eficientes, mientras que si permitimos que las reglas de comunicación sean *cooperativas*, es decir, que la parte izquierda de dichas reglas puedan tener longitud, al menos, 2, entonces podemos obtener soluciones eficientes a problemas presuntamente intratables. Es decir, que pasar de restringir la longitud de la parte izquierda de las reglas de comunicación de 1 a 2 en el marco de los sistemas P con reglas de comunicación evolutiva y división equivale a pasar de la no eficiencia a la presumible eficiencia.

Pasando al uso de las reglas de separación, dichos sistemas no pueden resolver problemas presuntamente intratables si la longitud de las reglas es a lo sumo 2. Por otra parte, gracias a la solución presentada en la sección anterior al problema `SAT` por medio de sistemas P de $\mathcal{CSEC}(2,2)$ podemos confirmar que con reglas de comunicación con longitud a lo sumo 4 alcanzamos la presumible eficiencia. Esto confirma una frontera entre usar reglas de comunicación de longitud 2 a longitud 4, aunque es desconocido si los sistemas P de $\mathcal{CSEC}(3)$ son o no presumiblemente eficientes.

Por otra parte, y dado que tanto la clase $\mathbf{PMC}_{\mathcal{CSEC}(1,k)}$ como la clase $\mathbf{PMC}_{\mathcal{CSEC}(k,1)}$, con $k \geq 1$ son clases no eficientes y que se ha demostrado que $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{\mathcal{CSEC}(2,2)}$, podemos concluir las siguientes fronteras:

- Pasar de restringir la parte derecha de 1 a 2 en el marco de sistemas P de $\mathcal{CSEC}(2,k)$ equivale a pasar de la no eficiencia a la presumible eficiencia.
- Pasar de restringir la parte izquierda de 1 a 2 en el marco de sistemas P de

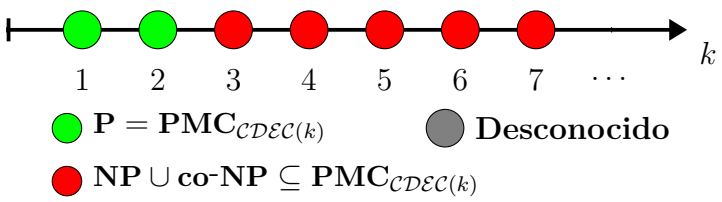


Figura 9.1: Fronteras de la eficiencia en sistemas P con reglas de comunicación evolutiva y división celular (métrica [60])

$\mathcal{CSEC}(k, 2)$ equivale a pasar de la no eficiencia a la presumible eficiencia.

Además, las soluciones proporcionadas en la sección anterior no hacen uso del entorno como un elemento activo del sistema, con lo que facilita la definición de fronteras que, en este caso, no existen con respecto a sus homólogos con entorno. Sí que podemos ver aquí una frontera entre los sistemas que no hacen uso del entorno cuando usan reglas de separación entre los sistemas P con reglas symport/antiport clásicas y reglas symport/antiport evolutivas, dado que en [26] se demuestra que los primeros son sistemas no eficientes mientras que en este documento se dan argumentos para demostrar que los segundos pueden resolver instancias de problemas presuntamente intratables.
En las Figuras 9.1, 9.2, 9.3 y 9.4 se muestra el panorama de esta clase de sistemas P desde el punto de vista de la complejidad computacional.

