UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
MAESTRÍA EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

UNA RELACIÓN ENTRE LAS LÓGICAS MODALES Y EL ENFOQUE TOPOLÓGICO DEL CÓMPUTO DISTRIBUIDO

TESIS
QUE PARA OPTAR POR EL GRADO DE
MAESTRO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

PRESENTA:
DIEGO ALEJANDRO VELÁZQUEZ CERVANTES

TUTORES PRINCIPALES:
DR. ARMANDO CASTAÑEDA ROJANO
INSTITUTO DE MATEMÁTICAS
DR. DAVID ARTURO ROSENBLUETH LAGUETTE
INSTITUTO DE INVESTIGACIONES EN MATEMÁTICAS APLICADAS Y EN SISTEMAS

CIUDAD UNIVERSITARIA, CD. MX. NOVIEMBRE DE 2019

Dedicado a Concepción, Pedro, Edith, Bruno y Mónica.

Agradecimientos

En lo personal, agradezco a mi madre, porque su muerte puso en perspectiva mi vida e influyó en la decisión de estudiar un posgrado. Agradezco a mi padre que siempre me ha apoyado cuando lo he necesitado. Agradezco también a mi esposa, quien me apoyó cuando tomé la decisión de dejar mi trabajo para seguir con mis estudios, aun cuando sabíamos que nuestra vida cambiaría por completo. Agradezco a mis hijos, Bruno y Mónica, porque verlos crecer este tiempo ha sido un gran regalo a mi vida. Por último agradezco a Guillermo y Virginia que me apoyaron en el cuidado de mis hijos cuando no me era posible hacerlo.

En lo académico, agradezco al Dr. Favio Ezequiel Miranda Perea, la Dra. Elisa Viso Gurovich y la Dra. Amparo López Gaona por brindarme cartas de recomendación para el ingreso a la maestría. También agradezco a todos los profesores y alumnos del posgrado que contribuyeron a mi desarrollo académico y personal, especialmente al Dr. David Arturo Rosenblueth Laguette y al Dr. Armando Castañeda Rojano, quienes además de ser mis guías, se convirtieron en personas a las que tengo muy alta estima. También agradezco al Dr. Sergio Rajsbaum Gorodezky, el Dr. Francisco Hernández Quiroz y la Dra. Lourdes del Carmen González Huesca por el tiempo invertido a la revisión y los comentarios al trabajo.

Por último, y sin mermar importancia, agradezco a todos los mexicanos, ya que por medio del CONACyT me permitieron realizar mis estudios en la maestría en Ciencia e Ingeniería de la Computación. Sin la beca que me proporcionó CONACyT me hubiera sido imposible continuar con mis estudios.

¡Muchas gracias a todos!

Resumen

El objetivo de este trabajo es presentar una relación que existe entre el enfoque de las lógicas modales y el enfoque topológico del cómputo distribuido. Para lograr dicho objetivo primero estudiamos cómo las estructuras que se usan en cada enfoque pueden representar la misma información y mostramos una relación entre las estructuras que se usan en cada enfoque: los *modelos de Kripke* en el enfoque de las lógicas modales y los *complejos simpliciales* en el enfoque topológico del cómputo distribuido. Para formalizar dicha relación proponemos algoritmos para transformar conjuntos de simplejos de un complejo simplicial a modelos de Kripke y viceversa. Dichos algoritmos se presentan desde el enfoque de las lógicas modales proposicionales. Nos concentramos en estructuras que representan ejecuciones acotadas de un caso particular de protocolos en el modelo *Iterated Immediate Snapshots* en el que en cada iteración los procesos solamente escriben su entrada. Después discutimos una relación entre las formas en las que cada enfoque describe la evolución de sus estructuras: del lado de las lógicas modales, particularmente la lógica epistémica proposicional, proponemos los *modelos de acciones extendidos*, que son una lógica epistémica dinámica basada en los *modelos de acciones* y del lado del enfoque topológico analizamos los mapeos portadores. Una diferencia entre ambas formas de evolución es que los modelos de acciones extendidos trabajan sobre el conjunto total de agentes del modelo de Kripke mientras que los mapeos portadores se pueden definir para cualquier simpliejo de un complejo simplicial. Para poder analizar el comportamiento definido explícitamente en los mapeos portadores, en el enfoque de la lógica epistémica proposicional proponemos proyecciones de modelos de Kripke sobre un conjunto de agentes, que corresponde a los procesos representados en el complejo simplicial.

Índice general

Capítulo 1

Introducción

El objetivo de este trabajo es mostrar una relación entre el enfoque de las lógicas modales y el enfoque topológico del cómputo distribuido. Mostramos cómo los modelos de Kripke, utilizados en el enfoque de las lógicas modales, y los complejos simpliciales, usados en el enfoque topológico del cómputo distribuido, pueden representar ejecuciones acotadas de protocolos de cómputo distribuido. Diseñamos algoritmos, desde el punto de vista de las lógicas modales, para transformar las estructuras que se usan en un enfoque a las que se usan en el otro. Además, discutimos una relación y una diferencia entre los mapeos portadores, la forma en la que el enfoque topológico describe la evolución de sus estructuras, y los modelos de acciones extendidos, una lógica epistémica dinámica que proponemos para describir la evolución del conocimiento.

En este capítulo damos un esbozo general al trabajo. En la sección 1.1 introducimos algunos conceptos básicos de cómputo distribuido haciendo analogías de conceptos en cómputo secuencial. En la sección 1.2 mostramos representaciones gráficas de las estructuras que usamos en el trabajo desde el punto de vista de cada enfoque. En al sección 1.3 discutimos la diferencia que hay entre las formas de describir la evolución de las estructuras en cada enfoque. En la sección 1.4 mencionamos algunos trabajos relacionados con el nuestro. Finalmente, en la sección 1.5 describimos la organización del resto de la tesis.

1.1. Analogías

En cómputo secuencial tenemos distintos modelos para resolver los problemas, como las máquinas de Turing, o el modelo RAM. En cómputo distribuido también existen diferentes modelos. Los modelos de cómputo distribuido definen, entre otras cosas, el tipo de sincronía entre los procesos y la forma en que se comunican. En esta tesis trabajamos con el modelo *Iterated Immediate Snapshots*. Dicho modelo es iterativo y asíncrono. En cada iteración, los procesos concurrentes primero escriben en un lugar específico para cada proceso e iteración en una memoria compartida, y después de que todos escriben, leen lo que se ha escrito hasta ese momento y en esa iteración.

En cómputo secuencial hablamos de funciones que podemos calcular con una máquina de Turing, partiendo de una entrada particular y resultando en una salida específica. En cómputo distribuido, llamamos tarea a la idea análoga de una función en cómputo secuencial. La entrada para una tarea está distribuida entre los diferentes procesos, y su salida también, pues cada proceso calcula una parte de ella.

Así como en cómputo secuencial tenemos algoritmos para calcular una función, en cómputo distribuido tenemos protocolos para resolver una tarea. Los protocolos son las instrucciones secuenciales que ejecuta cada proceso para calcular su parte de la salida. Usualmente se estudian protocolos de *información completa*, donde los procesos escriben toda la historia de lecturas en cada iteración. En este trabajo, estudiamos un caso particular de protocolos en el modelo de cómputo *Iterated Immediate Snapshots*. En nuestros protocolos de estudio, los procesos escriben únicamente su entrada en cada iteración. Elegimos ese tipo de protocolos por simplicidad, pero los resultados pueden extenderse para manejar protocolos de información completa. Asumimos que cada proceso guarda localmente la historia de las lecturas que ha hecho a lo largo de toda la ejecución.

1.2. Estructuras

En cuanto a la relación entre las estructuras que cada uno de los enfoques usa para representar información, nos interesan representaciones de las ejecuciones acotadas de nuestros protocolos de estudio. Para dichas representaciones, en el enfoque de las lógicas modales usamos *modelos de Kripke* y en el enfoque topológico del cómputo distribuido usamos *complejos simpliciales*.

Los modelos de Kripke son estructuras que representan posibles escenarios de la situación que representamos como *mundos*. Los mundos están relacionados desde el punto de vista de un individuo siempre que dos escenarios sean indistinguibles para él. Llamamos *agente* a cada individuo involucrado en la representación. En nuestro trabajo, los mundos de los modelos de Kripke representan ejecuciones acotadas de nuestros protocolos de estudio, y los agentes corresponden a los procesos involucrados en el protocolo.

Los complejos simpliciales son conjuntos de simplejos cerrados bajo contención. Un simplejo es un conjunto de vértices. En este trabajo los vértices representan estados posibles de los procesos. Particularmente, los vértices de los complejos simpliciales que analizamos se pueden identificar por su color, que asociamos a un proceso, y su vista, que representa lo que el proceso ha leído.

Tomemos en cuenta las posibles ejecuciones de una iteración para dos procesos, a y b, con entrada 0 de nuestros protocolos de estudio. Representamos gráficamente dichas ejecuciones como modelo de Kripke en la figura 1.1 y como complejo simplicial en la figura 1.2. En el modelo de Kripke etiquetamos los mundos con el estado de la memoria como una matriz de 2×2. El primer renglón corresponde a las entradas, y el segundo corresponde a lo que los procesos escriben. En el complejo simplicial los vértices de color gris oscuro representan al proceso a y los de color gris claro al proceso b. Representamos la vista en un arreglo dividido en dos renglones y dos columnas. El primer renglón corresponde a las entradas de los procesos y el segundo a lo que los procesos leyeron. La ausencia de información se denota con '_'. Observemos que cada proceso solo tiene información de su entrada en el primer renglón.

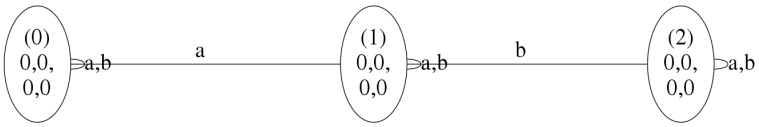

Figura 1.1: Modelo de Kripke de las ejecuciones de una iteración para dos procesos con entrada 0 en nuestros protocolos de estudio.

El hecho de poder representar la misma información en ambas estructuras nos lleva a pensar que puede haber un procedimiento para transformar complejos simpliciales en modelos de Kripke y viceversa. Trataremos esas transformaciones y algunas precisiones de las mismas en el capítulo 3.

Figura 1.2: Complejo simplicial de las ejecuciones de una iteración para dos procesos con entrada 0 en nuestros protocolos de estudio.

1.3. Evolución de las estructuras

En cuanto a la relación entre las formas de describir la evolución de las estructuras, analizamos los modelos de acciones extendidos del lado de las lógicas epistémicas dinámicas y los mapeos portadores, del lado del enfoque topológico del cómputo distribuido.

Los modelos de acciones extendidos son una lógica epistémica dinámica que describe las posibles acciones que se pueden llevar a cabo, su relación de indistinguibilidad desde el punto de vista de cada agente, las condiciones que se deben de cumplir para que puedan ocurrir y la información que se obtiene en la ocurrencia de cada acción. Los mapeos portadores son una función que asocia simplejos de un complejo simplicial a subcomplejos de otro complejo simplicial.

La figura 1.3 muestra gráficamente un modelo de acciones extendido donde las acciones corresponden a las posibles ejecuciones en nuestros protocolos de estudio para dos procesos con entrada 0. En dicha figura, representamos la información que los procesos a y b escriben con el etiquetamiento de las acciones, es decir $0, 0$. La función de precondición no está descrita en esa figura. Cuando hablemos de modelos de acciones extendidos, precisaremos la función en cada caso. Por ejemplo, podríamos decir que la función de precondición es la constante $\top$. Es decir, que las acciones no tienen ninguna restricción.

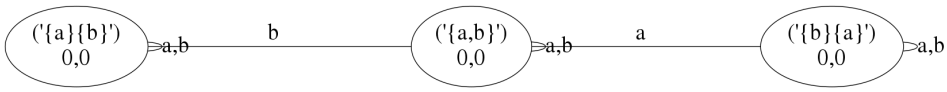

Figura 1.3: Modelo de acciones extendido.

Encontramos una diferencia en la manera en que ambas perspectivas describen la evolución de sus estructuras. Los mapeos portadores definen el comportamiento de un conjunto de procesos que depende del simplejo. Por

ejemplo, la definición del mapeo portador para un vértice puede describir el comportamiento del proceso que representa, para el caso que el proceso ejecute la iteración del protocolo primero. La definición del mapeo portador para una arista puede describir el comportamiento de los dos procesos que representa, para el caso que ambos ejecuten la iteración del protocolo primero. De esa forma tenemos una descripción fina del comportamiento de cada conjunto de procesos. Los modelos de acciones extendidos, al ser una lógica epistémica dinámica, se concentran solamente en el conjunto total de agentes, equivalente a todos los procesos que intervienen en el protocolo, de tal forma que en este enfoque tenemos una descripción gruesa del comportamiento de los agentes. Resolvemos esta situación parcialmente calculando proyecciones al modelo de Kripke sobre un conjunto de agentes, obteniendo las estructuras descritas por la definición de los mapeos portadores en simplejos que no representan la totalidad de los procesos.

1.4. Trabajo relacionado

Este trabajo tiene una relación particularmente estrecha con [5] y [8]. En [5] se proponen funtores para demostrar que las subcategorías completas de complejos simpliciales puros y cromáticos con colores en un conjunto finito A y la de marcos de Kripke propios sobre el conjunto de agentes A son categorías equivalentes. En [8] se definen los *modelos simpliciales*, que son complejos simpliciales a cuyos vértices se les asocia un conjunto de átomos proposicionales. En dicho trabajo, también se extiende la definición de los funtores propuestos en [5], para demostrar la equivalencia de la categoría de los modelos simpliciales y la categoría de modelos de Kripke propios. Además se proponen los *modelos de acciones simpliciales* que adecúan los *modelos de acciones* para su uso con modelos simpliciales.

Uno de los objetivos de los autores de [8] es estudiar qué tareas se pueden o no resolver en distintos modelos de cómputo. En [6], un trabajo posterior de los mismos autores, se utilizan los modelos simpliciales y los modelos de acciones simpliciales para estudiar la insolubilidad de la tarea de negación de igualdad, definida en [11], desde el punto de vista de la lógica. En [7], los mismos autores proponen una generalización de la tarea mencionada y se analiza su insolubilidad desde el punto de vista de topología combinatoria.

A diferencia de [5] y [8], donde los funtores toman en cuenta solamente las facetas del complejo simplicial, nosotros creamos algoritmos de transfor-

mación desde el enfoque de las lógicas modales que trabajan sobre cualquier subconjunto de simplejos para construir el marco de Kripke correspondiente. Además, tomamos en cuenta las vistas de los vértices en los simplejos para generar un etiquetamiento de los mundos del marco de Kripke dando lugar a un modelo de Kripke en el que podemos verificar propiedades representadas en fórmulas de lógica modal.

Un enfoque para representar la evolución del conocimiento es el de las lógicas epistémicas dinámicas. En [3] se recopilan algunas lógicas epistémicas dinámicas, entre ellas los *modelos de acciones*, presentados originalmente en [1]. En este trabajo y el de los autores de [5] y [8] tratamos la evolución de las estructuras desde dicho enfoque.

Otro enfoque para representar la evolución del conocimiento se estudia en [4]. En dicho trabajo, los autores presentan una extensión de la lógica epistémica que enriquece el lenguaje agregando operadores temporales. Dichos operadores tienen la finalidad de describir las ejecuciones de un protocolo a lo largo del tiempo, logrando razonar sobre el conocimiento que los procesos ganan en la ejecución de un protocolo. En ese trabajo se discute además la necesidad de conocimiento común para llegar a acuerdos, por ejemplo en el problema del consenso en el que un conjunto de procesos con entradas posiblemente distintas deben tener como salida la entrada de uno y solo uno de ellos.

Desde nuestro punto de vista el enfoque de las lógicas epistémicas dinámicas es más adecuado a la relación que queremos mostrar, ya que en dicho enfoque se actualizan los modelos de Kripke a lo largo del tiempo y de esa forma podemos cambiar de enfoque transformando las estructuras en cualquier punto de la ejecución de los protocolos.

1.5. Organización de la tesis

Organizamos el resto del documento como sigue. En el capítulo 2 presentamos formalmente los conceptos tanto de lógicas modales como de cómputo distribuido necesarias para entender el trabajo. En el capítulo 3 proponemos las transformaciones entre conjuntos de simplejos y modelos de Kripke que representan ejecuciones acotadas de nuestros protocolos de estudio. Además, describimos las características que deben tener los conjuntos de simplejos para inducir modelos de Kripke epistémicos y algunas observaciones más respecto a las estructuras. En el capítulo 4 discutimos la relación entre los

mapeos portadores y los modelos de acciones extendidos. Además, mencionamos una diferencia entre ambos y la forma en la que la subsanamos. En el capítulo 5 presentamos nuestras conclusiones y trabajo a futuro. Por último, en el apéndice A presentamos una versión literaria de la implementación que realizamos en el trabajo.

Capítulo 2

Fundamentos

En este capítulo presentamos, en dos secciones, los fundamentos para el entendimiento de este trabajo. En la primera sección tratamos temas relacionados con las lógicas modales. En la segunda sección presentamos contenidos vinculados con el enfoque topológico del cómputo distribuido.

2.1. Lógicas modales

En esta sección tratamos algunos conceptos relativos a las lógicas modales: una primera vista a las lógicas modales, las lógicas multimodales, la lógica epistémica y la lógica epistémica multiagente. Luego presentamos los modelos de Kripke que son estructuras que usaremos para representar modelos de lógica modal. Luego tocamos el concepto de bisimulación, que se puede ver como una forma de comparación entre modelos de Kripke. Por último presentamos una forma de tratar la lógica epistémica dinámica, los modelos de acciones, que extendemos en este trabajo para cubrir la necesidad de ampliar el conjunto de proposiciones con el que trabajamos.

Recuperamos el contenido en las subsecciones 2.1.1, 2.1.2, 2.1.3 y 2.1.4 de [10], el de la subsección 2.1.5 de [12] y [13] y el de la sección 2.1.6 de [3].

2.1.1. Lógica modal básica

En la lógica proposicional y la lógica de predicados, las fórmulas solamente pueden ser verdaderas o falsas. Sin embargo, en ocasiones es necesario distinguir entre distintos modos de verdad. Podemos necesitar modelar tiem-

po, lo que podría causar que una fórmula se cumpla en un momento y en otro instante no. Por ejemplo, en algún momento el que un vaso esté sobre una mesa puede ser cierto pero si alguien lo mueve de la mesa al lavavajilla ya no lo es. Podemos necesitar representar la noción de creencias, como que aunque la raíz cúbica del número 27 es 3, una persona pueda creer que no lo es. Otra noción de verdad importante es la noción de conocimiento, que da lugar a la lógica epistémica que exploraremos más adelante.

Para poder modelar distintos modos de verdad hacemos uso de las lógicas modales. Las lógicas modales se pueden ver como una extensión de alguna lógica. Particularmente, en este trabajo nos concentramos en las lógicas modales proposicionales, a las que nos referiremos simplemente como lógicas modales pues no trabajaremos con lógicas modales de mayor orden.

La lógica modal básica es la base de la cual podemos partir para definir lógicas modales más especializadas, como la lógica epistémica. A continuación presentamos la sintaxis y semántica de la lógica modal básica.

La lógica modal básica es una extensión de la lógica proposicional que agrega dos conectivos: el $\Box$ y el $\Diamond$. Ambos conectivos son unarios, de tal forma que se pueden aplicar a cualquier fórmula de lógica modal básica.