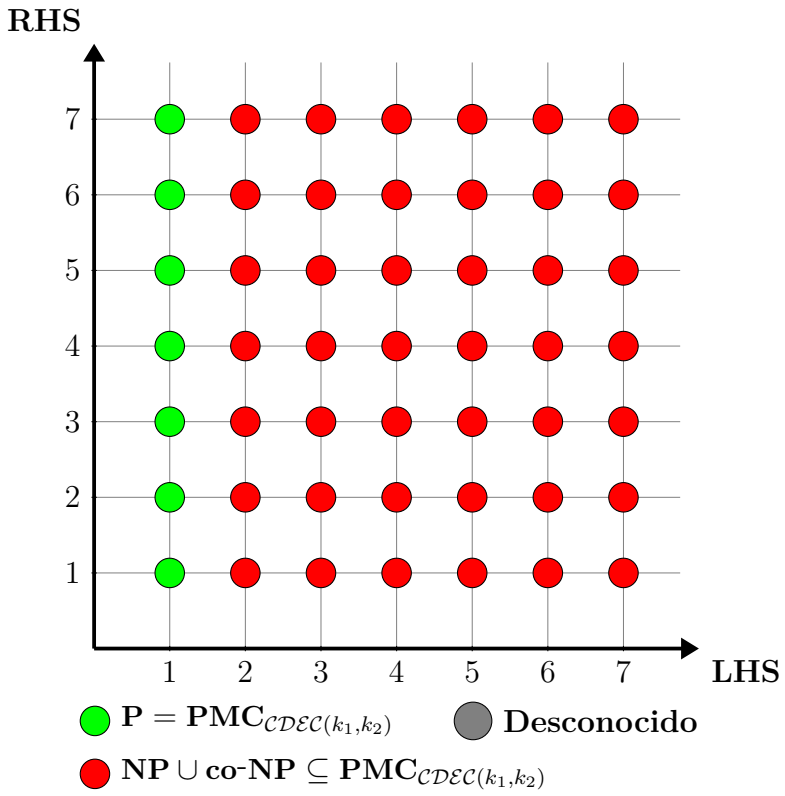


Figura 9.2: Fronteras de la eficiencia en sistemas P con reglas de comunicación evolutiva y división celular (métrica [39])

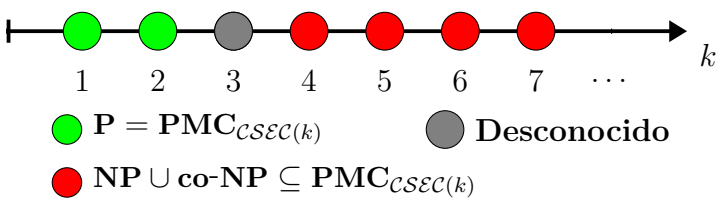


Figura 9.3: Fronteras de la eficiencia en sistemas P con reglas de comunicación evolutiva y separación celular (métrica [60])

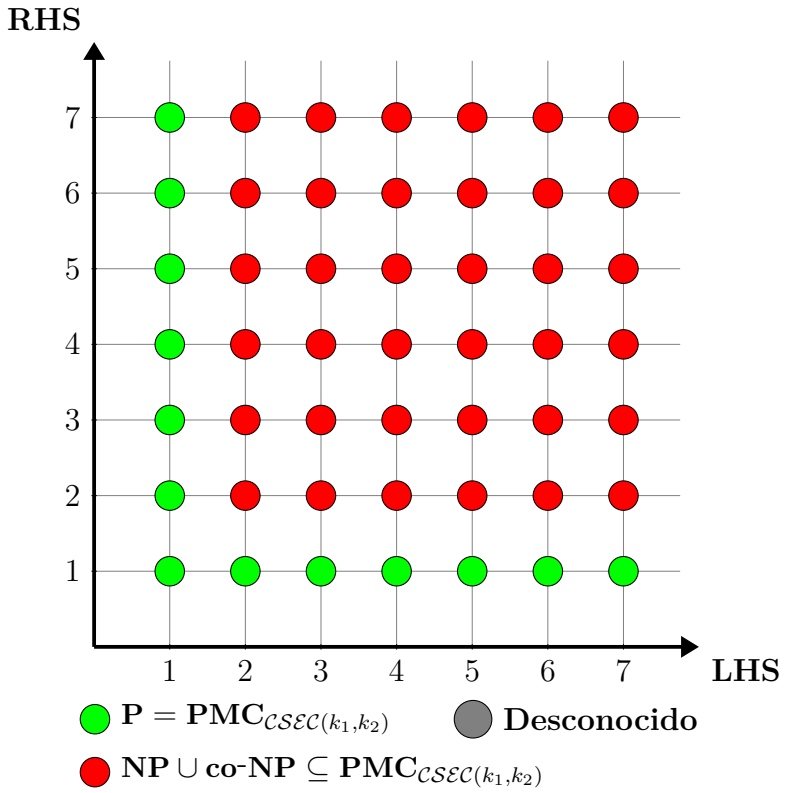


Figura 9.4: Fronteras de la eficiencia en sistemas P con reglas de comunicación evolutiva y separación celular (métrica [39])

Capítulo 10

Conclusiones y trabajo futuro

Finalizaremos el presente documento con las conclusiones de los resultados obtenidos y algunas líneas interesantes como trabajo futuro.

Conclusiones

Llama poderosamente la atención el hecho de que, con anterioridad al año 2002, se habían presentado diversas soluciones de problemas en el marco de *Membrane Computing* sin que, previamente, en dicha disciplina se hubiera definido, ni siquiera en un contexto semi-informal, el concepto de *resolución de un problema abstracto*. El desarrollo de una teoría de la complejidad computacional en el paradigma de la computación celular con membranas era una necesidad que debía, ser abordada de manera perentoria. Había que poner un poco de orden y de seriedad a la hora de manejar los modelos computacionales introducidos en el citado paradigma, a fin de proporcionar soluciones alternativas a problemas.

Así sucedió cuando uno de los codirectores de esta tesis se puso manos a la obra introduciendo, en primer lugar, los *sistemas P de decisión* [52], a principios de 2002, y los *sistemas P de aceptación* [53], poco después. La idea central de esos trabajos consistía en definir los conceptos básicos relacionados con la resolución de problemas de decisión en el marco de un paradigma que daba lugar a dispositivos computacionales no deterministas, denominados genéricamente *sistemas de membranas*. Teniendo presente que todo problema de decisión tiene asociado, de manera natural, un lenguaje formal, parecía aconsejable introducir unos sistemas de membranas específicos que tuvieran la habilidad de *reconocer lenguajes* (los sistemas P de decisión y los sistemas P de aceptación serían los precursores de los sistemas P reconocedores).

Ahora bien, a diferencia de lo que sucede con las máquinas de Turing, los sistemas de membranas tienen una descripción completamente finita, en lo que respecta a los ingredientes sintácticos básicos (alfabetos, multiconjunto de objetos, reglas de evolución, etc.) que en ella intervenían. De ahí que en la resolución de un problema abstracto debía intervenir, en general, una familia infinita y numerable de sistemas de membranas. Por otra parte, puesto que los dispositivos computacionales considerados eran de naturaleza no deterministas, era importante establecer, desde el principio, claras diferencias con la forma clásica de resolver problemas a través de máquinas de Turing no deterministas, pues se trataba de justificar que la disciplina de *Membrane Computing* ofrecía un nuevo marco para la resolución «eficiente» de problemas abstractos. En ese contexto, mientras que, por una parte, una máquina de Turing no determinista acepta una instancia de un problema si y sólo si existe al menos una computación de la máquina con entrada dicha instancia, que es de aceptación; por otra, un sistema de membranas acepta una instancia de un problema si y sólo si cualquier computación del sistema con entrada dicha instancia, es de aceptación.

Finalmente, era aconsejable tratar de *cuadrar el círculo* en el sentido de que la teoría de la complejidad computacional que se desarrollara debía dar cabida, incluir de alguna manera, las «soluciones» que habían sido proporcionadas previamente, en términos absolutamente informales.

En agosto de 2002, en el *Workshop on Membrane Computing* celebrado en Curtea de Arges, se presentó «en sociedad» la propuesta de teoria de la complejidad computacional en Membrane Computing que está «vigente» en la actualidad [52]. El correspondiente artículo fundacional sería publicado en el año 2003 [51]. En esa teoría se define formalmente el concepto de resolución eficiente y *de manera uniforme* de problemas abstractos de decisión a través de familias de sistemas de membranas, en donde los sistemas que procesan instancias de problemas han de ser *confluentes*, en el sentido de que o bien todas las computaciones son de aceptación, o bien todas las computaciones son de rechazo. Así mismo, se introdujo formalmente el concepto de *solución eficiente y semi-uniforme* de problemas abstractos, a fin de encontrar un encaje adecuado a las «soluciones» proporcionadas con anterioridad a 2002. Llegado este punto, conviene puntualizar una diferencia fundamental entre soluciones semi-uniformes y soluciones uniformes: en el caso de las soluciones semi-uniformes, cada instancia del problema tendrá asociada un sistema de membranas específico, mientras que en las soluciones uniformes, todas las instancias de un problema que tengan el «mismo tamaño» (respecto de una función computable en tiempo polinomial asociado al mismo) son procesadas por el mismo sistema

de membranas, si bien hay que añadir un multiconjunto de entrada específico para cada una de las diferentes instancias que tienen el mismo tamaño.

Desde sus inicios, la teoría de la complejidad computacional en el paradigma de la computación celular con membranas ha tenido una gran repercusión en el desarrollo de nuevos métodos para abordar el problema **P** *versus* **NP**. Conviene resaltar la gran variedad de técnicas que se han desarrollado a lo largo de los últimos años a fin de caracterizar las diferentes clases de complejidad estándar en términos de clases de complejidad computacional definidas en el paradigma antes citado.