La sintaxis de la lógica modal básica está definida de la siguiente forma:

$$\phi ::= \bot \mid \top \mid p \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \rightarrow \phi) \mid (\phi \leftrightarrow \phi) \mid (\Box\phi) \mid (\Diamond\phi)$$

En lógica modal básica, el $\Box$ se lee como cuadro y el $\Diamond$ como rombo. Cuando se aplican lógicas modales a las distintas nociones de verdad se pueden leer de forma apropiada. Por ejemplo, en la lógica epistémica el $\Box$ se lee como *el agente sabe*, mientras que el $\Diamond$ se lee como *es consistente con el conocimiento del agente*.

Debemos diferenciar entre los distintos modos de verdad por lo que para las lógicas modales introducimos el concepto de *modelo de Kripke*.

Def. Un *modelo de Kripke* sobre una lógica $\mathcal{L}$ bajo las proposiciones P se define como una estructura $\mathcal{M} = \langle W, R, L\rangle$ donde W es un conjunto de mundos, R es la relación de accesibilidad entre los mundos y $L : W \longrightarrow 2^P$ es una función que relaciona a cada mundo las proposiciones ciertas en el mismo.

Intuitivamente cada $w \in W$ es un mundo posible y si $(w, w') \in R, w, w' \in W$ se dice que el mundo w' es accesible desde el mundo w. Podemos representar los modelos de Kripke de forma gráfica. Por ejemplo, si tenemos:

$$W = \{x_1, x_2, x_3, x_4, x_5, x_6\}$$

$$R = \{(x_1, x_2), (x_1, x_3), (x_2, x_2), (x_2, x_3), (x_3, x_2), (x_4, x_5), (x_5, x_4), (x_5, x_6)\}$$

$$L = \{(x_1, \{q\}), (x_2, \{p, q\}), (x_3, \{p\}), (x_4, \{q\}), (x_5, \emptyset), (x_6.\{p\}\}$$

La figura 2.1 muestra una representación del modelo de Kripke $M = \langle W, R, L\rangle$ sobre el conjunto de proposiciones $\{p, q\}$.

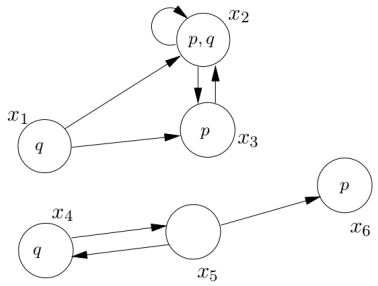

Figura 2.1: $\mathcal{M}$, un modelo de Kripke.[10]

La semántica se da por la relación de satisfacción que se define en seguida:

Def. Sea $M = \langle W, R, L\rangle$ un modelo de Kripke. Supongamos que $x \in W$ y que ϕ y ψ son fórmulas de lógica modal básica. Definimos cuándo una fórmula ϕ es cierta en el mundo x por inducción estructural sobre ϕ:

$\mathcal{M}, x \vDash \top$
$\mathcal{M}, x \nvDash \bot$
$\mathcal{M}, x \vDash p \ sii \ p \in L(x)$
$\mathcal{M}, x \vDash \neg\phi \ sii \ \mathcal{M}, x \nvDash \phi$
$\mathcal{M}, x \vDash \phi \wedge \psi \ sii \ \mathcal{M}, x \vDash \phi \ y \ \mathcal{M}, x \vDash \psi$
$\mathcal{M}, x \vDash \phi \vee \psi \ sii \ \mathcal{M}, x \vDash \phi \ o \ \mathcal{M}, x \vDash \psi$
$\mathcal{M}, x \vDash \phi \rightarrow \psi \ sii \ \mathcal{M}, x \nvDash \phi \ o \ \mathcal{M}, x \vDash \psi$
$\mathcal{M}, x \vDash \phi \leftrightarrow \psi \ sii \ \mathcal{M}, x \vDash \phi \rightarrow \psi \ y \ \mathcal{M}, x \vDash \psi \rightarrow \phi$
$\mathcal{M}, x \vDash \Box\phi \ sii \ \forall y \in W, \ (x, y) \in R, \ \mathcal{M}, y \vDash \phi$
$\mathcal{M}, x \vDash \Diamond\phi \ sii \ \exists y \in W$ tal que $(x, y) \in R, \ \mathcal{M}, x \vDash \phi$

Def. Sea $M = \langle W, R, L\rangle$ un modelo de Kripke y ϕ una fórmula de lógica modal básica. Decimos que $\mathcal{M}$ satisface ϕ si y solo si para todo $x \in W$ $\mathcal{M}, x \vDash \phi$.

2.1.2. Propiedades de la relación de accesibilidad

La gramática para lógica modal básica especifica fórmulas de lógica modal básica. En ocasiones es útil hablar de familias de fórmulas que comparten la misma "forma". A dichas familias se les conoce como *esquemas de fórmulas*. Por ejemplo, $\Box(\phi) \to \phi$ es un esquema de fórmulas. A las fórmulas de lógica proposicional con la forma del esquema de fórmulas se les llama ejemplares. Por ejemplo, las fórmulas $\Box(p) \to p$, y $\Box(p \wedge q) \to p \wedge q$ son ejemplares del esquema $\Box(\phi) \to \phi$. El esquema **K** correspondiente a
$\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi)$ es un esquema importante y nos referiremos a él más adelante.

Recordemos que las relaciones de accesibilidad pueden tener distintas propiedades. En seguida mencionamos algunas de nuestro particular interés. R es *reflexiva* si por cada $x \in W$, $(x, x) \in R$. R es *simétrica* si por cada $x, y \in W$, $(x, y) \in R$ implica $(y, x) \in R$. R es transitiva si por cada $x, y, z \in W$, $(x, y), (y, z) \in R$ implica $(x, z) \in R$. R es *euclidiana* si para cada $x, y, z \in W$, $(x, y), (x, z) \in R$ implica $(y, z) \in R$. Por último, R es una *relación de equivalencia* si R es reflexiva, simétrica y transitiva.

Las propiedades de las relaciones de accesibilidad se pueden relacionar con esquemas de fórmulas. En seguida damos la relación entre propiedades de nuestro interés y los esquemas de fórmulas que le corresponden. Podemos encontrar una tabla más amplia al respecto en [10].

Nombre	Esquema de fórmula	Propiedad
T	$\Box\phi \to \phi$	Reflexiva
B	$\phi \to \Box\Diamond\phi$	Simétrica
4	$\Box\phi \to \Box\Box\phi$	Transitiva
5	$\Diamond\phi \to \Box\Diamond\phi$	Euclidiana

2.1.3. Lógica epistémica

La lógica epistémica, también llamada **KT45** o **S5**, cumple con los esquemas **K**, **T**, **4** y **5**. Esta lógica es usada para razonar sobre el conocimiento. En dicha lógica, la fórmula $\Box\phi$ significa que el agente Q[1] sabe que la fórmula ϕ es cierta. El esquema **K** nos dice que la lógica es cerrada bajo consecuencia lógica. El esquema **T** nos indica que el agente Q sólo sabe cosas ciertas. El esquema **4** nos indica que el agente Q tiene introspección positiva, es decir que si sabe algo, sabe que lo sabe. Y por último, el esquema **5** nos indica que

[1] En KT45 solo hay un agente, el agente Q.

el agente tiene introspección negativa, es decir que si no sabe algo, sabe que no lo sabe.

La semántica de la lógica epistémica debe considerar relaciones de accesibilidad reflexivas, transitivas y euclidianas. Lo anterior es equivalente a que la relación sea reflexiva, simétrica y transitiva, es decir que sea una relación de equivalencia.

2.1.4. Lógica epistémica multiagente

En un sistema multiagente los agentes tienen diferente conocimiento del mundo. Los agentes pueden razonar sobre su propio conocimiento o el de otros agentes. Tomemos en cuenta el problema de los niños lodosos expuesto a continuación.

Hay un grupo de n ***niños jugando en el jardín y un número*** $0 \leq k \geq n$ ***de ellos se pueden ensuciar con lodo su frente. Cada niño puede ver la frente de los demás pero no la suya. Su padre los llama y anuncia que al menos uno de ellos está sucio y repetidamente pregunta: ¿Alguno sabe si tiene lodo en su frente?. Los primeras*** $k-1$ ***veces que el padre les pregunta todos contestan "no", pero a la*** k***-ésima pregunta de su padre los niños con la frente sucia pueden contestar "sí".***

Analicemos el caso en el que hay tres niños y uno de ellos tiene la frente sucia. Como el padre les dijo que alguno de ellos tiene la frente sucia, los dos niños que no tienen lodo en sus frentes ven al que sí tiene y pueden pensar que no tienen la frente sucia. Por el contrario, el niño que tiene la frente sucia ve que los demás no tienen la frente sucia, y como se supone que alguno de ellos tiene la frente sucia no queda más que él tenga la frente sucia, por lo que a la primera pregunta él responde "sí".

Ahora pensemos en el caso donde dos de ellos tienen la frente sucia. El niño con la frente limpia ve a los otros dos con la frente sucia, por lo que aún es posible que él tenga la frente limpia y responde "no". Los dos niños con la frente sucia ven al otro con la frente sucia, por lo que ambos piensan que es posible tener la frente limpia y responden "no". La segunda vez que el padre pregunta, el niño con la frente limpia razona de forma similar, sin embargo, los niños con la frente sucia, al ver que el otro niño con la frente sucia respondió "no" en la pregunta pasada y al otro con la frente limpia llegan a la conclusión de que ellos tienen la frente sucia, de tal forma que responden "sí".

Para modelar este tipo de escenarios necesitamos generalizar la lógica KT45. Consideremos un conjunto de agentes A. En lugar de tener solo un símbolo $\Box$ tendremos uno por cada agente $a \in A$ y escribiremos los conectores modales como K_a[2]. Por ejemplo, si tenemos a p como una proposición del conjunto de átomos proposicionales y $a \in A$ la fórmula $K_a p$ significa que el agente a sabe p. En términos más precisos, $K_a\phi$ significa que en todos los escenarios posibles desde el punto de vista del agente a, la fórmula ϕ se cumple.

Adicionalmente se agregan los conectivos modales E_G, C_G y D_G, donde $G \subseteq A$. El conectivo E_G es para decir que todos los agentes en G saben algo. Por ejemplo, la fórmula $E_G p$ significa que $\forall a \in G,\ K_a p$. El conectivo C_G representa conocimiento común en G[3]. Por último, el conectivo D_G representa conocimiento distribuido entre los agentes en el conjunto G.

Sea A el conjunto de agentes, $G \subseteq A$, *Props* el conjunto de proposiciones y $p \in Props$ algún átomo proposicional. La sintaxis para generar fórmulas de lógica epistémica multiagente es:

$$\phi ::= \bot \mid \top \mid p \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \to \phi) \mid (\phi \leftrightarrow \phi) \mid (K_a\phi) \mid (E_G\phi) \mid (C_G\phi) \mid (D_G\phi)$$

Redefinimos los modelos de Kripke para lógica modal multiagente de la siguiente forma:

Def. Un modelo de Kripke $\mathcal{M} = \langle W, R, L\rangle$, sobre el conjunto de agentes A y el conjunto de proposiciones $Props$, es una estructura que consta de W el conjunto de mundos en el modelo, $R = \bigcup R_a,\ a \in A \subseteq (W \times 2^A \times W)$ las relaciones de accesibilidad de cada agente y $L : W \longrightarrow 2^{Props}$ la función de etiquetamiento de los mundos.

Def. Sea $M = \langle W, R, L\rangle$ un modelo de Kripke sobre el conjunto de agentes A. Supongamos que $x \in W$, que $G \subseteq A$ y que ϕ y ψ son fórmulas de lógica epistémica multiagente. Definimos cuándo una fórmula ϕ es cierta en el mundo x por inducción estructural sobre ϕ:

$\mathcal{M}, x \vDash \top$

[2]La K es por *knowledge*.

[3]El conocimiento común se puede ver como una conjunción infinita de $E_G^i\phi$, es decir, $C_G\phi = E_G\phi \wedge E_G E_G\phi \wedge E_G E_G E_G\phi \wedge \ldots$.

$\mathcal{M}, x \nvDash \bot$
$\mathcal{M}, x \vDash p$ *sii* $p \in L(x)$
$\mathcal{M}, x \vDash \neg\phi$ *sii* $\mathcal{M}, x \nvDash \phi$
$\mathcal{M}, x \vDash \phi \wedge \psi$ *sii* $\mathcal{M}, x \vDash \phi$ *y* $\mathcal{M}, x \vDash \psi$
$\mathcal{M}, x \vDash \phi \vee \psi$ *sii* $\mathcal{M}, x \vDash \phi$ *o* $\mathcal{M}, x \vDash \psi$
$\mathcal{M}, x \vDash \phi \rightarrow \psi$ *sii* $\mathcal{M}, x \nvDash \phi$ *o* $\mathcal{M}, x \vDash \psi$
$\mathcal{M}, x \vDash \phi \leftrightarrow \psi$ *sii* $\mathcal{M}, x \vDash \phi \rightarrow \psi$ *y* $\mathcal{M}, x \vDash \psi \rightarrow \phi$
$\mathcal{M}, x \vDash K_a\phi$ *sii* $\forall\ (x, Ags, y) \in R$, *si* $a \in Ags$ *entonces* $\mathcal{M}, y \vDash \phi$
$\mathcal{M}, x \vDash E_G\phi$ *sii* $\forall a \in G,\ \mathcal{M}, x \vDash K_a\phi$
$\mathcal{M}, x \vDash C_G\phi$ *sii* $\forall k \in \mathbb{Z}^+,\ \mathcal{M}, x \vDash E_G^k\phi$
donde E_G^k *significa* $E_G E_G \ldots E_G$ *k veces.*
$\mathcal{M}, x \vDash D_G\phi$ *sii* $\forall\ (x, Ags, y) \in R$, *si* $E \subseteq Ags$ *entonces* $\mathcal{M}, y \vDash \phi$

A lo largo de este trabajo representamos los modelos de Kripke para lógicas modales multiagente con gráficas cuyos mundos tienen su identificador entre paréntesis, y su etiquetamiento representado como un estado de memoria. La relación de accesibilidad se representa como aristas etiquetadas por agentes. Por ejemplo, pensemos en que queremos razonar sobre los valores que tiene una matriz de 2×2 cuyos átomos proposicionales sean de la forma $p_{r,c,v}$ con r el renglón, c la columna y v su valor en la matriz. El modelo de Kripke $\mathcal{M} = \langle W, R, L\rangle$ donde

$W = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$

$R = \{(0, \{ab\}, 0), (0, \{a\}, 1), (0, \{b\}, 6), (1, \{a\}, 0), (1, \{a, b\}, 1), (1, \{b\}, 2),$
$(2, \{b\}, 1), (2, \{a, b\}, 2), (2, \{a\}, 5), (3, \{a, b\}, 3), (3, \{a\}, 4), (3, \{b\}, 9),$
$(4, \{a\}, 3), (4, \{a, b\}, 4), (4, \{b\}, 5), (5, \{a\}, 2), (5, \{b\}, 4), (5, \{a, b\}, 5),$
$(6, \{b\}, 0), (6, \{a, b\}, 6), (6, \{a\}, 7), (7, \{a\}, 6), (7, \{a, b\}, 7), (7, \{b\}, 8),$
$(8, \{a, b\}, 8), (8, \{a\}, 11), (9, \{b\}, 3), (9, \{a, b\}, 9), (9, \{a\}, 10), (10, \{a\}, 9),$
$(10, \{a, b\}, 10), (10, \{b\}, 11), (11, \{a\}, 8), (11, \{b\}, 10), (11, \{a, b\}, 11)\}$

y

$L = \{(0, \{p_{0,0,1}, p_{0,1,1}, p_{1,0,1}, p_{1,1,1}\}), (1, \{p_{0,0,1}, p_{0,1,1}, p_{1,0,1}, p_{1,1,1}\}),$
$(2, \{p_{0,0,1}, p_{0,1,1}, p_{1,0,1}, p_{1,1,1}\}), (3, \{p_{0,0,1}, p_{0,1,0}, p_{1,0,1}, p_{1,1,0}\}),$
$(4, \{p_{0,0,1}, p_{0,1,0}, p_{1,0,1}, p_{1,1,0}\}), (5, \{p_{0,0,1}, p_{0,1,0}, p_{1,0,1}, p_{1,1,0}\}),$
$(6, \{p_{0,0,0}, p_{0,1,1}, p_{1,0,0}, p_{1,1,1}\}), (7, \{p_{0,0,0}, p_{0,1,1}, p_{1,0,0}, p_{1,1,1}\}),$
$(8, \{p_{0,0,0}, p_{0,1,1}, p_{1,0,0}, p_{1,1,1}\}), (9, \{p_{0,0,0}, p_{0,1,0}, p_{1,0,0}, p_{1,1,0}\}),$
$(10, \{p_{0,0,0}, p_{0,1,0}, p_{1,0,0}, p_{1,1,0}\}), (11, \{p_{0,0,0}, p_{0,1,0}, p_{1,0,0}, p_{1,1,0}\})\}$

se muestra en la figura 2.2.

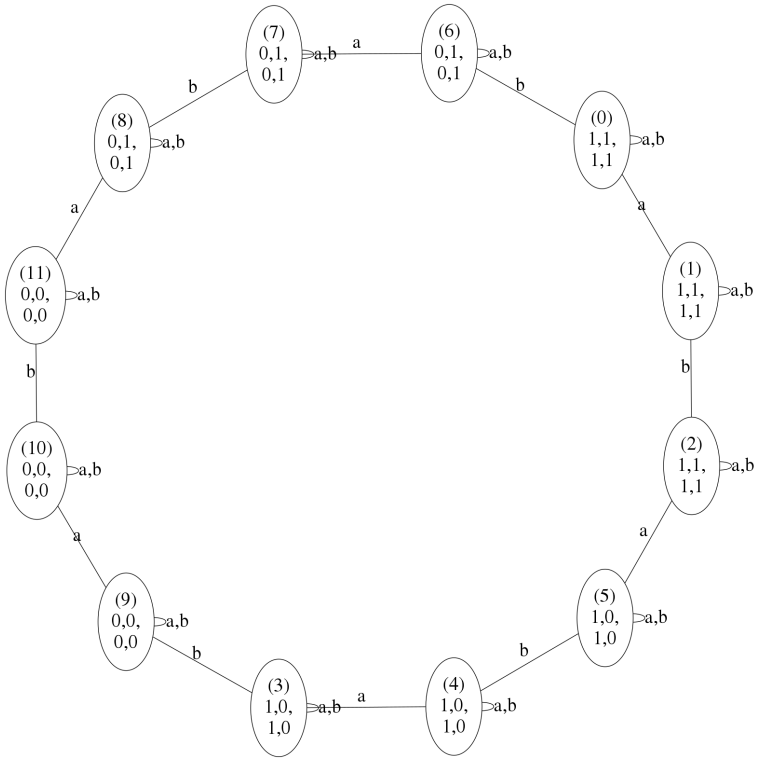

Figura 2.2: Ejemplo de modelo de Kripke.

2.1.5. Bisimulación y bisimilaridad

En esta subsección primero presentamos la definición de un sistema de transiciones etiquetadas ya que en particular los modelos de Kripke que presentaremos más adelante son en esencia un sistema de transiciones etiquetadas. Enseguida presentamos la definición de bisimulación que nos da pauta para poder hablar de sistemas de transiciones etiquetadas con el mismo comportamiento. Luego revisamos el concepto de bisimilaridad, que nos da una manera de calcular sistemas de transiciones etiquetados mínimos en tamaño. En nuestro trabajo necesitamos manejar modelos de Kripke mínimos en tamaño para poder identificar los mundos con una fórmula lógica. Por último presentamos el algoritmo de bisimilaridad de Kanellakis y Smolka que adecuamos e implementamos en este trabajo para calcular los modelos de Kripke mínimos bajo bisimulación.

2.1.5.1. Sistemas de transiciones etiquetadas

A continuación presentamos la definición de un sistema de transiciones etiquetadas.

Def. Un *sistema de transiciones etiquetadas*, *labeled transition system*, o *LTS*, por sus siglas en inglés, es una tripleta $(Pr, Act, \longrightarrow)$ donde Pr es el dominio del sistema de transiciones etiquetadas; Act es el dominio de acciones o etiquetas; y $\longrightarrow \subseteq Pr \times Act \times Pr$ es su relación de transición.

En la figura 2.3 se muestra gráficamente un sistema de transiciones etiquetadas para una máquina expendedora de café y té tomado de [12] donde $Pr = \{P_1, P_2, P_3, P_4\}$, $Act = \{1c, request - tea, request - coffee, tee, coffee\}$ y $\longrightarrow = \{(P_1, 1c, P_2), (P_2, request - tea, P_3), (P_2, request - coffee, P_4), (P_3, tea, P_1), (P_4, coffee, P_1)\}$.

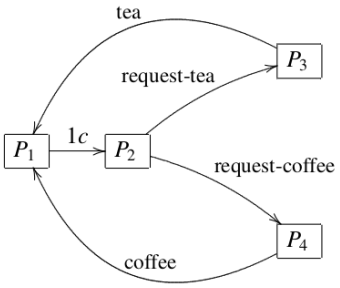

Figura 2.3: Sistema de transiciones etiquetadas para una maquina expendedora de café y té.

2.1.5.2. Bisimulación y bisimilaridad

Consideremos un sistema de transiciones etiquetadas $(Pr, Act, \longrightarrow)$. Sean $P, Q \in Pr$. Escribimos $P \xrightarrow{\mu} Q$ cuando $(P, \mu, Q) \in \longrightarrow$. Para cada μ, $\xrightarrow{\mu}$ es una relación binaria en Pr.

Def. Una *relación de procesos* es una relación binaria sobre Pr.

Def. Una relación de proceso $\mathcal{R}$ es una *bisimulación* si, siempre que ${}_P\mathcal{R}_Q$ para cualquier μ tenemos que:

1. Para toda P' con $P \xrightarrow{\mu} P'$ existe una Q' tal que $Q \xrightarrow{\mu} Q'$ y ${}_{P'}\mathcal{R}_{Q'}$.
2. Para toda Q' con $Q \xrightarrow{\mu} Q'$ existe una P' tal que $P \xrightarrow{\mu} P'$ y ${}_{P'}\mathcal{R}_{Q'}$

Def. *Bisimilaridad* es la unión de todas las bisimulaciones.

2.1.5.3. Algoritmo de Kanellakis y Smolka

Consideremos un sistema de transiciones etiquetadas $(Pr, Act, \longrightarrow)$. Ahora presentamos el algoritmo de Kanellakis y Smolka para el cálculo de bisimilaridad.

Sea $\pi = \{B_1, \ldots, B_k\}, k \geq 0$ una partición del conjunto de estados Pr. El algoritmo de Kanellakis y Smolka está basado en el concepto de *divisores*.

Def. Un *divisor* para un bloque $B_i \in \pi$ es un bloque $B_j \in \pi$ tal que para alguna acción $a \in Act$ algunos estados en B_i tienen transiciones hacia algún estado en B_j y otros no.

Intuitivamente, la existencia de un bloque divisor B_j para el bloque B_i indica una razón por la que se deben distinguir dos subconjuntos de estados en B_i: los que van a B_j y los que no. Así, el bloque B_i da lugar a dos bloques B_i^1 y B_i^2 refinando la partición π a una nueva partición $\pi' = \{B_1, B_2, \ldots, B_{i-1}, B_i^1, B_i^2, B_{i+1}, \ldots, B_k\}$. La idea básica del algoritmo consiste en encontrar bloques divisores hasta que no haya refinación posible.

Por ejemplo consideremos el sistema de transiciones etiquetadas mostrado en la figura 2.4.

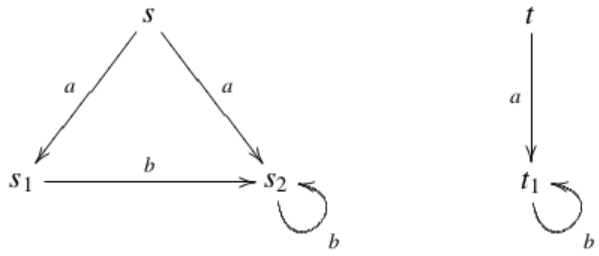

Figura 2.4: Un sistema de transiciones etiquetadas.

Como primera partición podemos tomar a $\pi = \{\{s, s_1, s_2, t, t_1\}\}$.

P_r es un bloque divisor para él mismo ya que algunos estados tienen transiciones etiquetadas con a y otros no. Si dividimos Pr con Pr respecto a la acción a obtenemos $\pi' = \{\{s, t\}, \{s_1, s_2, t_1\}\}$. Podemos observar que los bloques de π no se pueden dividir más con respecto a ninguna acción por lo que podemos decir que los estados en el mismo bloque son bisimilares uno a uno, mientras que los estados en diferentes bloques no lo son.

El algoritmo 1 que define la función SPLIT y el algoritmo 2 define el

algoritmo KS para calculo de bisimilaridad de Kanellakis y Smolka que usa SPLIT como subrutina.

Algoritmo 1 Función SPLIT

```
1: function SPLIT(B, a, π)
2:     Sea s ∈ B
3:     B_1 ← ∅
4:     B_2 ← ∅
5:     for all t ∈ B do
6:         if s y t alcanzan los mismos bloques con la acción a then
7:             B_1 ← B_1 ∪ {t}
8:         else
9:             B_2 ← B_2 ∪ {t}
10:        end if
11:    end for
12:    if B_2 = ∅ then
13:        return {B_1}
14:    else
15:        return {B_1, B_2}
16:    end if
17: end function
```

2.1.5.4. Adecuación del algoritmo de Kanellakis y Smolka

Teniendo en cuenta que un modelo de Kripke $\mathcal{M} = \langle W, R, L\rangle$ es un sistema de transiciones etiquetadas cuyos mundos (estados) están etiquetados por la función L debemos particionar inicialmente el conjunto W con respecto a los etiquetamientos de los mundos para asegurar que dos mundos con etiquetas distintas no resulten en el mismo bloque después del refinamiento de la partición.

2.1.6. Modelos de acciones

En esta subsección presentamos los *modelos de acciones* así como un caso de ejemplo. Los modelos de acciones nos brindan una forma de modelar cambio en los modelos epistémicos. Particularmente nos interesa el cambio en el conocimiento de los agentes al comunicarse. Los modelos de Kripke

Algoritmo 2 Algoritmo de Kanellakis y Smolka

```
1: function KS((Pr, Act, ⟶))
2:     π ← {Pr}
3:     changed ← true
4:     while changed do
5:         changed ← false
6:         for all B ∈ π do
7:             for all a ∈ Act do
8:                 if SPLIT(B, a, π) = {B1, B2} then
9:                     π ← (π − {B}) ∪ {B1, B2}
10:                    changed ← true
11:                end if
12:            end for
13:        end for
14:    end while
15: end function
```

operados con un *modelo de acciones* por el *producto modal restringido* da como resultado un nuevo modelo de Kripke que refleja el cambio epistémico representado en el modelo de acciones.

Def. Sea $\mathcal{L}$ el lenguaje de lógica epistémica proposicional sobre el conjunto de proposiciones P y A el conjunto de agentes sobre el que se define $\mathcal{L}$. Un *modelo de acciones* para lógica epistémica es una estructura $M = \langle S, \sim, pre \rangle$ tal que

1. S es el dominio de acciones
2. $\sim \ = \bigcup \sim_a, a \in A$ con $\sim_a$ una relación de equivalencia en S
3. $pre : S \longrightarrow \mathcal{L}$ una función de precondiciones

Def. Sean $\mathcal{M} = \langle W, R, L \rangle$, $\mathcal{M}' = \langle W', R', L' \rangle$ modelos de Kripke y $M = \langle S, \sim, pre \rangle$ un modelo de acciones. El producto modal restringido $\mathcal{M}' = \mathcal{M} \otimes M$ se define de la siguiente forma:

- $W' = \{(w, s) \mid w \in W, s \in S, \ y \ \mathcal{M}, w \vDash pre(s)\}$
- $((w_1, s_1), (w_2, s_2)) \in R' \Leftrightarrow (w_1, w_2) \in R \ y \ (s_1, s_2) \in \ \sim$
- $p \in L'((w, s)) \Leftrightarrow p \in L(w)$

2.1.6.1. Ejemplo

A continuación presentamos un ejemplo de un modelo de acciones que representa el siguiente escenario: Ana y Benito se encuentran en una mesa tomando café. Ambos esperan saber si el valor de las acciones de la compañía X subieron. Esta información está en una carta que está en su poder. Benito deja la mesa por un momento para ir al baño y cuando regresa sospecha que Ana leyó la carta. Ana no leyó la carta pero Benito en ese momento no puede distinguir entre tres escenarios: Ana leyó la carta y en efecto el valor de las acciones subió, Ana leyó la carta y el valor de las acciones no subió, o Ana no leyó la carta. Notemos que Ana puede distinguir entre las tres posibilidades.

Un modelo de acciones para el ejemplo presentado en el párrafo anterior consiste en tres puntos de acción que podemos nombrar como **p** para cuando en que Ana lee la carta y ve que las acciones subieron, **np** para cuando Ana lee la carta y las acciones no subieron y **t** para cuando Ana no lee la carta. Las relaciones entre los puntos de acción del modelo son la identidad para Ana ya que distingue entre los tres escenarios y la universal para Benito pues no puede distinguir entre ninguno. En cuanto a las precondiciones la precondición para **np** es $\neg p$, para **p** es p y para **t** es $\top$.

En la figura 2.5 se puede visualizar el modelo de acciones, el modelo de Kripke inicial y el modelo de Kripke que resulta. Debemos aclarar que el etiquetamiento del modelo de Kripke inicial del 0 es $\emptyset$, mientras que el del mundo 1 es $\{p\}$.

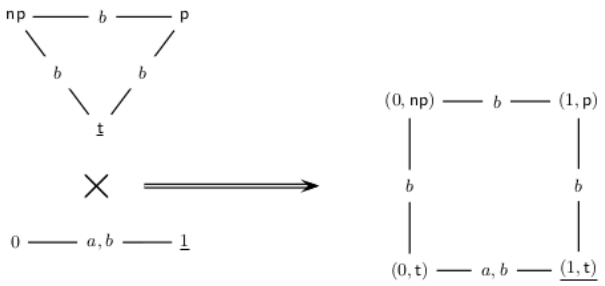

Figura 2.5: Modelo de acciones, modelo epistémico inicial y modelo epistémico final del ejemplo. [3]

Notemos que en el modelo de Kripke que resulta, el del lado derecho, los mundos $(0, p)$ y $(1, np)$ no están ya que **p** tiene como precondición p y **np** tiene como precondición $\neg p$.

2.2. Cómputo distribuido

En la segunda sección tratamos temas relacionados al cómputo distribuido. Primero la perspectiva topológica del mismo y los mapeos portadores con los que se modela la evolución de los protocolos. Finalmente presentamos el modelo de cómputo *Iterated Immediate Snapshots*. En este trabajo estudiamos un tipo particular de protocolos en el modelo mencionado donde en cada iteración los procesos solo escriben su entrada, lo que refleja la concurrencia de los procesos pero hace más sencilla la representación gráfica de las estructuras. El contenido de esta sección fue tomado de [9] y [2].

2.2.1. Perspectiva topológica del cómputo distribuido

Una forma de analizar el cómputo distribuido es a través de la topología combinatoria. En este enfoque se representan a todas las posibles ejecuciones de un número determinado de procesos haciendo uso de *complejos simpliciales*. A continuación presentamos definiciones formales respecto a los complejos simpliciales.

Def. Dado un conjunto S y una familia $\mathcal{A}$ de subconjuntos de S, decimos que A es un *complejo simplicial* sobre S si lo siguiente se satisface:

1. $X \in \mathcal{A}$, y $Y \subseteq X, entonces\ Y \in \mathcal{A}$.
2. $\{v\} \in \mathcal{A}, \forall v \in S$.

Def. Sea $\mathcal{A}$ un complejo simplicial, y $\sigma \in \mathcal{A}$. La *dimensión* de σ es $|\sigma|-1$.
Def. Sean σ, τ simplejos de un complejo simplicial $\mathcal{A}$, decimos que:

1. τ es una *cara* de σ si $\tau \subseteq \sigma$
2. τ es una *cara propia* de σ si $\tau \subset \sigma$
3. σ es una *faceta* en $\mathcal{A}$ si no es una cara propia de ningún otro simplejo en $\mathcal{A}$

Def. Sea $\mathcal{A}$ un complejo simplicial, decimos que $\mathcal{A}$ es *puro* si todas sus facetas tienen la misma dimensión.

En este trabajo es importante estudiar la evolución del conocimiento que tienen los procesos a lo largo de un protocolo. La evolución de los complejos simpliciales se define por medio de mapeos portadores a los que definimos a continuación.

Def. Dados $\mathcal{A},\ \mathcal{B}$ complejos simpliciales, un *mapeo portador* $\Phi : \mathcal{A} \to 2^{\mathcal{B}}$ manda cada simplejo $\sigma \in \mathcal{A}$ a un subcomplejo $\Phi(\sigma)$ de $\mathcal{B}$ tal que $\forall\ \sigma, \tau \in A,\ si\ \sigma \subseteq \tau\ entonces\ \Phi(\sigma) \subseteq \Phi(\tau)$.

Def. Un *m-etiquetamiento*, o simplemente *etiquetamiento*, de un complejo simplicial $\mathcal{A}$ es una función que asocia cada vértice de $\mathcal{A}$ a un elemento de un dominio de cardinalidad m, en otras palabras, es una función $\phi : V(\mathcal{A}) \longrightarrow D$ donde $|D| = m$.

Def. Una *m-coloración*, o simplemente *coloración*, de un complejo simplicial $\mathcal{A}$ de dimensión n es un m-etiquetamiento $\chi : V(\mathcal{A}) \longrightarrow \Pi$ inyectivo en cada simplejo de $\mathcal{A}$, es decir, que asocia un color distinto a cada elemento del simplejo.

Def. Llamamos *complejo simplicial cromático* a un complejo simplicial $\mathcal{A}$ junto con una coloración χ.

2.2.2. Iterated Immediate Snapshots

El modelo de cómputo *Iterated Immediate Snapshots* fue presentado por primera vez en [2]. Dicho modelo es iterativo y en la iteración i los procesos escriben y después leen lo que todos los demás procesos escribieron en esa iteración. La lectura se hace por medio de un *snapshot* que es una lectura atómica de memoria. Este modelo se definió mediante los modelos *Immediate Snapshot* y *one-shot immediate snapshot* que mencionamos enseguida.

El modelo Immediate Snapshot comprende ejecuciones dónde un número maximal de escrituras es seguido de un número maximal de snapshots de los mismos procesos. Es decir, todos los procesos que concurren en la ejecución primero escriben y después, terminadas todas las escrituras, hacen un snapshot de la memoria. Este comportamiento se puede condensar en una operación WriteRead(*Value*) atómica.

Se dice que un one-shot immediate snapshot es un modelo *Immediate Snapshot* que permite que cada proceso escriba solamente una vez en el registro de memoria que le corresponde.

En el modelo *Iterated Immediate Snapshots* se tiene una secuencia infinita $M_0, M_1, \ldots$ de memorias one-shot immediate snapshot, una para cada iteración de un protocolo. La ejecución de un protocolo de información completa, en donde los procesos escriben todo lo que han visto hasta el momento, comienza con que cada uno de los procesos P_i realicen una operación WriteRead(I_i) en la memoria M_0, donde I_i es la entrada del proceso i. En

iteraciones posteriores se aplica la salida del snapshot de M_i, como entrada la memoria M_{i+1}.

Una ejecución en este modelo es una secuencia infinita de particiones ordenadas del conjunto de procesos P_i.

2.2.3. Protocolos de estudio

Generalmente al estudiar el modelo de cómputo *Iterated Immediate Snapshots* se estudian los protocolos de información completa. En este trabajo, por simplicidad estudiamos protocolos en este modelo en los que en cada iteración los procesos escriben en memoria solamente su entrada. La esencia que captura este tipo de protocolos es la concurrencia de las ejecuciones. La memoria local de los procesos es infinita y en cada iteración cada proceso sabe lo que ha leído en cada una de las iteraciones. La memoria local infinita es necesaria para que se pueda distinguir entre todas las posibles ejecuciones pues las ejecuciones en este modelo son secuencias infinitas de particiones del conjunto de procesos y se necesita recordar la lectura en cada iteración.

2.2.4. Subdivisiones cromáticas

Un concepto importante en este trabajo son las subdivisiones cromáticas, ya que a lo largo de la ejecución de los protocolos del modelo *Iterated Immediate Snapshots* los complejos simpliciales evolucionan justamente por medio de subdivisiones cromáticas.

Intuitivamente, una *subdivisión* de un complejo simplicial $\mathcal{A}$ es un complejo simplicial $\mathcal{B}$ construido dividiendo los simplejos $\sigma \in \mathcal{A}$ en simplejos más pequeños.

En su versión combinatoria definimos a la subdivisión cromática estándar como sigue:

Def. Sea $\mathcal{A}$ un complejo simplicial cromático cuya función de coloración es χ. La *subdivisión cromática estándar* $Ch\ \mathcal{A}$ de $\mathcal{A}$ es el complejo simplicial con vértices de la forma (i, σ_i) donde $i \in 1, \ldots, n$, σ_i no es una cara vacía de $\mathcal{A}$, e $i \in \chi(\sigma_i)$. Una $(k+1)$-tupla $(\sigma_0, \ldots, \sigma_k)$ es un simplejo de $Ch\ \mathcal{A}$ si y solo si:

- La tupla puede ser indexada de forma que $\sigma_0 \subseteq \cdots \subseteq \sigma_k$ y
- Para $0 \leq i, j \leq n$, si $i \in \chi(\sigma_j)$ entonces $\sigma_i \subseteq \sigma_j$

El modelo de cómputo *Iterated Immediate Snapshots* produce subdivisiones cromáticas del complejo simplicial en cada iteración para protocolos de información completa. Particularmente, nuestros protocolos de estudio funcionan de la misma forma que los de información completa en la primera iteración. En la figura 2.6 se muestra la subdivisión cromática para tres procesos con entrada 0. Cada faceta del complejo simplicial está etiquetada con la ejecución parcial para una iteración de un protocolo.