En este trabajo se ha presentado, de una manera sistemática, una nueva metodología para abordar el problema **P** *versus* **NP** dentro del marco de un paradigma computacional arbitrario. La metodología se materializa, propiamente, a través de la búsqueda de fronteras entre la no eficiencia y la presumible eficiencia de modelos de computación. A lo largo de esta memoria, esa metodología ha sido aplicada al paradigma de la computación celular con membranas, tanto a sistemas de membranas que trabajan a modo de células como a sistemas de membranas que trabajan a modo de tejidos.

Para el establecimiento de la no eficiencia de sistemas de membranas se han utilizado, básicamente, tres técnicas diferentes: (a) la técnica del grafo de dependencia; (b) la técnica algorítmica; y (c) la técnica de la simulación eficiente. Para el establecimiento de la presumible eficiencia de sistemas de membranas, se han proporcionado soluciones eficientes (en tiempo polinomial) y uniformes de problemas **NP**-completos.

Se han obtenido fronteras entre la no eficiencia y la presumible eficiencia de los siguientes sistemas de membranas que trabajan a modo de células:

- Sistemas P de membranas activas, sin polarización y sin disolución tales que:
  - Permiten reglas de división o separación de membranas elementales y ciertos tipos de cooperación en reglas de evolución de objetos.
  - Permiten reglas de división o separación de membranas elementales y no elementales, así como cooperación minimal con producción minimal en reglas de comunicación.
- Sistemas P con reglas de comunicación (del tipo symport/antiport) pero con evolución, tales que:
  - Permiten reglas de división o separación de membranas elementales.

  - Permiten reglas de división o separación de membranas elementales pero en los que no se usan el entorno.

Así mismo, se han obtenido fronteras entre la no eficiencia y la presumible eficiencia de los siguientes sistemas de membranas que trabajan a modo de tejidos:

- Sistemas P de tejidos con reglas de comunicación (del tipo symport/antiport) pero con evolución, tales que:
  - Permiten reglas de división o separación de membranas elementales.
  - Permiten reglas de división o separación de membranas elementales pero en los que no se usan el entorno.

Analizando con detalle las soluciones presentadas para el establecimiento de la presumible eficiencia de ciertos sistemas de membranas y comparándolas con otras dadas con anterioridad, se ha observado la existencia de algunos ingredientes en esos sistemas que parecen ser *necesarios* para poder obtener soluciones eficientes y uniformes de problemas presuntamente intratables. Específicamente, los sistemas de membranas parecen compartir los siguientes elementos comunes:

1. Capacidad para generar de una cantidad de espacio exponencial, expresada en términos del número de objetos y del número de membranas/células, en tiempo polinomial (a través de unos mecanismos implementados por reglas de evolución para los objetos, o bien por reglas de división, separación o creación para las membranas/células).

2. Existencia de algún mecanismo adicional de cooperación o similar: polarizaciones, catalizadores, reglas de disolución o reglas cooperativas.

3. Existencia de algún tipo de contexto en términos de membranas/células: polarizaciones para la sincronización, ejecución de las reglas en la misma membrana/célula, o creación de membranas «diferenciadas».

Los mecanismos de sincronización que se han usado en las soluciones presentadas en esta memoria han permitido, por una parte, que algunas de ellas se materialicen a través de familias de sistemas deterministas, y, por otra, han propiciado que los correspondientes procesos de verificación formal, hayan sido más simples, más «llevaderos», siempre dentro del contexto de su enorme complejidad.

Trabajo futuro

La investigación desarrollada a lo largo de esta memoria ha proporcionado una serie de resultados que representan importantes mejoras respecto de otros ya establecidos previamente, a la vez que genera nuevo conocimiento y deja abierto algunos problemas que pueden ser interesantes de cara a futuras investigaciones. De hecho, algunas de estas líneas de trabajo ya han sido planteadas y están siendo afrontadas por miembros del Grupo de Investigación en Computación Natural de la Universidad de Sevilla. Entre esas líneas queremos destacar las siguientes:

1. Estudio de las clases de complejidad de sistemas que permiten el uso de reglas de creación de membranas, pero cuya semántica es derivada de la estándar asociada a los sistemas P con membranas activas.

2. Diseño de soluciones eficientes a problemas de conteo (*counting problems*) o a problemas numéricos en distintas variantes de sistemas de membranas.

3. Desarrollo de una teoría de la complejidad computacional en el marco de los sistemas de membranas que trabajan a modo de neuronas.