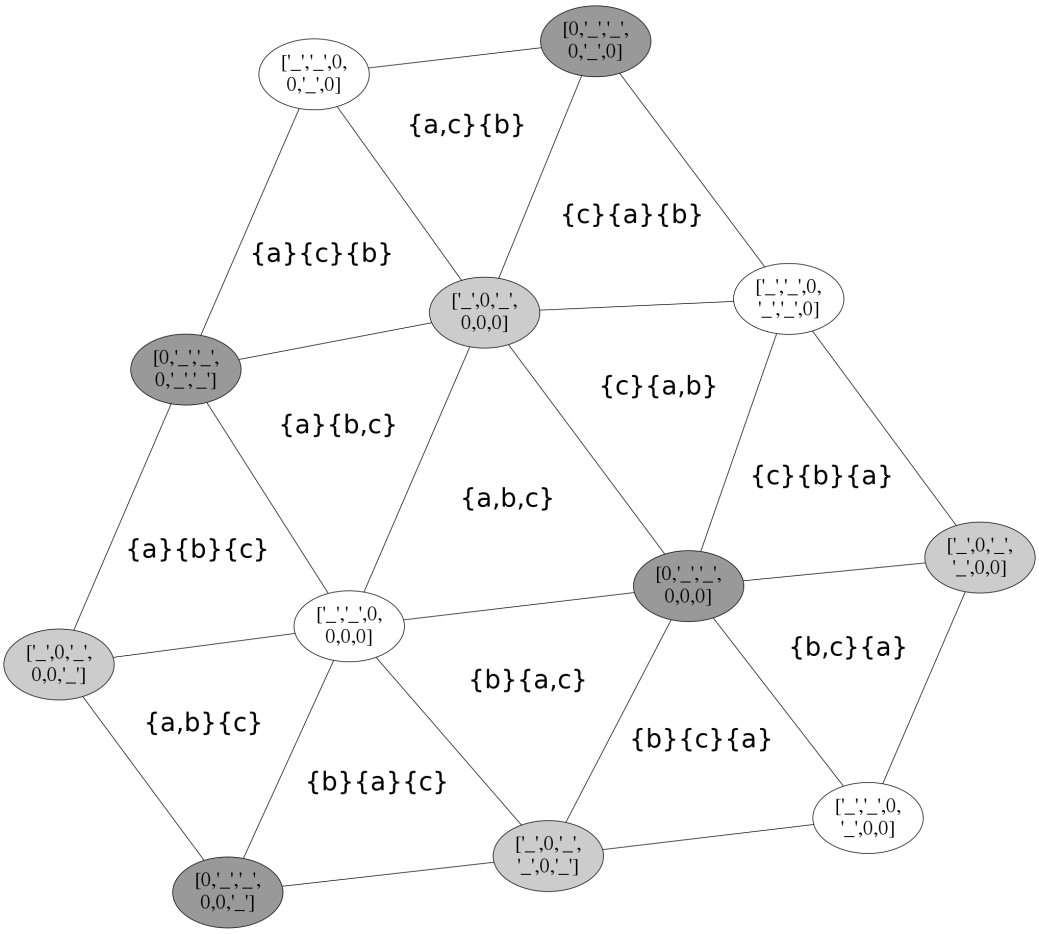

Figura 2.6: Subdivisión cromática después de la primera iteración para tres procesos con entrada 0 en el modelo *Iterated Immediate Snapshots*.

Capítulo 3

Transformaciones

En este capítulo se presenta primero una operación de proyección sobre procesos de complejos simpliciales que tiene como propósito trabajar con las diferentes dimensiones representadas en el complejo simplicial. Luego presentamos las transformaciones de conjuntos de simplejos en complejos simpliciales con las características señaladas en la sección 3.1 a modelos de Kripke y viceversa. Después damos ejemplos de ambas transformaciones. Por último comentamos algunos detalles relacionados con las transformaciones que presentamos.

Tomemos en cuenta los procesos que interactúan a lo largo de un protocolo, mismo que se describe por un conjunto de instrucciones. Decimos que un *bloque de instrucciones* es una secuencia no vacía de instrucciones del protocolo. Decimos que un proceso p está *vivo* en un lapso de tiempo l si p se encuentra en posibilidad de interactuar con los demás procesos a lo largo de l. Finalmente, decimos que un proceso está *muerto* si no está vivo.

En el capítulo 1 se mencionaron las transformaciones que se proponen en [5] para transformar elementos en las categorías de complejos simpliciales a elementos en la categoría de marcos de Kripke tomando en cuenta solamente las facetas de complejos simpliciales cromáticos puros.

En este capítulo, a diferencia del trabajo en [5], tratamos las transformaciones desde un punto de vista más amplio en el sentido que podremos representar cualquier conjunto de simplejos del complejo simplicial original como un modelo de Kripke, pero también restringiendo el tipo de complejos simpliciales que manejamos.

Las transformaciones se presentan desde un punto de vista de las lógicas multimodales. En casos particulares, la lógica multimodal corresponde al de

la lógica epistémica. Cuando haya correspondencia con la lógica epistémica, podemos decir que un modelo de Kripke se construye a partir del conocimiento distribuido entre los procesos representados en el complejo simplicial. En contraste, un complejo simplicial se construye a partir del conocimiento de cada uno de los agentes en el modelo de Kripke.

3.1. Los complejos simpliciales que tratamos

Trataremos con complejos simpliciales cromáticos cuyos vértices se pueden identificar por medio de su color y su vista. Las vistas de los procesos son necesarias desde el punto de vista de la lógica pues ellas darán lugar a los átomos proposicionales sobre los que trabajará el modelo de Kripke. Además, tendremos como restricción que las vistas de todos los vértices del complejo simplicial tengan el mismo tamaño, de tal forma que cada posición da lugar a una familia de proposiciones que describimos más adelante.

3.2. Una diferencia de perspectiva

En [5] se propone construir los marcos de Kripke correspondientes a un complejo simplicial cromático puro a través de la dimensión más alta, transformando cada faceta del complejo en un mundo del marco de Kripke. Esa forma de transformar es solo un caso posible en el que *solamente se representan las interacciones que se llevan a cabo entre todos los procesos*. Esta forma de transformación es útil en modelos de cómputo asíncronos donde no se puede distinguir entre la "muerte" de un proceso o simplemente un retraso del mismo.

En este trabajo, a diferencia de [5], proponemos formas de tratar con cualquier dimensión del complejo simplicial. Si se quiere obtener un modelo epistémico, se proponen proyecciones sobre procesos del complejo simplicial, para después transformar las facetas del complejo simplicial proyectado en un modelo de Kripke[1]. En otro caso, la transformación que proponemos toma cualquier subconjunto del complejo simplicial y construye el modelo de Kripke que le corresponde.

[1]Las proyecciones nos dan la posibilidad de reducir el tamaño del modelo de Kripke según nuestras necesidades de análisis.

Observemos que solo algunos modelos se podrán analizar epistémicamente pues si se representan distintas dimensiones del complejo simplicial en el modelo de Kripke, la relación de accesibilidad deja de ser reflexiva desde el punto de vista de algún agente[2]. Lo anterior da entrada a la posibilidad de hacer análisis usando una lógica distinta a la lógica epistémica, que capture la esencia de lo que un agente a ve condicionando a que el proceso correspondiente al agente a haya participado en la interacción modelada, es decir, que siendo w el mundo que representa el escenario mencionado wR_aw.[3]

3.3. Proyecciones de complejos simpliciales

En los complejos simpliciales se representan múltiples dimensiones. Cada simplejo en el complejo simplicial tiene una dimensión igual a uno menos que su cardinalidad, y cada uno representa una interacción entre los procesos correspondientes a los colores de los vértices que contiene.

En esta sección presentamos un procedimiento para proyectar un complejo simplicial con las características mencionadas al principio del capítulo, a una dimensión inferior mediante un conjunto de procesos con el supuesto de que solamente ellos están vivos y que ningún otro ha ejecutado el bloque en el que ellos concurren.[4]

Para calcular los subcomplejos simpliciales de la forma mencionada primero se deben eliminar del complejo simplicial los simplejos que contengan vértices con colores que no correspondan a los procesos vivos, pues suponemos que los demás están muertos. Luego tenemos que eliminar del complejo simplicial los simplejos que contengan vértices con vistas no compatibles con el supuesto de que solo los procesos en el conjunto están vivos y que ningún otro ha ejecutado el bloque representado en el complejo simplicial. Finalmente debemos hacer que el complejo simplicial sea puro, ya que todos los

[2]En la transformación en 3.4 se calcula la relación de accesibilidad por medio del conjunto de colores en la intersección de los simplejos, por lo que si se toma un simplejo σ de dimensión menor a la del complejo simplicial $\mathcal{C}$ al que pertenece, $|colores(\sigma \cap \sigma)| < colores(\mathcal{C})$.

[3]Si wR_aw figura en el modelo de Kripke, quiere decir que el agente a participó en la interacción de los procesos que representa w.

[4]La proyección sobre procesos vivos de un complejo simplicial y la proyección sobre agentes de un modelo de Kripke que se presenta en 4.4.2 están relacionadas con el cálculo de la estructura inducida por mapeos portadores en dimensiones inferiores, lo que se explica en 4.4.3.

procesos que prevalecen se suponen vivos.

3.3.1. Vistas compatibles para el modelo *Iterated Immediate Snapshots*

La noción de vistas compatibles con el supuesto de que solamente un conjunto de procesos están vivos en un lapso de tiempo puede variar dependiendo de las reglas que siga el modelo de cómputo representado. En el modelo *Iterated Immediate Snapshots*, donde cada proceso escribe en una posición de la memoria en cada iteración, *el concepto de vista compatible correspondería a vistas en donde el renglón y las columnas correspondientes a la última iteración de procesos supuestamente muertos tengan valor* $\perp$.

Recordemos que en el modelo de cómputo mencionado, en cada iteración los procesos hacen una escritura en una posición específica para cada iteración y proceso, por lo que podemos representar la memoria usada en un protocolo de este modelo como un arreglo bidimensional de k renglones y n columnas.

En el algoritmo 3 se presenta el procedimiento para decidir si una vista es compatible con el supuesto de que solo un conjunto de procesos terminan la ejecución del bloque de instrucciones de la última iteración ejecutada de un protocolo en el modelo *Iterated Immediate Snapshots* en un lapso específico de tiempo.

Como entrada recibe:

- *Vista*, la vista a revisar como un arreglo unidimensional de tamaño $(k+1)\times N$, donde k es el número de iteraciones ejecutadas del protocolo
- N, el número de procesos que participan en el protocolo
- $Columnas$, el conjunto de columnas en las que los procesos vivos escriben

Como resultado obtiene:

- $true$, en caso de que *Vista* en las posiciones correspondientes a la última iteración y las columnas $c \notin Columnas$ tengan valor $\perp$
- $false$, en otro caso

Algoritmo 3 Algoritmo para decidir si una vista es compatible, en nuestros protocolos de estudio, con el supuesto de un conjunto exclusivo de procesos vivos.

```
1: function VistaCompatibleIIS(Vista, N, Columnas)
2:     tam ← tamaño de Vista
3:     r ← tam/N − 1
4:     for c ∈ {0, 1, ..., N − 1} − Columnas do
5:         if Vista[r × N + c] ≠ ⊥ then
6:             return false
7:         end if
8:     end for
9:     return true
10: end function
```

3.3.2. Proyección de complejos simpliciales sobre agentes vivos para el modelo *Iterated Immediate Snapshots*

En esta subsección presentamos el algoritmo 4, que calcula las proyecciones de un complejo simplicial que representa las ejecuciones de nuestros protocolos de estudio en el modelo *Iterated Immediate Snapshots* sobre un conjunto de procesos que terminan la ejecución de la ultima iteración representada. Suponemos que el conjunto de procesos dados son los únicos que han ejecutado dicho bloque.

Como entrada recibe:

- $\mathcal{C}$, el complejo simplicial a proyectar
- N, El número de procesos que participan en el protocolo
- $VivosRColumnas$, una relación de los procesos vivos presentados por sus colores correspondientes y la columna en las que escribe cada uno

Como resultado obtiene:

- $\mathcal{C}'$, el complejo simplicial proyectado

Obviamos la existencia de varias funciones ya que su definición es sencilla.

Algoritmo 4 Algoritmo para proyectar un complejo simplicial sobre un conjunto de procesos vivos.

```
1: function ProyecciónIIS(C, N, VivosRColumnas)
2:     coloresVivos ← colores(VivosRColumnas)
3:     C' ← {}
4:     for all σ ∈ C do                ▷ Eliminar simplejos con procesos no vivos
5:         coloresSimplejo ← colores(σ)
6:         if coloresSimplejo − coloresVivos = ∅ then
7:             C' ← C' ∪ {σ}
8:         end if
9:     end for
10:    C ← C'
11:    C' ← {}
12:    columnasVivos ← columnas(VivosRColumnas)
13:    for all σ ∈ C do                ▷ Eliminar simplejos con vistas incompatibles
14:        vista ← vista(σ)
15:        if VistaCompatibleIIS(vista, N, columnasVivos) then
16:            C' ← C' ∪ {σ}
17:        end if
18:    end for
19:    cambio ← true
20:    while cambio do                 ▷ Eliminar facetas de dimensión menor
21:        cambio ← false
22:        F ← {f ∈ facetas(C') | dim(f) < |coloresVivos| − 1}
23:        if F ≠ ∅ then
24:            C' ← C' − F
25:            cambio ← true
26:        end if
27:    end while
28:    return C'
29: end function
```

Teorema 3.3.1. *Sea $\mathcal{C}$ un complejo simplicial cromático, cuyos vértices se pueden identificar por medio de su color y su vista y en el que las vistas de todos sus vértices tengan el mismo tamaño. Si $\mathcal{C}'$ es una proyección generada con el algoritmo 4 a partir de $\mathcal{C}$, $\mathcal{C}'$ es un complejo simplicial cromático puro.*

Demostración. Demostramos que el algoritmo 4 calcula un complejos simplicial cromático puro haciendo un seguimiento de las asignaciones de la variable $\mathcal{C}'$.

Después de la ejecución del ciclo que comienza en la línea 4 del algoritmo 4, quedan en $\mathcal{C}'$ todos los simplejos del complejo simplicial que representan interacciones entre los procesos que se suponen vivos. Observemos que el resultado sigue siendo un complejo simplicial pues si un simplejo σ se agrega a $\mathcal{C}'$ cualquier simplejo $\sigma' \subseteq \sigma$ también será agregado. Además notemos que cualquier simplejo contenido en $\mathcal{C}'$ es de dimensión menor a $|coloresVivos|$.

Después de la ejecución del ciclo que comienza en la línea 13 del algoritmo 4, $\mathcal{C}'$ contiene todos los simplejos del complejo simplicial que representan interacciones entre los procesos que se suponen vivos y que además cumplen con la característica de que todos sus vértices tienen vistas compatibles. Notemos que $\mathcal{C}'$ sigue siendo un complejo simplicial ya que si un simplejo σ no es agregado a $\mathcal{C}'$ es porque no tiene una vista compatible. Sin embargo, cualquier $\sigma' \supseteq \sigma$ tampoco será agregado pues en su vista habrá al menos el mismo número de incompatibilidades que en la vista de σ.

Como al inicio del ciclo que comienza en la línea 20 del algoritmo 4, $\mathcal{C}'$ es un complejo simplicial y además en cada ciclo solamente se eliminan facetas, al final del ciclo $\mathcal{C}'$ sigue siendo un complejo simplicial. Más aún es un complejo simplicial puro ya que el ciclo termina una vez que no haya facetas de dimensión menor a uno menos que el número de procesos que se suponen vivos. Como todos los simplejos en $\mathcal{C}'$ estaban contenidos en el complejo simplicial de entrada, podemos concluir que $\mathcal{C}'$ es un complejo simplicial cromático puro.

□

3.4. Conjunto de simplejos a Modelo de Kripke

En esta sección presentamos el procedimiento para construir un modelo de Kripke a partir de un subconjunto de simplejos de un complejo simplicial. Primero damos una intuición acerca del proceso, para luego presentar algoritmos específicos para nuestros protocolos de estudio.

Sea $\mathcal{C}$ un complejo simplicial con las características mencionadas en 3.1, $\mathcal{S} \subseteq \mathcal{C}$ un subconjunto de simplejos de $\mathcal{C}$ y A un subconjunto de colores de vértices en $\mathcal{S}$. El procedimiento para construir un modelo de Kripke $\mathcal{M} = \langle W, R, L \rangle$ a partir del complejo simplicial $\mathcal{C}$ y $\mathcal{S}$ consta de tres pasos. Primero se construye W, un conjunto de mundos en biyección con cada simplejo en $\mathcal{S}$. Luego se construye $R = \bigcup R_a,\ a \in A$ de tal forma que siendo $w_1, w_2 \in W$ correspondientes a $s_1, s_2 \in \mathcal{S}$ respectivamente, $w_1 R_a w_2$ sii $a \in colores(s_1 \cap s_2)$. Finalmente se construye L, el etiquetamiento del modelo de Kripke inducido por las vistas de los procesos representados en $\mathcal{C}$.

3.4.1. Cálculo de etiquetamiento para nuestros protocolos de estudio

Nuestros protocolos de estudio se representan como una iteración de k ciclos de tal forma que en cada iteración cada proceso escribe en una posición de memoria específica para cada iteración y proceso y después lee una fotografía de la memoria en ese instante. Además, los procesos que concurren, primero hacen todos su escritura y después hacen todos su lectura. Así que podemos pensar que la memoria se puede representar lógicamente como un arreglo de k renglones, uno por cada iteración por n columnas, siendo n el número de procesos que participan en el protocolo.

Podemos pensar en una lógica cuyo conjunto de átomos proposicionales de la forma $p_{k,a,v}$ con semántica "En la iteración k el agente a escribió v en su posición correspondiente de memoria", de tal forma que podemos generar un etiquetamiento de la siguiente forma:

Dada la memoria representada por las vistas podemos suponer que la vista representa el arreglo lógico bidimensional de memoria en una secuencia de $(k+1) \times n$ posiciones donde las primeras n son todas $\perp$ a excepción de la posición en la que escribe el proceso dueño de la vista que tiene su valor de entrada y los k siguientes grupos de n, corresponden a los valores leídos

en cada iteración.

Comparando entrada por entrada, los valores de las vistas de los procesos que interactúan en el simplejo haremos una "fusión" de vistas para generar después el etiquetamiento del mundo correspondiente como sigue: Si todas las entradas en las vistas tienen el mismo valor, preservar el valor. Si hay valores distintos[5] preservar el valor distinto a $\perp$[6].

Una vez generada la fusión de vistas, el etiquetamiento para el mundo correspondiente a cada simplejo σ correspondiente al mundo w en el modelo de Kripke será $L(w) = \{p_{k,a,v} |$ *La fusión de vistas de los vértices en el simplejo correspondiente a* w *en la posición* $k \times n + col(a)$ *tiene escrito el valor* $v\}$ comenzando con $k = 0$.

3.4.1.1. Fusión de vistas

A continuación presentamos el algoritmo 5 para la fusión de vistas del modelo *Iterated Immediate Snapshots*. Recordemos que en los complejos simpliciales con los que trabajamos todos los vértices del mismo tienen vistas del mismo tamaño y que particularmente en el modelo de estudio, cada proceso escribe valores en memoria disjunta para cada iteración y proceso.

Como entrada recibe:

- $Vistas$, una lista con las vistas de los vértices contenidos en un simplejo

Como resultado produce:

- f, La fusión de las vistas que contiene los valores escritos y vistos por al menos un proceso representado en el simplejo del que las vistas provienen

El algoritmo 5 simplemente recolecta la información leída por todos los procesos a los que pertenecen las vistas. Notemos que esta información es el conocimiento distribuido de los procesos con las vistas contenidas en $Vistas$.

[5]Solo pueden variar a $\perp$ cuando alguno de los procesos que interactúan aún no ha procesado la escritura al momento que se genera el snapshot.

[6]Notemos que $\perp$ es un valor posible cuando se supone una falla en algún proceso que evite la escritura respectiva, sin embargo no puede haber distintos valores pues cada proceso escribe solo una vez y en una localidad distinta de memoria a los demás.