4. Establecimiento de cotas superiores de las clases de complejidad computacional asociadas a ciertas variantes de sistemas de membranas con el objetivo de caracterizarlas a través de clases de complejidad clásicas.

5. Estudio de otras medidas de complejidad en sistemas de membranas, como podrían ser el espacio o la comunicación.

6. Análisis de la complejidad descriptiva en el marco de la computación de membranas, de las soluciones de problemas **NP**-completos presentadas en este trabajo, usando técnicas derivadas de las *Sevilla Carpets*.

7. Desarrollo de nuevas técnicas para demostrar la *no eficiencia* de algunas variantes de sistemas de membranas.

8. Estandarización de la nomenclatura de clases de sistemas de membranas.

Bibliografía

[1] L. M. Adleman. Molecular Computation of Solutions to Combinatorial Problems. *Science*, **266**, 11 (1994), 1021–1024.

[2] A. Alhazov, R. Freund, Gh. Păun. P Systems with Active Membranes and Two Polarizations. En Gh. Păun, A. Riscos-Núñez, A. Romero-Jiménez, F. Sancho-Caparrini (eds.), *Proceedings of the Second Brainstorming Week on Membrane Computing*, Sevilla, Spain, February 2-7 2004. Fénix Editora. pp. 20–36.

[3] A. Alhazov, L. Pan, Gh. Păun. Trading polarizations for labels in P systems with active membranes. *Acta Informatica*, **41**, 2 (2004), 111–144.

[4] C. H. Bennett. Logical Reversibility of Computation. *IBM J. Res. Dev.*, **17**, 6 (1973), 525–532.

[5] C. H. Bennett. The thermodynamics of computation – a review. *International Journal of Theoretical Physics*, **21**, 12 (1982), 905–940.

[6] M. Blum. A Machine-Independent Theory of the Complexity of Recursive Functions. *J. ACM*, **14**, 2 (1967), 322–336.

[7] D. Boneh, C. Dunworth, R. J. Lipton. Breaking DES using a molecular computer. En *Proceedings of DIMACS workshop on DNA computing*, 1995. pp. 37–66.

[8] G. Boole. *The mathematical analysis of logic, being an essay towards a calculos of deductive reasoning.* Cambridge: Macmillan, Barclay and Macmillan, London: George Bell, 1847.

[9] A. Church. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, **58** (1936), 345–363.

[10] A. Cobham. The Intrinsic Computational Difficulty of Functions. En Y. Bar-Hillel (ed.), *Logic, Methodology and Philosophy of Science: Proceedings of the 1964 International Congress (Studies in Logic and the Foundations of Mathematics)*, pp. 24–30. North-Holland Publishing, 1965.

[11] S. A. Cook. The Complexity of Theorem-proving Procedures. En *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, New York, NY, USA, 1971. ACM. pp. 151–158.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.

[13] D. Díaz-Pernil, M. J. Pérez-Jiménez, Á. Romero-Jiménez. Efficient simulation of tissue-like P systems by transition cell-like P systems. *Natural Computing*, **8**, 4 (2008), 797.

[14] J. Edmonds. Minimum partition of a matroid into independent subsets. *Journal of Research of the National Bureau of Standards - B. Mathematics and Mathematical Physics*, **69**, 1-2 (2018), 229–241.

[15] R. P. Feynman. There's plenty of room at the bottom [data storage]. *Journal of Microelectromechanical Systems*, **1**, 1 (1992), 60–66.

[16] M. R. Garey, D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[17] M. A. Gutiérrez-Naranjo, M. J. Pérez-Jiménez, R.-N. Agustin, F. J. Romero-Campero, Á. Romero-Jiménez. Characterizing Tractability by Cell-Like Membrane Systems. En K. Subramanian, K. Rangarajan, M. Mukund (eds.), *Formal Models, Languages and Applications*, volume 66. World Scientific, Series in Machine Perception and Artificial Intelligence, 2007. pp. 137–154.

[18] M. A. Gutiérrez-Naranjo, M. J. Pérez-Jiménez, A. Riscos-Núñez, F. J. Romero-Campero. On the Power of Dissolution in P Systems with Active Membranes. En R. Freund, Gh. Păun, Gr. Rozenberg, A. Salomaa (eds.), *Membrane Computing*, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. pp. 224–240.

[19] R. W. Hamming. We Would Know What They Thought When They Did It. En N. Metropolis, J. Howlett, G.-C. Rota (eds.), *A History of*

Computing in the Twentieth Century, pp. 3–9. Academic Press, San Diego, 1980.