Algoritmo 5 Algoritmo para fusión de vistas en nuestros protocolos de estudio.

```
1: function FusionDeVistasIIS(Vistas)
2:     l ← longitud(cabeza(Vistas))  ▷ Las vistas tienen el mismo tamaño.
3:     Sea f un arreglo de longitud l
4:     for i ← 1, l do
5:         values = {v|vista ∈ Vistas ∧ vista[i] = v} − {⊥}
6:         if values = {v} then
7:             f[i] ← v
8:         else
9:             f[i] ← ⊥
10:        end if
11:    end for
12:    return f
13: end function
```

3.4.1.2. Etiquetamiento para nuestros protocolos de estudio

A continuación se presenta el algoritmo 6 para el etiquetamiento de un mundo correspondiente a un simplejo dado.

Como entradas recibe:

- N, el número de procesos que interactúan en el complejo simplicial inicial
- Id, el identificador de mundo que se etiquetará
- $Simplejo$, el simplejo correspondiente al mundo con identificador id

Como resultado obtiene:

- $\langle Id, Etiqueta \rangle$, una pareja con el Id de un mundo y su etiquetamiento en el modelo de Kripke respectivo

El algoritmo 6 calcula primero la fusión de las vistas de los vértices del simplejo y después para cada valor en la fusión de las vistas calcula la proposición que le corresponde en el etiquetamiento del mundo con identificador Id.

Ahora presentamos el algoritmo 7 para etiquetar un modelo de Kripke.

Como entradas recibe:

Algoritmo 6 Algoritmo para el etiquetado de un mundo a partir de un simplejo en nuestros protocolos de estudio.

```
1: function EtiquetadoDeMundoIIS(N, Id, Simplejo)
2:     vistas ← vistas(Simplejo)
3:     f ← FusionDeVistasIIS(vistas)
4:     l = longitud(f)
5:     Etiqueta = ∅
6:     for i ∈ {0, 1, ..., l − 1} do
7:         k ← ⌊i/n⌋
8:         a ← agente a tal que col(a) = i mod n
9:         Etiqueta ← Etiqueta ∪ {p_{k,a,f[i]}}
10:    end for
11:    return ⟨Id, Etiqueta⟩
12: end function
```

- N, el número de procesos representados en el complejo simplicial inicial
- $IdsSimplejos$, la relación de identificadores de mundos y simplejos

Como resultado obtiene:

- Et, la función de etiquetamiento de mundos

Algoritmo 7 Algoritmo para calcular el etiquetamiento de un modelo de Kripke correspondiente a un conjunto de simplejos de un complejo simplicial para nuestros protocolos de estudio.

```
1: function EtiquetadoIIS(n, IdsSimplejos)
2:     Et ← ∅
3:     for all ⟨Id, Simplejo⟩ ∈ IdsSimplejos do
4:         Et ← Et ∪ EtiquetadoDeMundoIIS(N, Id, Simplejo)
5:     end for
6:     return Et
7: end function
```

El algoritmo 7 simplemente usa como subrutina el algoritmo 6 para calcular el etiquetamiento de cada uno de los mundos identificados por cada Id en los elementos $\langle Id, Simplejo\rangle$ de $IdsSimplejos$ y regresa la unión de todos los etiquetados.

3.4.2. Conjunto de simplejos a modelos de Kripke

Presentamos el algoritmo 8 para transformar subconjuntos de simplejos de un complejo simplicial con las características mencionadas en 3.1 en modelos de Kripke. Este algoritmo es específico para nuestros protocolos de estudio ya que usa el etiquetamiento presentado en el algoritmo 7.

Como entrada recibe:

- n, el número de procesos que modela el complejo simplicial $\mathcal{C}$
- $\mathcal{S}$, simplejos de $\mathcal{C}$ que se modelarán.

Como resultado obtiene:

- $\mathcal{M} = \langle W, R, L \rangle$, el modelo de Kripke correspondiente a los simplejos en $\mathcal{S}$

Argumentemos que el algoritmo 8 es correcto. Notemos que en el ciclo de las lineas 5-9 se construye el conjunto W y se biyecta con $\mathcal{S}$ por medio de la función iRs. En el ciclo de las lineas 12-17 se crea la relación del mundo identificado por id_τ con un mundo identificado por id_σ. La relación mencionada se construye para cada par posible de mundos en el ciclo de las lineas 11-18, por lo que al terminar el ciclo R contiene las relaciones como se describieron al principio. En la línea 19 se construye la función de etiquetamiento de los mundos. Por lo anterior el modelo de Kripke $\mathcal{M} = \langle W, R, L \rangle$ es el que correspondiente al conjunto de simplejos $\mathcal{S}$.

Algoritmo 8 Algoritmo para calcular el modelo de Kripke correspondiente a un subconjunto de simplejos de un complejo simplicial.

```
1: function SimplejosAModelo(N, S)
2:     i ← 0
3:     W ← ∅                                  ▷ Mundos del modelo de Kripke
4:     iRs ← ∅                                ▷ Relación de id de mundos y simplejos
5:     for all σ ∈ S do
6:         W ← W ∪ {i}
7:         iRs ← iRs ∪ ⟨i, σ⟩
8:         i ← i + 1
9:     end for
10:    R ← ∅                                  ▷ Relación de accesibilidad del modelo de Kripke
11:    for all σ ∈ S do
12:        for all τ ∈ S do
13:            κ ← σ ∩ τ
14:            Sea id_σ tal que ⟨id_σ, σ⟩ ∈ iRs
15:            Sea id_τ tal que ⟨id_τ, τ⟩ ∈ iRs
16:            R ← R ∪ ⟨id_σ, colores(κ), id_τ⟩
17:        end for
18:    end for
19:    L ← EtiquetadoIIS(n, iRs)
20:    return M = ⟨W, R, L⟩
21: end function
```

3.5. Conjuntos de simplejos y modelos epistémicos

Recordemos que los modelos $\mathcal{M} = \langle W, R, L \rangle$ de lógica epistémica sobre el conjunto de agentes A son de tal forma que cada $R|a$, $a \in A$, es una relación de equivalencia, así que para poder analizar epistémicamente un modelo generado tenemos que asegurar la propiedad mencionada.

Particularmente si generamos un modelo de Kripke representando varias dimensiones de un complejo simplicial, el modelo no es epistémico, ya que en algunos mundos no figurarían algunos procesos de tal forma que la relación con respecto a algún agente deja de ser reflexiva.

Si solamente tomamos en cuenta simplejos que contengan vértices de los mismos colores[7] que modelan las interacciones entre todos los procesos de un conjunto, el modelo que le corresponde será un modelo epistémico. Por ejemplo, podemos pensar en las facetas de alguna proyección del complejo simplicial calculada por medio del algoritmo 4.

Es importante tomar en cuenta el modelo de cómputo y los protocolos de estudio para seleccionar el conjunto de simplejos adecuado a analizar. Por ejemplo, si en el modelo de cómputo que usamos, los procesos no mueren, los únicos simplejos que deberemos tomar en cuenta para construir el modelo de Kripke serán las facetas, pues en ellas se representan las interacciones en las cuales todos los procesos participan en el protocolo. En contraste, en un modelo de cómputo que admite un número arbitrario de muertes explicitas, debemos tomar en cuenta también simplejos de dimensiones inferiores.

A pesar de que en algunos modelos de cómputo sea necesario lidiar con distintas dimensiones, tenemos la posibilidad de hacer análisis epistémico en una parte específica del complejo simplicial. Por ejemplo, podríamos analizar el caso en el que un proceso en particular muere en la ejecución de un protocolo y con ayuda de proyecciones como la del algoritmo 4 o cualquiera definida a nuestra conveniencia[8], seleccionando las facetas del complejo proyectado para construir el modelo y analizarlo epistémicamente.

[7] No necesariamente todos los colores del complejo simplicial original.

[8] Podríamos borrar toda interacción posible de un conjunto de procesos.

3.6. Modelo de Kripke a conjunto de simplejos

En esta sección, presentamos un procedimiento que a partir de un modelo de Kripke $\mathcal{M} = \langle W, R, L\rangle$, que representa ejecuciones de nuestros protocolos de estudio, sobre el conjunto de agentes A, construye el subconjunto de simplejos $\mathcal{S}$ que representan sus mundos, así como el complejo simplicial $\mathcal{C}$ mínimo que lo contiene. La transformación de un modelo de Kripke a conjuntos de simplejos debe ser el proceso inverso al presentado en la sección 3.4 y partir del conjunto de simplejos calculado, construimos el complejo simplicial más pequeño que los contiene. Intuitivamente, lo que debemos hacer es construir un simplejo por cada mundo en el modelo de Kripke, de tal forma que cada simplejo tenga los vértices que debe y cada vértice tenga la vista que debe.

Recordemos que en el proceso de etiquetado para un mundo del modelo de Kripke se hacía una fusión de las vistas de los vértices contenidos en el simplejo correspondiente para que a partir de ella se construyera el etiquetamiento del mundo en el modelo. Ahora debemos tratar de reconocer qué es lo que el proceso correspondiente a cada agente podría ver. Particularmente, la relación de accesibilidad de mundos para cada agente del modelo de Kripke modela la *indistinguibilidad* entre mundos para dicho agente, de tal forma que la vista de un proceso en el simplejo correspondiente a un mundo del modelo de Kripke se debería poder construir a partir del etiquetamiento en común que tuvieran los mundos accesibles desde el correspondiente al simplejo específico. En particular, cuando el modelo es epistémico, hay una correspondencia con la modalidad K_a.

El cálculo de las vistas para el modelo *Iterated Immediate Snapshots* tomando en cuenta el etiquetamiento propuesto en el algoritmo 7 se puede calcular de la siguiente forma:

Sea $\mathcal{M} = \langle W, R, L\rangle$ un modelo de Kripke sobre el conjunto de n agentes A, cuyos mundos se etiquetaron como se propone en el algoritmo 7, $a \in A$ y $w \in W$. La vista $Vista_{w,a}$ correspondiente al proceso a en el simplejo correspondiente al mundo w es de la siguiente forma:

Para cada $k \in \{0, 1, \ldots, k\}$ y para cada agente $a' \in A$

$$Vista_a[k \times n + col(a')] = \begin{cases} v & \mathcal{M}, w \vDash \Box_a(p_{k,a',v}) \\ \perp & \text{en otro caso} \end{cases}$$

9

Presentamos el algoritmo 9 para calcular los simplejos que representa un modelo de Kripke etiquetado por el algoritmo 7, así como el complejo simplicial más pequeño que los contiene.

Como entrada recibe:

- K, el número de iteraciones que se representarán en el complejo simplicial
- N, el número de procesos que modelará el complejo simplicial
- $\mathcal{M} = \langle W, R, L \rangle$, el modelo de Kripke de referencia

Como resultado obtiene:

- $\mathcal{S}$, el conjunto de simplejos que representa el modelo de Kripke
- $\mathcal{C}$, el complejo simplicial mímino que contiene a $\mathcal{S}$

Argumentemos que el algoritmo 9 es correcto. En el ciclo de las lineas 9-16 se crea la vista correspondiente a un agente a en una iteración i respecto a todos los agentes a' de la ejecución de uno de nuestros protocolos de estudio. En el ciclo de las lineas 8-17 se hace el proceso para todas las iteraciones del protocolo, construyendo así la vista completa para un agente a y agregándolo al simplejo σ. En el ciclo de las líneas 6-19 se repite el proceso para todos los agentes a, agregando a σ al conjunto $\mathcal{S}$ y a su conjunto potencia al conjunto $\mathcal{C}$. Todo el proceso se hace para cada mundo del modelo de Kripke $\mathcal{M}$ en el ciclo de las lineas 6-19 por lo que en el par $\langle \mathcal{S}, \mathcal{C} \rangle$ que se regresa en la línea 23, $\mathcal{S}$ es el conjunto de simplejos correspondientes a los mundos de $\mathcal{M}$ y $\mathcal{C}$ es el complejo simplicial mínimo que contiene a $\mathcal{S}$.

[9] Sobrecargamos el uso de los corchetes para indicar las posiciones en la vista siendo $col(a')$ la columna correspondiente al agente a' comenzando en 0.

Algoritmo 9 Algoritmo para calcular los simplejos que representa un modelo de Kripke etiquetado por el algoritmo 7, así como el complejo simplicial más pequeño que los contiene.

```
1: function SIMPLEJOS(K, N, M = ⟨W, R, L⟩)
2:     S ← ∅
3:     C ← ∅
4:     for all w ∈ W do
5:         σ = ∅
6:         for all a ∈ Agentes do
7:             Sea vista_a una vista para el agente a de longitud (K + 1) × N
8:             for all i ∈ {0, ..., k} do
9:                 for all a' ∈ Agentes do
10:                    Sea v tal que p_{0,a',v} ∈ L(w)
11:                    if M, w ⊨ □_a(p_{i,a',v}) then
12:                        vista_a[i × n + col(a')] ← v
13:                    else
14:                        vista_a[i × n + col(a')] ← ⊥
15:                    end if
16:                end for
17:            end for
18:            σ ← σ ∪ {⟨a, vista_a⟩}
19:        end for
20:        S ← S ∪ {σ}
21:        C ← C ∪ ℘(σ)
22:    end for
23:    return ⟨S, C⟩
24: end function
```

3.7. Ejemplos de transformación

Como ejemplo mostraremos la transformación de la proyección sobre los procesos a (gris oscuro) y b (gris claro) de la subdivisión cromática de tres procesos en el modelo *Iterated Immediate Snapshots* que se muestra en la figura 3.1 y también la transformación del modelo de Kripke al complejo simplicial así como los simplejos que modela el modelo de Kripke que se muestra en la figura 3.1.

3.7.1. Conjunto de simplejos a modelo de Kripke

Nuestro complejo simplicial de inicio, $\mathcal{C}$, se muestra en la figura 3.1, del cual solamente modelaremos $\mathcal{S} = facetas(\mathcal{C})$.

Primero asignamos un identificador a cada simplejo a modelar, para identificar los mundos correspondientes en el modelo de Kripke. En nuestro caso transformaremos solamente las facetas del complejo simplicial, es decir, a las aristas del mismo. Así, los identificadores quedan como se muestran en la figura 3.2.

El siguiente paso es construir los mundos que corresponden a los simplejos a modelar, como se muestra en la figura 3.3. Cada uno de ellos tiene el mismo identificador que en el complejo simplicial.

El siguiente paso es construir las relaciones entre los mundos construidos, como se muestra en la figura 3.4. Observemos que los mundos 6 y 7, correspondientes a las facetas 6 y 7 en el complejo simplicial están relacionados desde el punto de vista del agente a que es justamente el color del vértice (gris oscuro) que ambas facetas tienen en común. La misma regla se usa para construir las relaciones restantes.

Por último se construye el etiquetamiento de los mundos fusionando las vistas de los procesos que se representan en los simplejos representados, en este caso las facetas. Notemos que los mundos 6, 7 y 8 tienen el etiquetamiento $[0, 1, \perp, 0, 1, \perp]$ correspondiente a las proposiciones $\{p_{0,a,0}, p_{0,b,1}, p_{0,c,\perp}, p_{1,a,0}, p_{1,b,1}, p_{1,c,\perp}\}$. Esto es porque las vistas de los procesos a y b correspondientemente: $[0, \perp, \perp, 0, 1, \perp]$ y $[\perp, 1, \perp, \perp, 1, \perp]$ para el mundo mundo 6, $[0, \perp, \perp, 0, 1, \perp]$ y $[\perp, 1, \perp 0, 1, \perp]$ para el mundo mundo 7, y $[0, \perp, \perp, 0, \perp, \perp]$ y $[\perp, 1, \perp, 0, 1, \perp]$ para el mundo mundo 8, se fusionan todas de al etiquetamiento $[0, 1, \perp, 0, 1, \perp]$. Similarmente para los mundos $\{9, 10, 11\}$, $\{3, 4, 5\}$ y $\{0, 1, 2\}$. El resultado se muestra en la figura 3.5.

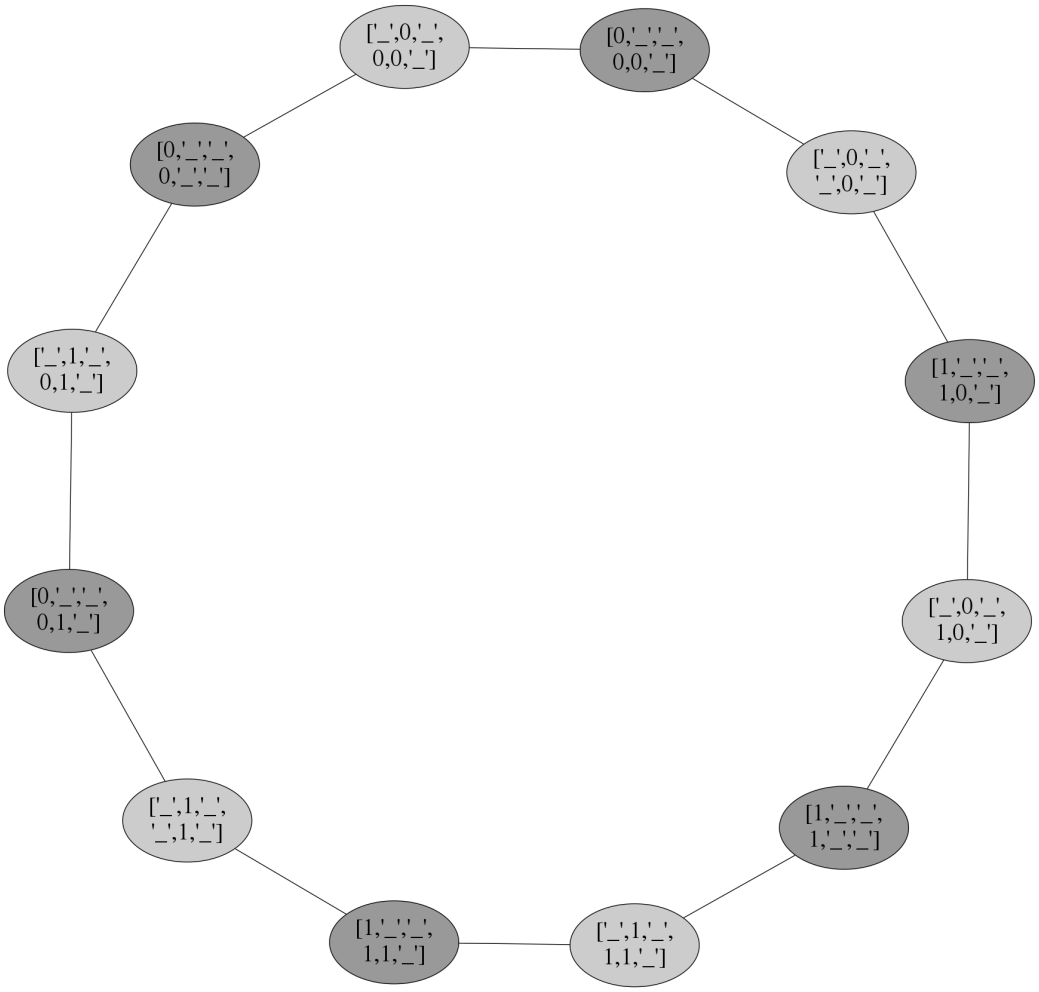

Figura 3.1: Proyección sobre los procesos a (gris oscuro) y b (gris claro) de la subdivisión cromática de tres procesos en nuestros protocolos de estudio.

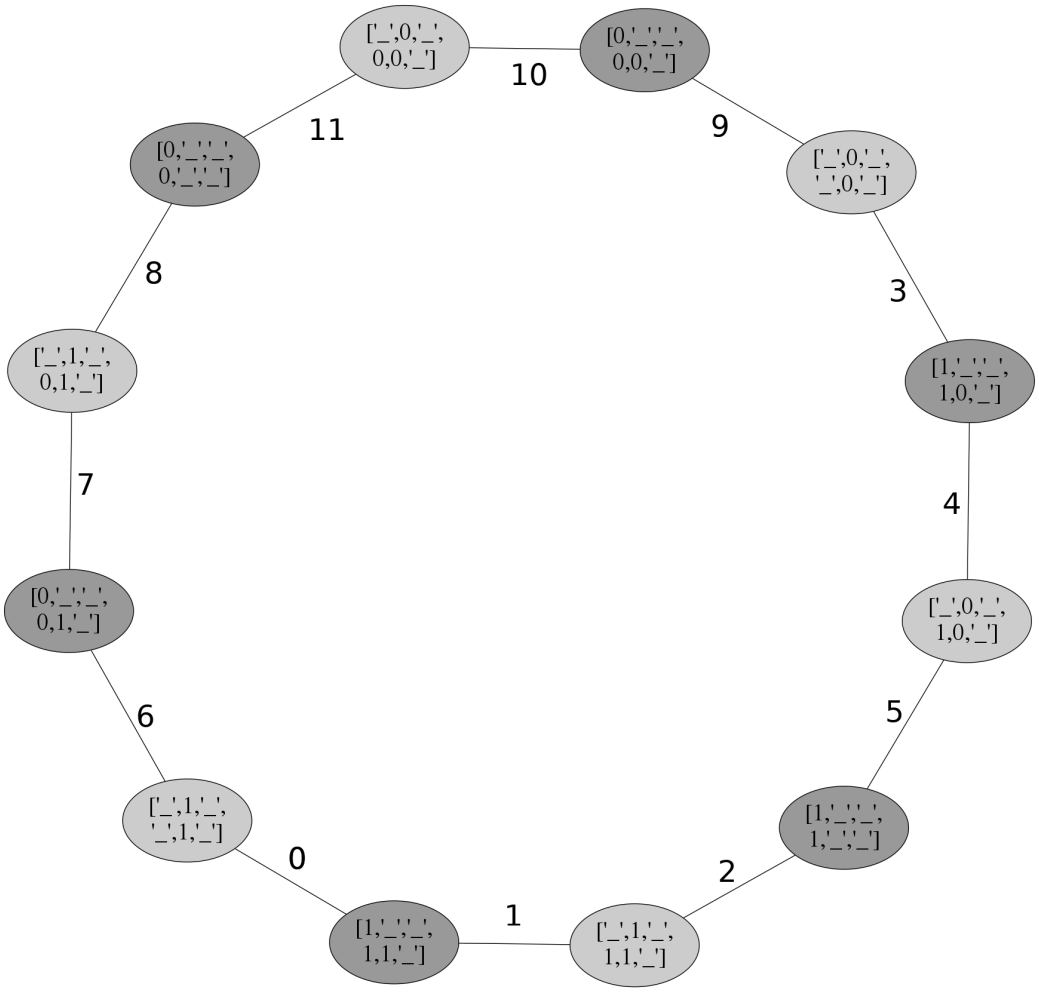

Figura 3.2: Complejo de la figura 3.1 con identificadores de sus facetas.

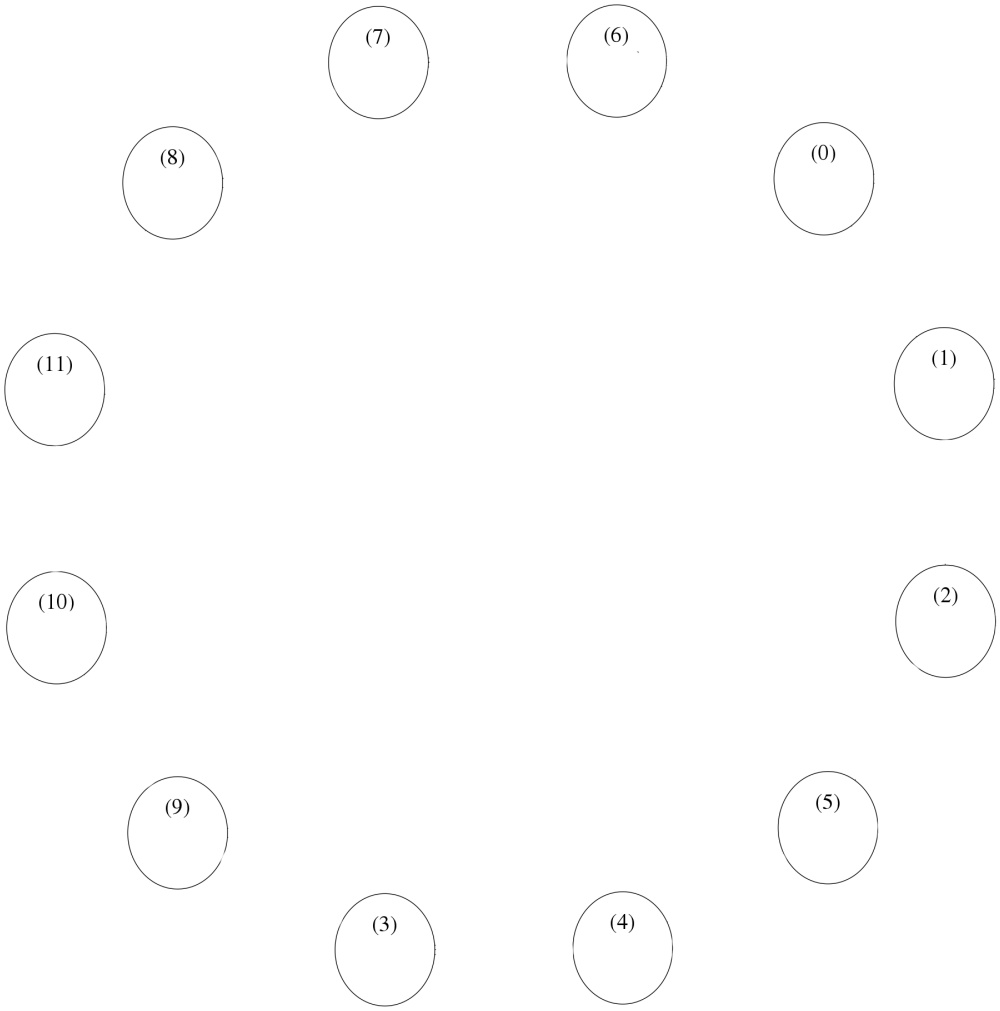

Figura 3.3: Mundos en biyección a los simplejos identificados en la figura 3.2.

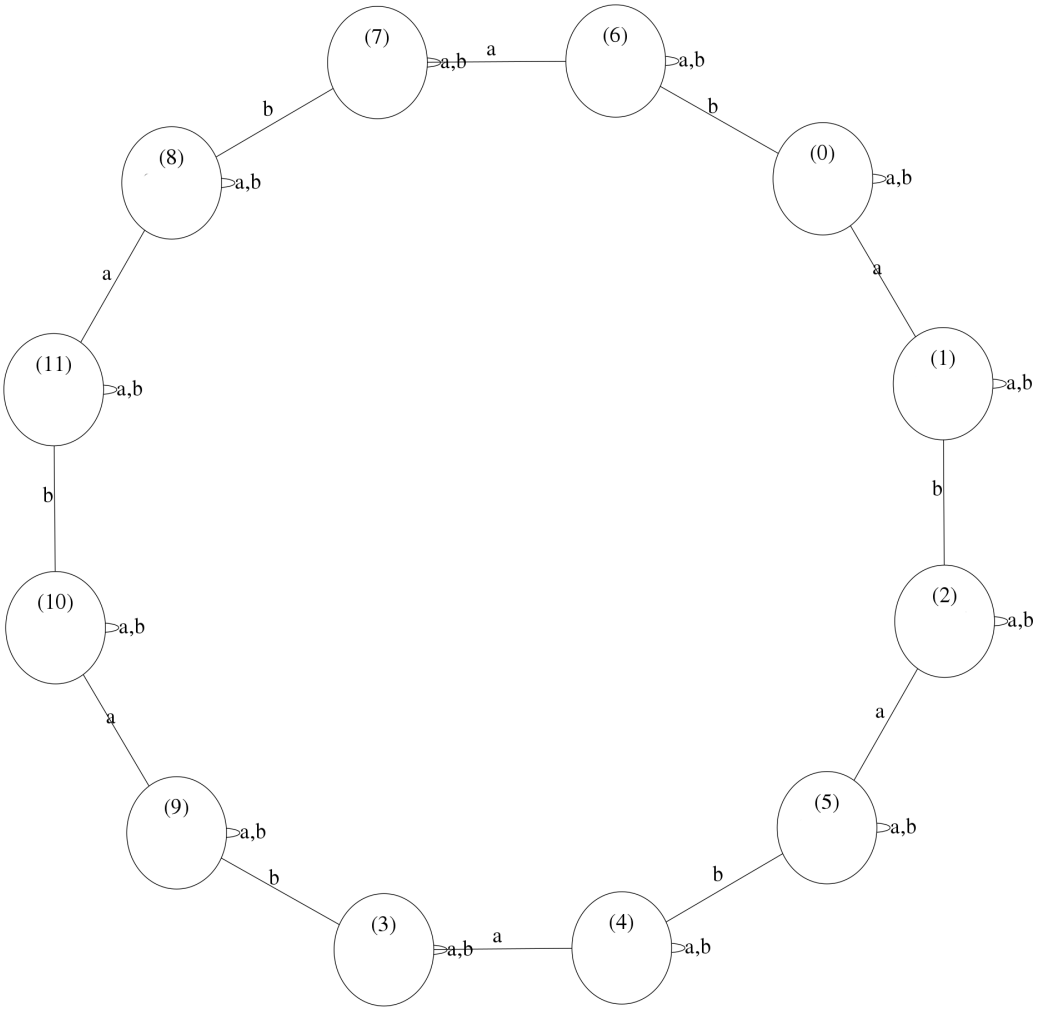

Figura 3.4: Relación de los mundos en la figura 3.3.

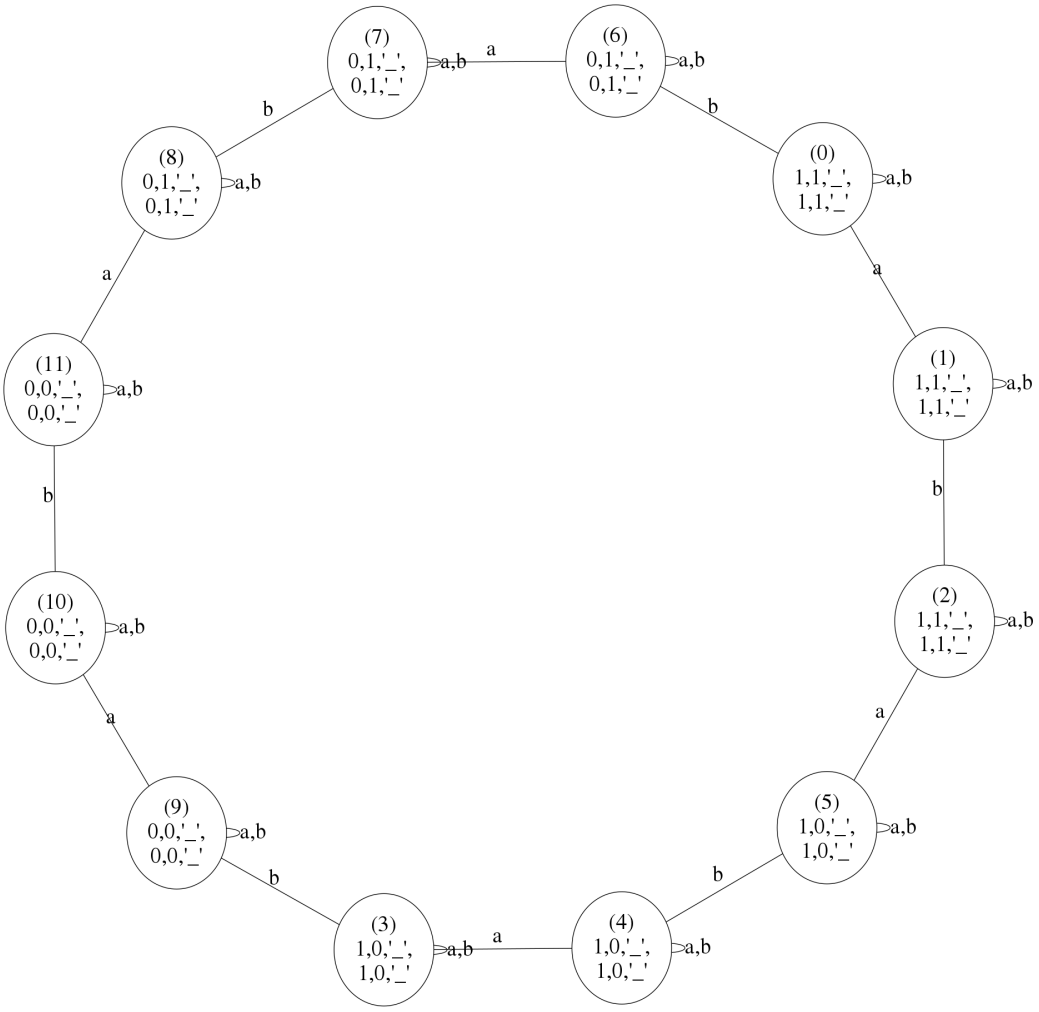

Figura 3.5: Modelo de Kripke correspondiente al complejo simplicial de la figura 3.1.

3.7.2. Modelo de Kripke a conjunto de simplejos

Debido a que el algoritmo 9 actualiza en cada iteración del ciclo que comienza en la línea 4 a $\mathcal{S}$ y $\mathcal{C}$, mostraremos a detalle solo la primera iteración; luego solo mostraremos como van cambiando las variables $\mathcal{S}$, y $\mathcal{C}$ hasta que obtenemos el resultado $\langle \mathcal{S}, \mathcal{C} \rangle$.

A partir del modelo de Kripke en la figura 3.5, tomemos los mundos $w \in W$ en el orden natural de su identificador. Tomemos en cuenta la figura

3.6 que muestra la vecindad del mundo w_0.

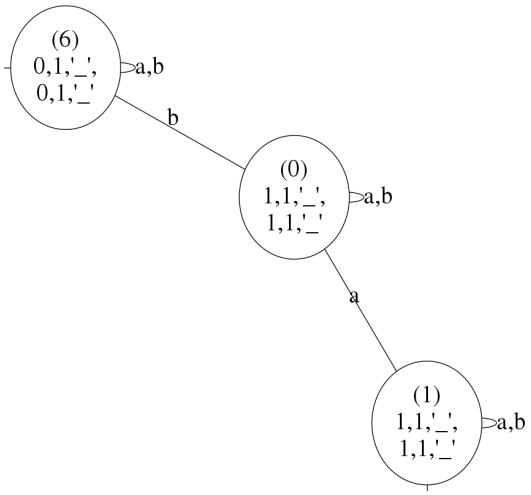

Figura 3.6: Vecindad del mundo 0 en el modelo de Kripke de la figura 3.1.

Para construir el vértice del agente a:

Cuando la variable a representa al agente a, $i = 0$ y la variable a' representa al agente a, $v = 1$ y como $\mathcal{M}, w_0 \vDash \Box_a p_{0,a,1}$ entonces $vista_a = [1, \bot, \bot, \bot, \bot, \bot]$.

Cuando la variable a representa al agente a, $i = 0$ y la variable a' representa al agente b, $v = 1$ y como $\mathcal{M}, w_0 \vDash \Box_a p_{0,b,1}$ entonces $vista_a = [1, 1, \bot, \bot, \bot, \bot]$.

Cuando la variable a representa al agente a, $i = 1$ y la variable a' representa al agente a, $v = 1$ y como $\mathcal{M}, w_0 \vDash \Box_a p_{0,a,1}$ entonces $vista_a = [1, 1, \bot, 1, \bot, \bot]$.

Cuando la variable a representa al agente a, $i = 1$ y la variable a' representa al agente b, $v = 1$ y como $\mathcal{M}, w_0 \vDash \Box_a p_{0,b,1}$ entonces $vista_a = [1, 1, \bot, 1, 1, \bot]$.

En este punto agregamos al vértice correspondiente al agente a a σ de tal forma que
$\sigma \leftarrow \{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle\}$.

Notemos que en la vista calculada para el agente a aparece el valor de la entrada del proceso b. Aún cuando en el complejo simplicial original estas situaciones no pasan, dicha situación, lejos de ser un error, es normal pues el agente a al haber leído el valor de b en la iteración 1, sabe que ese valor

también es el de su entrada.

Para construir el vértice del agente b:

Cuando $a \leftarrow b$, $i \leftarrow 0$ y $a' \leftarrow a$, $v \leftarrow 1$ y como $\mathcal{M}, w_0 \nvDash \Box_b p_{0,a,1}$ entonces $vista_a = [\bot, \bot, \bot, \bot, \bot, \bot]$.

Cuando $a \leftarrow b$, $i \leftarrow 0$ y $a' \leftarrow b$, $v \leftarrow 1$ y como $\mathcal{M}, w_0 \vDash \Box_b p_{0,b,1}$ entonces $vista_a = [\bot, 1, \bot, \bot, \bot, \bot]$.

Cuando $a \leftarrow b$, $i \leftarrow 1$ y $a' \leftarrow a$, $v \leftarrow 1$ y como $\mathcal{M}, w_0 \nvDash \Box_b p_{0,a,1}$ entonces $vista_a = [\bot, 1, \bot, \bot, \bot, \bot]$.

Cuando $a \leftarrow b$, $i \leftarrow 1$ y $a' \leftarrow b$, $v \leftarrow 1$ y como $\mathcal{M}, w_0 \vDash \Box_b p_{0,b,1}$ entonces $vista_a = [\bot, 1, \bot, \bot, 1, \bot]$.

En este punto agregamos el nuevo vértice a σ por lo que
$\sigma = \{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}$.

Luego, se agrega σ a $\mathcal{S}$ y 2^σ a $\mathcal{C}$ actualizando las variables como se muestra a continuación.
$\mathcal{S} = \{\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle\}, \{\langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\},$
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}\}$.

Tomando en cuenta la figura 3.7, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 1$
$\sigma = \{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\},$
$\mathcal{S} = \{\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\},$
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle\}, \{\langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\},$
$\{\langle b, [1, 1, \bot, 1, 1, \bot]\rangle\},$
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\},$
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}\}$.

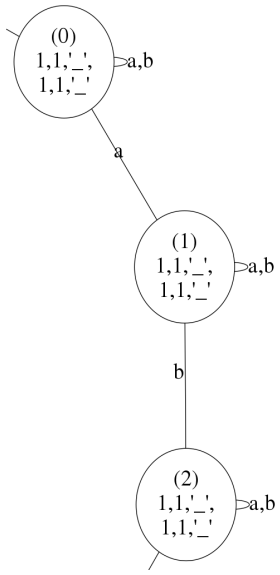

Figura 3.7: Vecindad del mundo 1 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.8, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 2$
$\sigma = \{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}$,
$\mathcal{S} = \{\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}$,
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}$,
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle\}, \{\langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}$,
$\{\langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}, \{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle\}$,
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}$,
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}$,
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}\}$.