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

[21] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[22] R. Impagliazzo. A Personal View of Average-case Complexity. En *Proceedings of the 10th Annual Structure in Complexity Theory Conference (SCT'95)*, SCT '95, Washington, DC, USA, 1995. IEEE Computer Society. pp. 134–147.

[23] R. M. Karp. *Reducibility among Combinatorial Problems*, pp. 85–103. Springer US, Boston, MA, 1972.

[24] L. A. Levin. Average Case Complete Problems. *SIAM J. Comput.*, **15**, 1 (1986), 285–286.

[25] P. M. Lewis, R. E. Stearns, J. Hartmanis. Memory Bounds for Recognition of Context-free and Context-sensitive Languages. En *Proceedings of the 6th Annual Symposium on Switching Circuit Theory and Logical Design (SWCT 1965)*, FOCS '65, Washington, DC, USA, 1965. IEEE Computer Society. pp. 191–202.

[26] L. F. Macías-Ramos, M. J. Pérez-Jiménez, A. Riscos-Núñez, L. Valencia-Cabrera. Membrane fission versus cell division: When membrane proliferation is not enough. *Theoretical Computer Science*, **608** (2015), 57–65.

[27] L. F. Macías-Ramos, M. J. Pérez-Jiménez, A. Riscos-Núñez, M. Rius-Font, L. Valencia-Cabrera. The Efficiency of Tissue P Systems with Cell Separation Relies on the Environment. En E. Csuhaj-Varjú, M. Gheorghe, Gr. Rozenberg, A. Salomaa, G. Vaszil (eds.), *Membrane Computing*, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. pp. 243–256.

[28] L. F. Macías-Ramos, B. Song, T. Song, L. Pan, M. J. Pérez-Jiménez. Limits on Efficient Computation in P Systems with Symport/Antiport Rules. En C. Graciani, Gh. Păun, A. Riscos-Núñez, L. Valencia-Cabrera (eds.), *Proceedings of the Fifteenth Brainstorming Week on Membrane*

Computing, Sevilla, Spain, January 31 - February 3 2017. Fénix Editora. pp. 147–160.

[29] L. F. Macías-Ramos, B. Song, L. Valencia-Cabrera, L. Pan, M. J. Pérez-Jiménez. Membrane fission: A computational complexity perspective. *Complexity*, **21**, 6 (2016), 321–334.

[30] C. Martin-Vide, J. Pazos, Gh. Păun, A. Rodríguez-Patón. A New Class of Symbolic Abstract Neural Nets: Tissue P Systems. En *Proceedings of the 8th Annual International Conference on Computing and Combinatorics*, COCOON '02, London, UK, UK, 2002. Springer-Verlag. pp. 290–299.

[31] W. S. McCulloch, W. Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. En J. A. Anderson, E. Rosenfeld (eds.), *Neurocomputing: Foundations of Research*, pp. 15–27. MIT Press, Cambridge, MA, USA, 1988.

[32] M. L. Minsky, S. A. Papert. *Perceptrons: Expanded Edition*. MIT Press, Cambridge, MA, USA, 1988.

[33] D. Orellana-Martín. Efficiency Frontiers through Bio-inspired Computing Models. Master's thesis, Universidad de Sevilla, Universidad de Sevilla, julio 2016.

[34] D. Orellana-Martín, M. A. Martínez-del Amor, L. Valencia-Cabrera, B. Song, L. Pan, M. J. Pérez-Jiménez. P systems with symport/antiport rules: When do the surroundings matter? *Theoretical Computer Science*, (2018).

[35] D. Orellana-Martín, L. Valencia-Cabrera, A. Riscos-Núñez, M. J. Pérez-Jiménez. A path to computational efficiency through membrane computing. *Theoretical Computer Science*, (in press (2019)).

[36] L. Pan, T.-O. Ishdorj. P Systems with Active Membranes and Separation Rules. *Journal of Universal Computer Science*, **10**, 5 (2004), 630–649. http://www.jucs.org/jucs_10_5/p_systems_with_active.

[37] L. Pan, M. J. Pérez-Jiménez. Computational complexity of tissue-like P systems. *Journal of Complexity*, **26**, 3 (2010), 296–315.

[38] L. Pan, M. J. Pérez-Jiménez, A. Riscos-Núñez, M. Rius-Font. New frontiers of the efficiency in tissue P systems. En *Pre-Proceedings of the Asian*

Conference on Membrane Computing (ACMC 2012), Huazhong University of Science and Technology, Wuhan, China, October 15-18 2012. pp. 61–73.