Tomando en cuenta la figura 3.9, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 3$
$\sigma = \{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle, \langle b, [\bot, 0, \bot, \bot, 0, \bot]\rangle\}$,
$\mathcal{S} = \{\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}$,
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}$,
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}$,
$\{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle, \langle b, [\bot, 0, \bot, \bot, 0, \bot]\rangle\}\}$
y

$\mathcal{C} = \{\emptyset, \{\langle a, [1,1,\bot,1,1,\bot]\rangle\}, \{\langle b, [\bot,1,\bot,\bot,1,\bot]\rangle\},$
$\{\langle b, [1,1,\bot,1,1,\bot]\rangle\}, \{\langle a, [1,\bot,\bot,1,\bot,\bot]\rangle\},$
$\{\langle a, [1,0,\bot,1,0,\bot]\rangle\}, \{\langle b, [\bot,0,\bot,\bot,0,\bot]\rangle\},$
$\{\langle a, [1,1,\bot,1,1,\bot]\rangle, \langle b, [\bot,1,\bot,\bot,1,\bot]\rangle\},$
$\{\langle a, [1,1,\bot,1,1,\bot]\rangle, \langle b, [1,1,\bot,1,1,\bot]\rangle\},$
$\{\langle a, [1,\bot,\bot,1,\bot,\bot]\rangle, \langle b, [1,1,\bot,1,1,\bot]\rangle\},$
$\{\langle a, [1,0,\bot,1,0,\bot]\rangle, \langle b, [\bot,0,\bot,\bot,0,\bot]\rangle\}\}.$

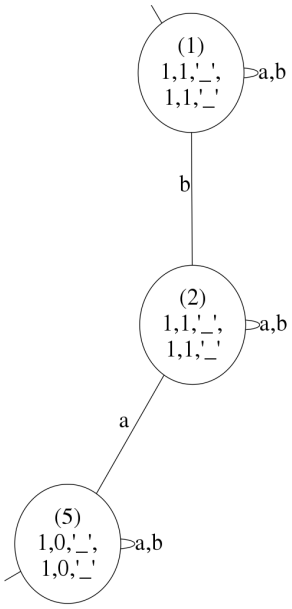

Figura 3.8: Vecindad del mundo 2 en el modelo de Kripke de la figura 3.1.

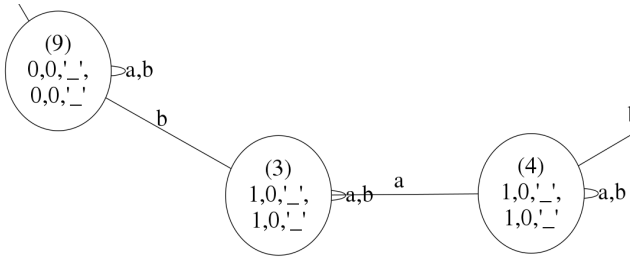

Figura 3.9: Vecindad del mundo 3 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.10, al final de la iteración del ciclo de las

líneas 4-22 en el que $w = 4$
$\sigma = \{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}$,
$\mathcal{S} = \{\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle\}, \{\langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}$,
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}\}$.

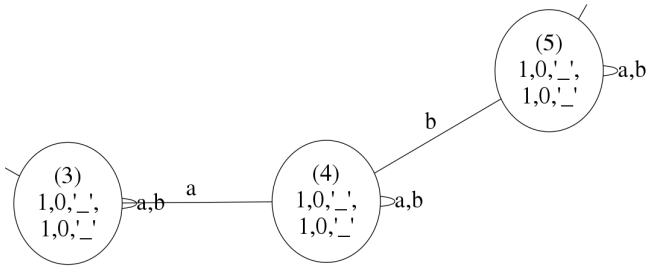

Figura 3.10: Vecindad del mundo 4 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.11, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 5$
$\sigma = \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}$,
$\mathcal{S} = \{\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}$,
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}\}$
y

$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle\}, \{\langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\},$
$\{\langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\},$
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}\}.$

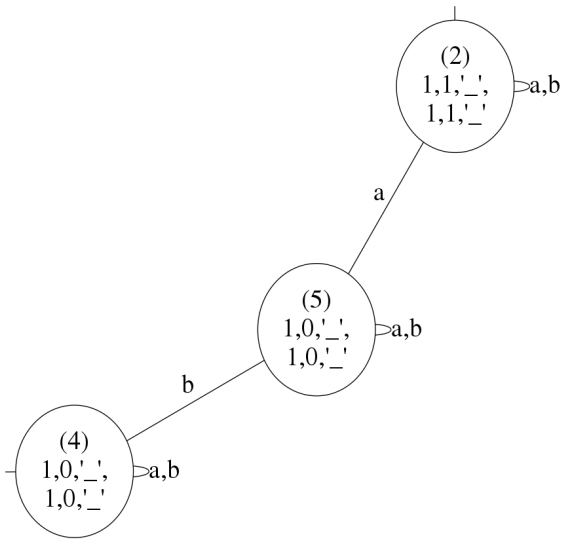

Figura 3.11: Vecindad del mundo 5 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.12, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 6$
$\sigma = \{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\mathcal{S} = \{\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}\}$

y
$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle\}, \{\langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\},$
$\{\langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\},$
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle\},$
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}\}.$

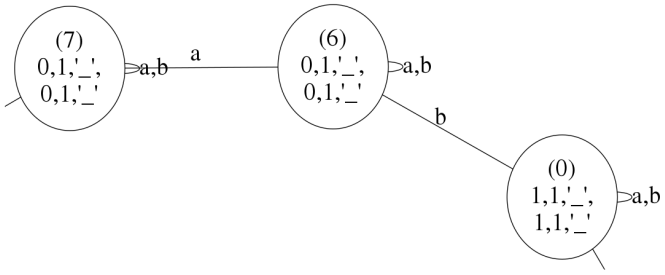

Figura 3.12: Vecindad del mundo 6 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.13, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 7$
$\sigma = \{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [0, 1, \perp, 0, 1, \perp]\rangle\},$
$\mathcal{S} = \{\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [0, 1, \perp, 0, 1, \perp]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\},$

$\{\langle a,[1,0,\bot,1,0,\bot]\rangle\},\{\langle b,[\bot,0,\bot,\bot,0,\bot]\rangle\},$
$\{\langle b,[1,0,\bot,1,0,\bot]\rangle\},\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle\},$
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle\},\{\langle b,[0,1,\bot,0,1,\bot]\rangle\},$
$\{\langle a,[1,1,\bot,1,1,\bot]\rangle,\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\},$
$\{\langle a,[1,1,\bot,1,1,\bot]\rangle,\langle b,[1,1,\bot,1,1,\bot]\rangle\},$
$\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle,\langle b,[1,1,\bot,1,1,\bot]\rangle\},$
$\{\langle a,[1,0,\bot,1,0,\bot]\rangle,\langle b,[\bot,0,\bot,\bot,0,\bot]\rangle\},$
$\{\langle a,[1,0,\bot,1,0,\bot]\rangle,\langle b,[1,0,\bot,1,0,\bot]\rangle\},$
$\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle,\langle b,[1,0,\bot,1,0,\bot]\rangle\},$
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle,\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\},$
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle,\langle b,[0,1,\bot,0,1,\bot]\rangle\}\}.$

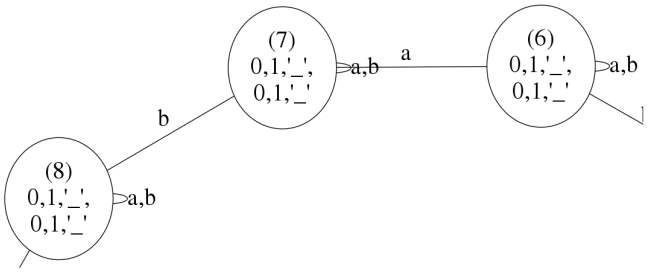

Figura 3.13: Vecindad del mundo 7 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.14, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 8$
$\sigma = \{\langle a,[0,\bot,\bot,0,\bot,\bot]\rangle,\langle b,[0,1,\bot,0,1,\bot]\rangle\},$
$\mathcal{S} = \{\{\langle a,[1,1,\bot,1,1,\bot]\rangle,\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\},$
$\{\langle a,[1,1,\bot,1,1,\bot]\rangle,\langle b,[1,1,\bot,1,1,\bot]\rangle\},$
$\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle,\langle b,[1,1,\bot,1,1,\bot]\rangle\},$
$\{\langle a,[1,0,\bot,1,0,\bot]\rangle,\langle b,[\bot,0,\bot,\bot,0,\bot]\rangle\},$
$\{\langle a,[1,0,\bot,1,0,\bot]\rangle,\langle b,[1,0,\bot,1,0,\bot]\rangle\},$
$\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle,\langle b,[1,0,\bot,1,0,\bot]\rangle\},$
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle,\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\},$
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle,\langle b,[0,1,\bot,0,1,\bot]\rangle\},$
$\{\langle a,[0,\bot,\bot,0,\bot,\bot]\rangle,\langle b,[0,1,\bot,0,1,\bot]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset,\{\langle a,[1,1,\bot,1,1,\bot]\rangle\},\{\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\},$
$\{\langle b,[1,1,\bot,1,1,\bot]\rangle\},\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle\},$
$\{\langle a,[1,0,\bot,1,0,\bot]\rangle\},\{\langle b,[\bot,0,\bot,\bot,0,\bot]\rangle\},$

$\{\langle b, [1, 0, \bot, 1, 0, \bot]\rangle\}, \{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle\},$
$\{\langle a, [0, 1, \bot, 0, 1, \bot]\rangle\}, \{\langle b, [0, 1, \bot, 0, 1, \bot]\rangle\},$
$\{\langle a, [0, \bot, \bot, 0, \bot, \bot]\rangle\},$
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\},$
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\},$
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\},$
$\{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle, \langle b, [\bot, 0, \bot, \bot, 0, \bot]\rangle\},$
$\{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle, \langle b, [1, 0, \bot, 1, 0, \bot]\rangle\},$
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 0, \bot, 1, 0, \bot]\rangle\},$
$\{\langle a, [0, 1, \bot, 0, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\},$
$\{\langle a, [0, 1, \bot, 0, 1, \bot]\rangle, \langle b, [0, 1, \bot, 0, 1, \bot]\rangle\},$
$\{\langle a, [0, \bot, \bot, 0, \bot, \bot]\rangle, \langle b, [0, 1, \bot, 0, 1, \bot]\rangle\}\}.$

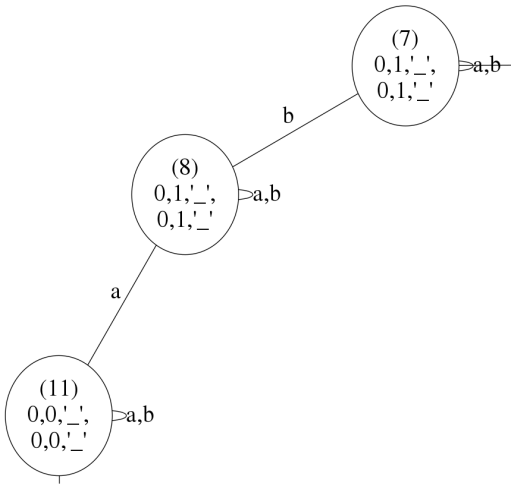

Figura 3.14: Vecindad del mundo 8 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.15, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 9$
$\sigma = \{\langle a, [0, 0, \bot, 0, 0, \bot]\rangle, \langle b, [\bot, 0, \bot, \bot, 0, \bot]\rangle\},$
$\mathcal{S} = \{\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\},$
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\},$
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\},$
$\{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle, \langle b, [\bot, 0, \bot, \bot, 0, \bot]\rangle\},$
$\{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle, \langle b, [1, 0, \bot, 1, 0, \bot]\rangle\},$
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 0, \bot, 1, 0, \bot]\rangle\},$

$\{\langle a,[0,1,\bot,0,1,\bot]\rangle,\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\}$,
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle,\langle b,[0,1,\bot,0,1,\bot]\rangle\}$,
$\{\langle a,[0,\bot,\bot,0,\bot,\bot]\rangle,\langle b,[0,1,\bot,0,1,\bot]\rangle\}$,
$\{\langle a,[0,0,\bot,0,0,\bot]\rangle,\langle b,[\bot,0,\bot,\bot,0,\bot]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset,\{\langle a,[1,1,\bot,1,1,\bot]\rangle\},\{\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\}$,
$\{\langle b,[1,1,\bot,1,1,\bot]\rangle\},\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle\}$,
$\{\langle a,[1,0,\bot,1,0,\bot]\rangle\},\{\langle b,[\bot,0,\bot,\bot,0,\bot]\rangle\}$,
$\{\langle b,[1,0,\bot,1,0,\bot]\rangle\},\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle\}$,
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle\},\{\langle b,[0,1,\bot,0,1,\bot]\rangle\}$,
$\{\langle a,[0,\bot,\bot,0,\bot,\bot]\rangle\},\{\langle a,[0,0,\bot,0,0,\bot]\rangle\}$,
$\{\langle a,[1,1,\bot,1,1,\bot]\rangle,\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\}$,
$\{\langle a,[1,1,\bot,1,1,\bot]\rangle,\langle b,[1,1,\bot,1,1,\bot]\rangle\}$,
$\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle,\langle b,[1,1,\bot,1,1,\bot]\rangle\}$,
$\{\langle a,[1,0,\bot,1,0,\bot]\rangle,\langle b,[\bot,0,\bot,\bot,0,\bot]\rangle\}$,
$\{\langle a,[1,0,\bot,1,0,\bot]\rangle,\langle b,[1,0,\bot,1,0,\bot]\rangle\}$,
$\{\langle a,[1,\bot,\bot,1,\bot,\bot]\rangle,\langle b,[1,0,\bot,1,0,\bot]\rangle\}$,
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle,\langle b,[\bot,1,\bot,\bot,1,\bot]\rangle\}$,
$\{\langle a,[0,1,\bot,0,1,\bot]\rangle,\langle b,[0,1,\bot,0,1,\bot]\rangle\}$,
$\{\langle a,[0,\bot,\bot,0,\bot,\bot]\rangle,\langle b,[0,1,\bot,0,1,\bot]\rangle\}$,
$\{\langle a,[0,0,\bot,0,0,\bot]\rangle,\langle b,[\bot,0,\bot,\bot,0,\bot]\rangle\}\}$.

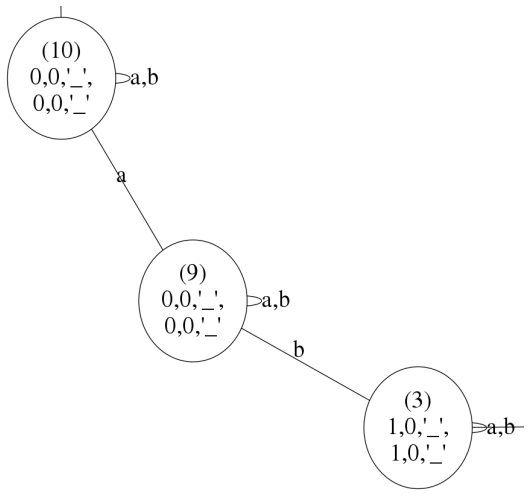

Figura 3.15: Vecindad del mundo 9 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.16, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 10$
$\sigma = \{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle, \langle b, [0, 0, \perp, 0, 0, \perp]\rangle\}$,
$\mathcal{S} = \{\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}$,
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}$,
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [0, 1, \perp, 0, 1, \perp]\rangle\}$,
$\{\langle a, [0, \perp, \perp, 0, \perp, \perp]\rangle, \langle b, [0, 1, \perp, 0, 1, \perp]\rangle\}$,
$\{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle, \langle b, [0, 0, \perp, 0, 0, \perp]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle\}, \{\langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\}$,
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle\}, \{\langle b, [0, 1, \perp, 0, 1, \perp]\rangle\}$,
$\{\langle a, [0, \perp, \perp, 0, \perp, \perp]\rangle\}, \{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle\}$,
$\{\langle b, [0, 0, \perp, 0, 0, \perp]\rangle\}, \{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}$,
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}$,
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\}$,
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [0, 1, \perp, 0, 1, \perp]\rangle\}$,
$\{\langle a, [0, \perp, \perp, 0, \perp, \perp]\rangle, \langle b, [0, 1, \perp, 0, 1, \perp]\rangle\}$,
$\{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\}$,
$\{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle, \langle b, [0, 0, \perp, 0, 0, \perp]\rangle\}\}$.

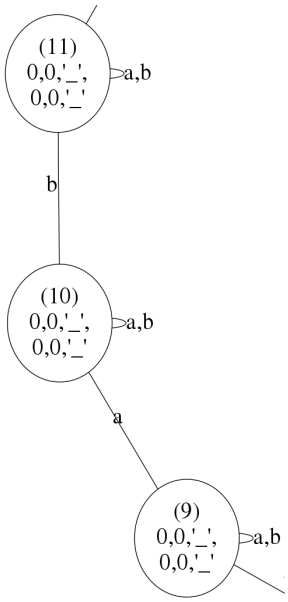

Figura 3.16: Vecindad del mundo 10 en el modelo de Kripke de la figura 3.1.

Tomando en cuenta la figura 3.17, al final de la iteración del ciclo de las líneas 4-22 en el que $w = 11$
$\sigma = \{\langle a, [0, \bot, \bot, 0, \bot, \bot]\rangle, \langle b, [0, 0, \bot, 0, 0, \bot]\rangle\}$,
$\mathcal{S} = \{\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}$,
$\{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}$,
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}$,
$\{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle, \langle b, [\bot, 0, \bot, \bot, 0, \bot]\rangle\}$,
$\{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle, \langle b, [1, 0, \bot, 1, 0, \bot]\rangle\}$,
$\{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle, \langle b, [1, 0, \bot, 1, 0, \bot]\rangle\}$,
$\{\langle a, [0, 1, \bot, 0, 1, \bot]\rangle, \langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}$,
$\{\langle a, [0, 1, \bot, 0, 1, \bot]\rangle, \langle b, [0, 1, \bot, 0, 1, \bot]\rangle\}$,
$\{\langle a, [0, \bot, \bot, 0, \bot, \bot]\rangle, \langle b, [0, 1, \bot, 0, 1, \bot]\rangle\}$,
$\{\langle a, [0, 0, \bot, 0, 0, \bot]\rangle, \langle b, [\bot, 0, \bot, \bot, 0, \bot]\rangle\}$,
$\{\langle a, [0, 0, \bot, 0, 0, \bot]\rangle, \langle b, [0, 0, \bot, 0, 0, \bot]\rangle\}$,
$\{\langle a, [0, \bot, \bot, 0, \bot, \bot]\rangle, \langle b, [0, 0, \bot, 0, 0, \bot]\rangle\}\}$
y
$\mathcal{C} = \{\emptyset, \{\langle a, [1, 1, \bot, 1, 1, \bot]\rangle\}, \{\langle b, [\bot, 1, \bot, \bot, 1, \bot]\rangle\}$,
$\{\langle b, [1, 1, \bot, 1, 1, \bot]\rangle\}, \{\langle a, [1, \bot, \bot, 1, \bot, \bot]\rangle\}$,
$\{\langle a, [1, 0, \bot, 1, 0, \bot]\rangle\}, \{\langle b, [\bot, 0, \bot, \bot, 0, \bot]\rangle\}$,

$\{\langle b, [1, 0, \perp, 1, 0, \perp]\rangle\}, \{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle\},$
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle\}, \{\langle b, [0, 1, \perp, 0, 1, \perp]\rangle\},$
$\{\langle a, [0, \perp, \perp, 0, \perp, \perp]\rangle\}, \{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle\},$
$\{\langle b, [0, 0, \perp, 0, 0, \perp]\rangle\}, \{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle a, [1, 1, \perp, 1, 1, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 1, \perp, 1, 1, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\},$
$\{\langle a, [1, 0, \perp, 1, 0, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [1, \perp, \perp, 1, \perp, \perp]\rangle, \langle b, [1, 0, \perp, 1, 0, \perp]\rangle\},$
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [\perp, 1, \perp, \perp, 1, \perp]\rangle\},$
$\{\langle a, [0, 1, \perp, 0, 1, \perp]\rangle, \langle b, [0, 1, \perp, 0, 1, \perp]\rangle\},$
$\{\langle a, [0, \perp, \perp, 0, \perp, \perp]\rangle, \langle b, [0, 1, \perp, 0, 1, \perp]\rangle\},$
$\{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle, \langle b, [\perp, 0, \perp, \perp, 0, \perp]\rangle\},$
$\{\langle a, [0, 0, \perp, 0, 0, \perp]\rangle, \langle b, [0, 0, \perp, 0, 0, \perp]\rangle\},$
$\{\langle a, [0, \perp, \perp, 0, \perp, \perp]\rangle, \langle b, [0, 0, \perp, 0, 0, \perp]\rangle\}\}.$

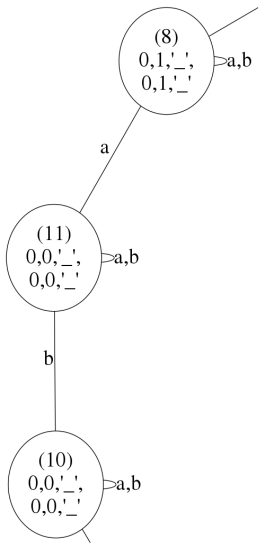

Figura 3.17: Vecindad del mundo 11 en el modelo de Kripke de la figura 3.1.