[39] L. Pan, B. Song, L. Valencia-Cabrera, M. J. Pérez-Jiménez. The computational complexity of tissue P systems with evolutional symport/antiport rules. *Complexity*, **2018**, Article ID 3745210 (2018), 21.

[40] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[41] A. Păun, Gh. Păun. The power of communication: P systems with symport/antiport. *New Generation Computing*, **20**, 3 (2002), 295–305.

[42] A. Păun, Gh. Păun, Gr. Rozenberg. Computing by Communication in Networks of Membranes. *International Journal of Foundations of Computer Science*, **13**, 06 (2002), 779–798.

[43] Gh. Păun. Computing with Membranes. *Journal of Computer and System Sciences*, **61**, 1 (2000), 108 – 143.

[44] Gh. Păun. Further twenty six open problems in membrane computing. En M. A. Gutiérrez-Naranjo, A. Riscos-Nú nez, F. J. Romero-Campero, D. Sburlan (eds.), *Proceedings of the Third Brainstorming Week on Membrane Computing*, Sevilla, Spain, January 30 - February 4 2005. Fénix Editora. pp. 20–36.

[45] Gh. Păun, M. J. Pérez-Jiménez, A. Riscos-Núñez. Tissue P systems with cell division. *International Journal of Computers, Communications & Control*, **3**, 3 (2008), 295–303.

[46] M. J. Pérez-Jiménez. Máquinas en la Naturaleza Viva. En P. Fortet (ed.), *Ciencia, Tecnología y Humanidades en la era del conocimiento*. Editorial de la Universidad de Sevilla, en prensa, 2018.

[47] M. J. Pérez-Jiménez, A. Riscos-Núñez. A Linear-Time Solution to the Knapsack Problem Using P Systems with Active Membranes. En C. Martín-Vide, G. Mauri, Gh. Păun, Gr. Rozenberg, A. Salomaa (eds.), *Membrane Computing*, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. pp. 250–268.

[48] M. J. Pérez-Jiménez, A. Riscos-Núñez. Solving the Subset-Sum problem by P systems with active membranes. *New Generation Computing*, **23**, 4 (2005), 339–356.

[49] M. J. Pérez-Jiménez, A. Riscos-Núñez, M. Rius-Font, F. J. Romero-Campero. A polynomial alternative to unbounded environment for tissue P systems with cell division. *Int. J. Comput. Math.*, **90**, 4 (2013), 760–775.

[50] M. J. Pérez-Jiménez, F. J. Romero-Campero. Trading Polarization for Bi-stable Catalysts in P Systems with Active Membranes. En G. Mauri, Gh. Păun, M. J. Pérez-Jiménez, Gr. Rozenberg, A. Salomaa (eds.), *Membrane Computing*, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. pp. 373–388.

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

[52] M. J. Pérez Jiménez, A. Romero Jiménez, F. Sancho Caparrini. Decision P Systems and the P $\neq$ NP Conjecture. En Gh. Păun, Gr. Rozenberg, A. Salomaa, C. Zandron (eds.), *Membrane Computing*, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. pp. 388–399.

[53] M. J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini. A Polynomial Complexity Class in P Systems Using Membrane Division. *J. Autom. Lang. Comb.*, **11**, 4 (2006), 423–434.

[54] M. J. Pérez-Jiménez, P. Sosík. An Optimal Frontier of the Efficiency of Tissue P Systems with Cell Separation. *Fundamenta Informaticae*, **138**, 1-2 (2015), 45–60.

[55] A. E. Porreca, N. Murphy, M. J. Pérez-Jiménez. An optimal frontier of the efficiency of tissue P systems with cell division. En M. García-Quismondo, L. F. Macías-Ramos, Gh. Păun, I. Pérez-Hurtado (eds.), *Proceedings of the Tenth Brainstorming Week on Membrane Computing*, volume II, Sevilla, Spain, January 30 - February 3 2012. Fénix Editora. pp. 141–166.

[56] Gh. Păun. Computing with Membranes: Attacking NP-Complete Problems. En I. Antoniou, C. S. Calude, M. J. Dinneen (eds.), *Unconventional Models of Computation, UMC'2K*, London, 2001. Springer London. pp. 94–115.

[57] Gh. Păun. *Membrane Computing: An Introduction*. Springer-Verlag, Berlin, Heidelberg, 2002.

[58] Gh. Păun, Gr. Rozenberg, A. Salomaa. *The Oxford Handbook of Membrane Computing*. Oxford University Press, Inc., New York, NY, USA, 2010.

[59] F. Rosenblatt. The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain. *Psychological Review*, **65** (1958), 386–408.

[60] B. Song, C. Zhang, L. Pan. Tissue-like P systems with evolutional symport/antiport rules. *Information Sciences*, **378** (2017), 177–193.

[61] R. E. Stearns, J. Hartmanis, P. M. Lewis. Hierarchies of Memory Limited Computations. En *Proceedings of the 6th Annual Symposium on Switching Circuit Theory and Logical Design (SWCT 1965)*, FOCS '65, Washington, DC, USA, 1965. IEEE Computer Society. pp. 179–190.

[62] A. M. Turing. Computability and $Lambda$-Definability. *Journal of Symbolic Logic*, **2**, 4 (1937), 153–163.

[63] L. Valencia-Cabrera, D. Orellana-Martín, M. A. Martínez-del Amor, R.-N. nezm Agustín, M. J. Pérez-Jiménez. Polarizationless P Systems with Active Membranes: Computational Complexity Aspects. *Journal of Automata, Languages and Combinatorics*, **21**, 1–2 (2016), 107–123.

[64] L. Valencia-Cabrera, D. Orellana-Martín, M. A. Martínez-del Amor, A. Riscos-Núñez, M. J. Pérez-Jiménez. Cooperation in Transport of Chemical Substances: A Complexity Approach within Membrane Computing. *Fundam. Inform.*, **154**, 1-4 (2017), 373–385.

[65] L. Valencia-Cabrera, D. Orellana-Martín, M. A. Martínez-del Amor, A. Riscos-Núñez, M. J. Pérez-Jiménez. Reaching efficiency through collaboration in membrane systems: Dissolution, polarization and cooperation. *Theoretical Computer Science*, **701** (2017), 226 – 234. At the intersection of computer science with biology, chemistry and physics - In Memory of Solomon Marcus.

[66] L. Valencia-Cabrera, D. Orellana-Martín, A. Riscos-Núñez, M. J. Pérez-Jiménez. Minimal cooperation in polarizationless P systems with active membranes. En C. Graciani, Gh. Păun, D. Orellana-Martín, A. Riscos-Núñez, L. Valencia-Cabrera (eds.), *Proceedings of the Fourteenth Brainstorming Week on Membrane Computing*, Sevilla, Spain, February 1 - 5 2016. Fénix Editora. pp. 327–356.

[67] L. Valencia-Cabrera, B. Song, L. F. Macías-Ramos, L. Pan, M. J. Riscos-Núñez, Pérez-Jiménez. Computational efficiency of P systems with symport/antiport rules and membrane separation. En L. F. Macías-Ramos, Gh. Păun, A. Riscos-Núñez, L. Valencia-Cabrera (eds.), *Proceedings of the Thirteenth Brainstorming Week on Membrane Computing*, Sevilla, Spain, February 2-6 2015. Fénix Editora. pp. 301–323.

[68] L. Valencia-Cabrera, B. Song, L. F. Macías-Ramos, L. Pan, M. J. Riscos-Núñez, Pérez-Jiménez. Minimal cooperation in P systems with symport/antiport: A complexity appoach. En L. F. Macías-Ramos, Gh. Păun, A. Riscos-Núñez, L. Valencia-Cabrera (eds.), *Proceedings of the Thirteenth Brainstorming Week on Membrane Computing*, Sevilla, Spain, February 2-6 2015. Fénix Editora. pp. 301–323.

[69] J. von Neumann. A certain zero-sum two-person game equivalent to the optimal assignment problem. *Contributions to the Theory of Games II;Annals of Mathematics Studies*, **28** (1953), 5–12.

[70] C. Zandron, C. Ferretti, G. Mauri. Solving NP-Complete Problems Using P Systems with Active Membranes. En I. Antoniou, C. S. Calude, M. J. Dinneen (eds.), *Unconventional Models of Computation, UMC'2K*, London, 2001. Springer London. pp. 289–301.

[71] MeCoSim website. `http://www.p-lingua.org/mecosim/`.

[72] P-Lingua website. `http://www.p-lingua.org/wiki/index.php/Main_Page`.

[73] PMCGPU project website. `https://sourceforge.net/projects/pmcgpu/`.

[74] PPage website. `http://ppage.psystems.eu/`.