Observemos que $\mathcal{S}$ contiene las facetas del complejo simplicial en la figura 3.1 cuyas vistas difieren únicamente en las entradas de los procesos. Como ya se comentó, esto es correcto y que por tanto, $\mathcal{C}$ es el complejo simplicial que se representa en la misma figura, concluyendo así el proceso de transformación.

3.8. Observaciones

3.8.1. Vistas de vértices y etiquetamientos de mundos

Una diferencia, a primera vista inquietante, entre las estructuras que usa cada enfoque es la forma en la que cada una representa la información. Los complejos simpliciales representan el conocimiento de los procesos por medio de la vista de cada vértice. En contraste, los modelos de Kripke representan la información de lo que los procesos ven a través de los etiquetameintos de sus mundos.

Recordemos que en esencia los complejos simpliciales son conjuntos de conjuntos, y que particularmente los vértices de los complejos simpliciales que nos ocupan se pueden identificar por un color y una vista. Debido a lo anterior, tener dos vértices con el mismo color y vista en un complejo simplicial no tendría sentido pues realmente nos referimos mismo.

Por otro lado, en un modelo de Kripke los etiquetamientos de los mundos que se generan a partir del complejo simplicial, corresponden a la acumulación de la información de los agentes que corresponden a algún proceso del complejo simplicial. Por ejemplo, en el modelo de cómputo *Iterated Immediate Snapshots*, una ejecución en la que los procesos a, b y c tengan entradas v_1, v_2 y v_3, respectivamente, y todos ellos terminen satisfactoriamente, sin importar la forma en que concurran, dejarán en su memoria respectiva al final de la iteración, la misma información, la cual se plasmará en el etiquetamiento de los mundos del modelo de Kripke.

El que haya mundos en un modelo de Kripke con el mismo etiquetamiento es normal, ya que las relaciones de indistinguibilidad entre mundos desde la perspectiva de cada agente le dan la capacidad de reconocer lo que cada uno ve (si el modelo no es epistémico) o sabe. Por ejemplo, en el modelo de Kripke de la figura 3.5, los mundos 0 y 1 y 2 tienen el mismo etiquetamiento. Sin embargo, en el mundo 0, el agente b **no sabe** $p_{0,a,1}$ ni $p_{1,a,1}$, aunque dichas proposiciones estén en el etiquetamiento del mundo 0, pues el agente b no distingue entre el nundo 0 y el 6 y en el mundo 6, $p_{0,a,1}$ y $p_{1,a,1}$ son falsas. Por otro lado, en el mundo 1, el agente b **sí sabe** $p_{0,a,1}$ y $p_{1,a,1}$ pues en los mundos 1 y 2, indistinguibles para el agente b, ambas proposiciones son verdaderas.

Aún cuando existe la diferencia mencionada en un principio, ésta simplemente es el resultado de que la información de los procesos en los complejos simpliciales se representa en cada uno de los vértices mientras que en un modelo de Kripke un mundo representa realmente a un conjunto de vértices,

y por tanto expresa el conocimiento distribuido entre los procesos correspondientes a los vértices de ese conjunto.

3.8.2. Las facetas de complejos simpliciales puros, caracterizados en 3.1 inducen modelos epistémicos

En esta sección, se demuestra formalmente que el procedimiento en el algoritmo 8, induce relaciones de equivalencia desde el punto de vista de los agentes del modelo de Kripke que se genera si es que se parte de un conjunto de simplejos donde todos ellos tienen la misma dimensión y los mismos colores.

Teorema 3.8.1. *Sea $\mathcal{C}$ un complejo simplicial crómatico cuyos vértices se puedan identificar por medio de su color y su vista y que las vistas de todos sus vértices tengan el mismo tamaño. Sea $\mathcal{M} = \langle W, R, L\rangle$ el modelo de Kripke sobre el conjunto de agentes A que se genera con el algoritmo 8 a partir de $\mathcal{S} \subseteq \mathcal{C}$ un conjunto de simplejos con la misma dimensión y colores. Cada relación R_a con $R = \bigcup R_a,\ a \in A$ es una relación de equivalencia.*

Demostración. Demostraremos que cualquier relación R_a, $a \in A$ es reflexiva, simétrica y transitiva.

1) R_a es reflexiva

Como $\mathcal{S}$ solamente tiene como elementos simplejos de $\mathcal{C}$ con los mismos colores y dimensión, el modelo $\mathcal{M}$ producido por el proceso en el algoritmo 8 asegura que $A = colores(\mathcal{S})$, $\forall a \in A, \forall w \in W, \forall \sigma \in \mathcal{S}, wR_aw$ pues $A = colores(\sigma) = colores(\sigma \cap \sigma)$.

2) R_a es simétrica

Del segundo punto del procedimiento en 3.4 $w_1R_aw_2$ *sii* $a \in colores(s_1 \cap s_2)$ de tal forma que R_a es simétrica pues la intersección entre conjuntos es simétrica.

3) R_a es transitiva

Supongamos que existen $w_1, w_2, w_3 \in W$ correspondientes a las facetas $\sigma_1, \sigma_2, \sigma_3 \in \mathcal{S}$ tales que $w_1R_aw_2$ y $w_2R_aw_3$. Si $w_1R_aw_2$ entonces $a \in colores(\sigma_1 \cap \sigma_2)$ similarmente, si $w_2R_aw_3$ entonces $a \in colores(\sigma_2 \cap \sigma_3)$. Notemos que al ser $\mathcal{M}$ el modelo correspondiente a $\mathcal{S}$, el vértice v_a de color a en la intersección $\sigma_1 \cap \sigma_2$ y el vértice v_a de color a en la intersección $\sigma_2 \cap \sigma_3$ es el mimso, de tal forma que $v_a \in (\sigma_1 \cap \sigma_3)$ por lo que $a \in colores(\sigma_1 \cap \sigma_3)$ y entonces $w_1R_aw_3$. Por lo tanto R_a es reflexiva.

Por 1), 2) y 3), R_a es una relación de equivalencia $\forall a \in A$. Por tanto $\mathcal{M}$ es un modelo epistémico.

□

Como consecuencia podemos enunciar lo siguiente:

Corolario 3.8.2. *Sea $\mathcal{C}$ un complejo simplicial generado con el algoritmo 4. Si $\mathcal{M}$ es el modelo de Kripke generado con el algoritmo 8 a partir de las facetas de $\mathcal{C}$ entonces $\mathcal{M}$ es un modelo epistémico.*

Demostración. Por el teorema 3.3.1 $\mathcal{C}$ es puro, por tanto sus facetas tienen la misma dimensión y colores. Por el teorema 3.8.1 las relaciones de accesibilidad para cada agente que se generan con el algoritmo 8 son de equivalencia, por lo que $\mathcal{M}$ es un modelo epistémico. □

3.8.3. Conjuntos de simplejos, modelos de Kripke y bisimulación

Es importante decir que cuando no se puede hacer diferenciación entre los etiquetamientos de los mundos de una componente conexa de un modelo de Kripke, el proceso de bisimulación puede compactar el modelo de Kripke correspondiente a un complejo simplicial. Aun cuando el complejo simplicial puede modelar diferentes interacciones de concurrencia entre los procesos, desde el punto de vista epistémico, independientemente de la forma en que se dan dichas interacciones, es conocimiento común que solo se puede escribir un valor, de tal forma que si se supone que todos los procesos terminan la ejecución de la iteración del protocolo, el modelo epistémico se reduce a un solo mundo.

Por ejemplo, podemos suponer que hay una interacción concurrente entre tres procesos en el modelo *Iterated Immediate Snapshots* donde todos los procesos escriben solamente el valor 0. El complejo simplicial y modelo de Kripke correspondiente a las facetas del complejo simplicial se muestran en las figuras 3.18 y 3.19, respectivamente. Observemos que todos los mundos tienen el mismo etiquetamiento en el modelo de Kripke. Además, recordando cómo es el mecanismo de reducción bajo bisimulación, solo podemos generar una clase de equivalencia al inicio, pues todos los mundos tienen el mismo etiquetamiento, por lo que todos los mundos se compactan en uno solo como se muestra en la figura 3.20. Notemos que la reducción del modelo de Kripke

bajo bisimulación deja de suceder a medida que el modelo de Kripke represente procesos que tienen más de una posible entrada, ya que de esa forma los etiquetamientos de los mundos son distintos y se puede crear una partición inicial más fina para comenzar el proceso de minimización bajo bisimulación.

Tomemos en cuenta el el algoritmo 8. Podríamos construir un número exponencial[10] de posibles modelos de Kripke correspondientes a un solo complejo simplicial siendo cualquier subconjunto de simplejos, los mundos posibles de un modelo de Kripke. Cabe señalar que aunque todos ellos serían modelos de Kripke, no todos ellos serían modelos válidos para lógica epistémica, ya que las relaciones restringidas a cada agente deberían ser relaciones de equivalencia. Además, muchos de ellos podrían no tener sentido de ser analizados aun cuando puedan ser construidos. Pensemos en la proyección de la subdivisión cromática para tres procesos de la figura 3.21. Si modelamos solo las facetas, suponiendo que no hay fallas, el modelo de Kripke correspondiente se muestra en la figura 3.22. Otro modelo de interés que podríamos obtener es si suponemos que alguno de ellos puede fallar, generaríamos adicionalmente cuatro mundos como se muestra en la parte superior de la figura 3.23.

Otro escenario es que también se tome en cuenta el caso en el que ambos puedan fallar, que solamente agrega un mundo más al modelo anterior como se muestra en la figura 3.24. Es importante notar que todos los casos son modelos generados con los simplejos de las facetas de las proyecciones sobre dos, uno o cero procesos y consecuentemente los modelos no necesariamente son modelos epistémicos, pues en el modelo se representan diferentes dimensiones del complejo simplicial causando la pérdida de la reflexividad.

3.8.4. Vistas en el proceso de transformación

Podemos pensar que el proceso de transformación de simplejos a modelos de Kripke es reversible. Sin embargo, hay ocasiones en que el etiquetamiento inducido por la fusión de las vistas y las relaciones mismas que se inducen por el complejo simplicial hacen que sea imposible reconstruir las vistas originales de los simplejos a partir de los cuales construimos un modelo de Kripke. Tomemos en cuenta el modelo de Kripke de la figura 3.19. En él todos los mundos tienen el mismo etiquetamiento, y por tanto las vistas calculadas por el algoritmo 9 serán iguales para todos los vértices generados, incluso el complejo simplicial que se genera a partir del modelo mencionado cons-

[10] En el tamaño del complejo simplicial.

ta solamente de un simplejo cuyos tres vértices tienen la misma vista. Esta situación se puede evitar forzando a que en cada componte conexa pueda haber una diferenciación entre el etiquetamiento de mundos. Una condición suficiente, mas no necesaria para ello, es agregar a los vértices de los simplejos que queremos modelar dentro de los simplejos a partir de los cuales construiremos el modelo de Kripke.

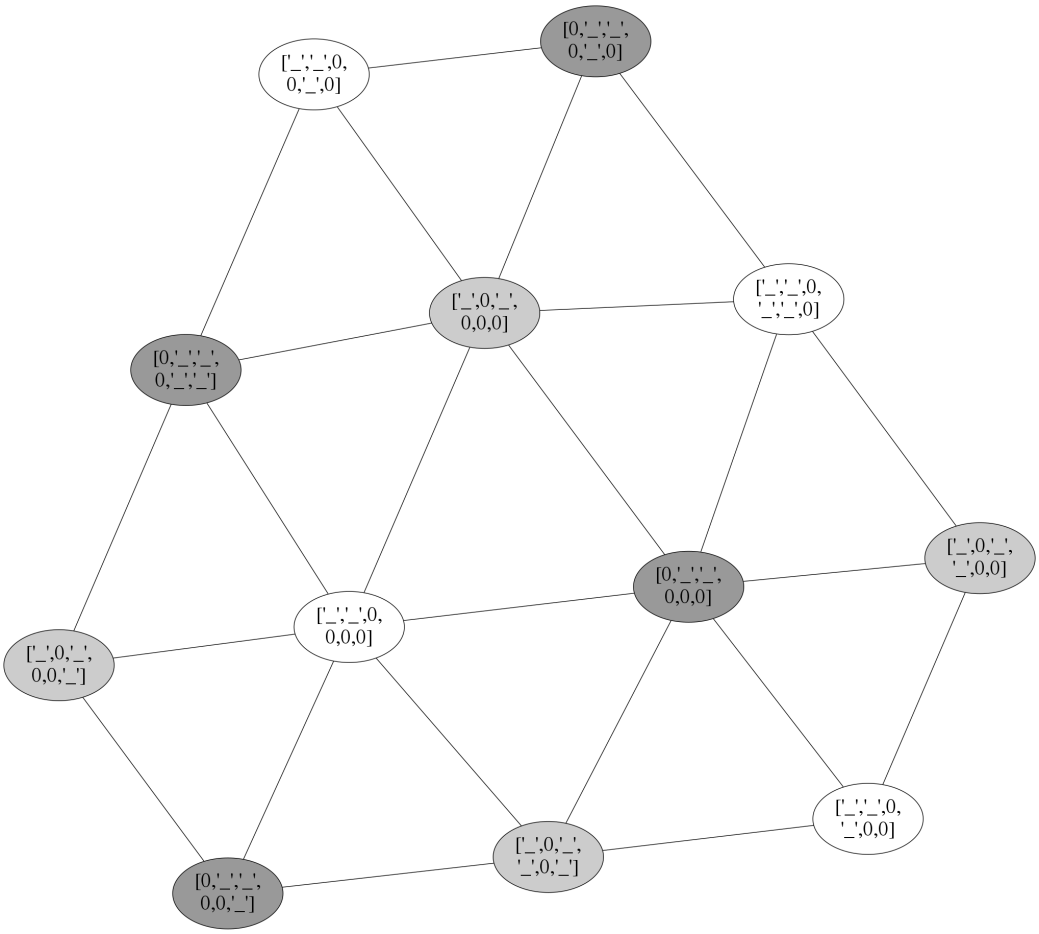

Figura 3.18: Complejo simplicial para una iteración de nuestros protocolos de estudio de tres procesos con entrada 0.

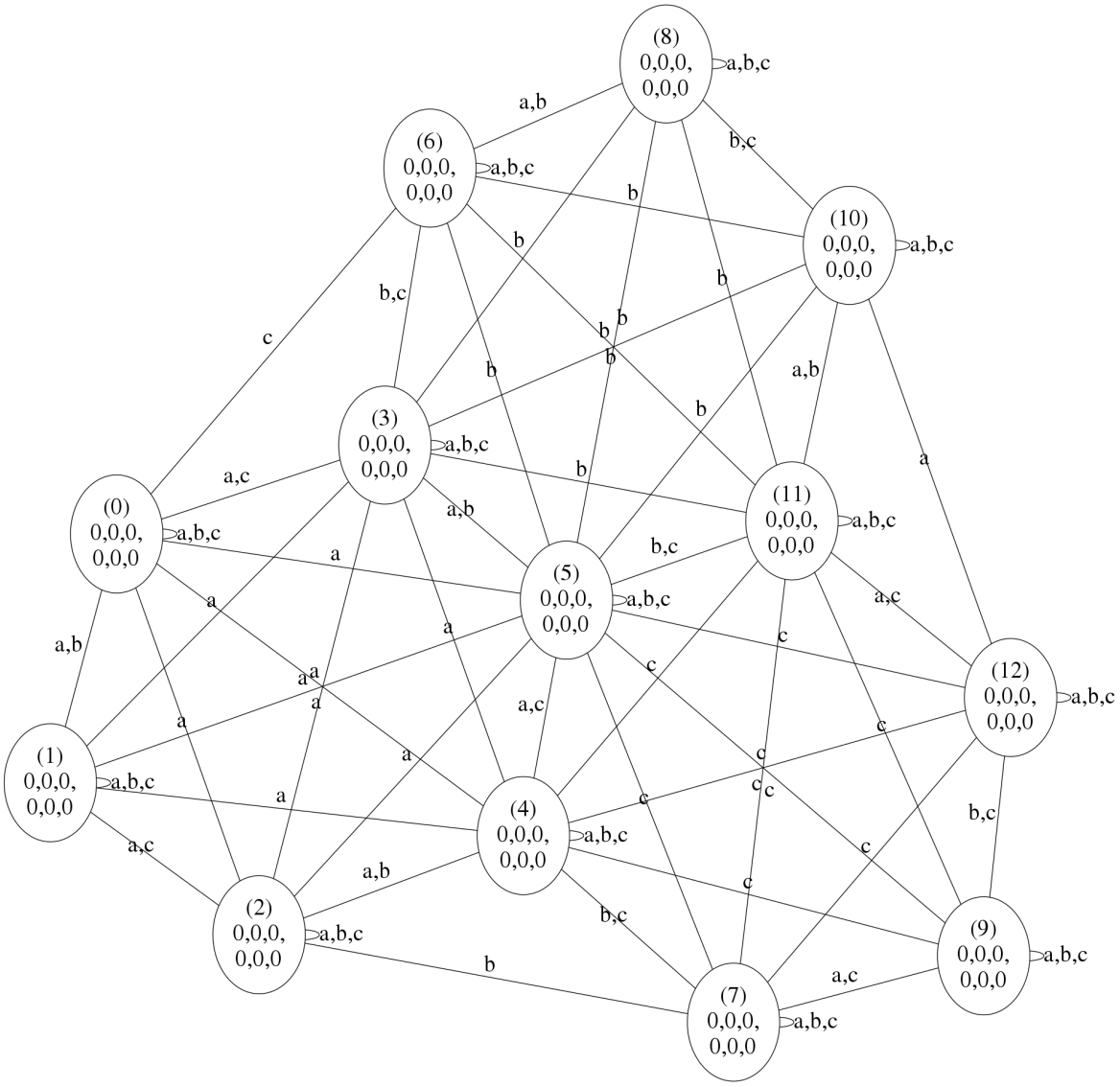

Figura 3.19: Modelo de Kripke correspondiente a las facetas del complejo simplicial de la figura 3.18.

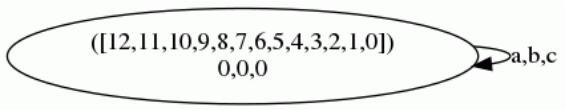

Figura 3.20: Modelo de Kripke en la figura 3.19 minimizado bajo bisimulación.

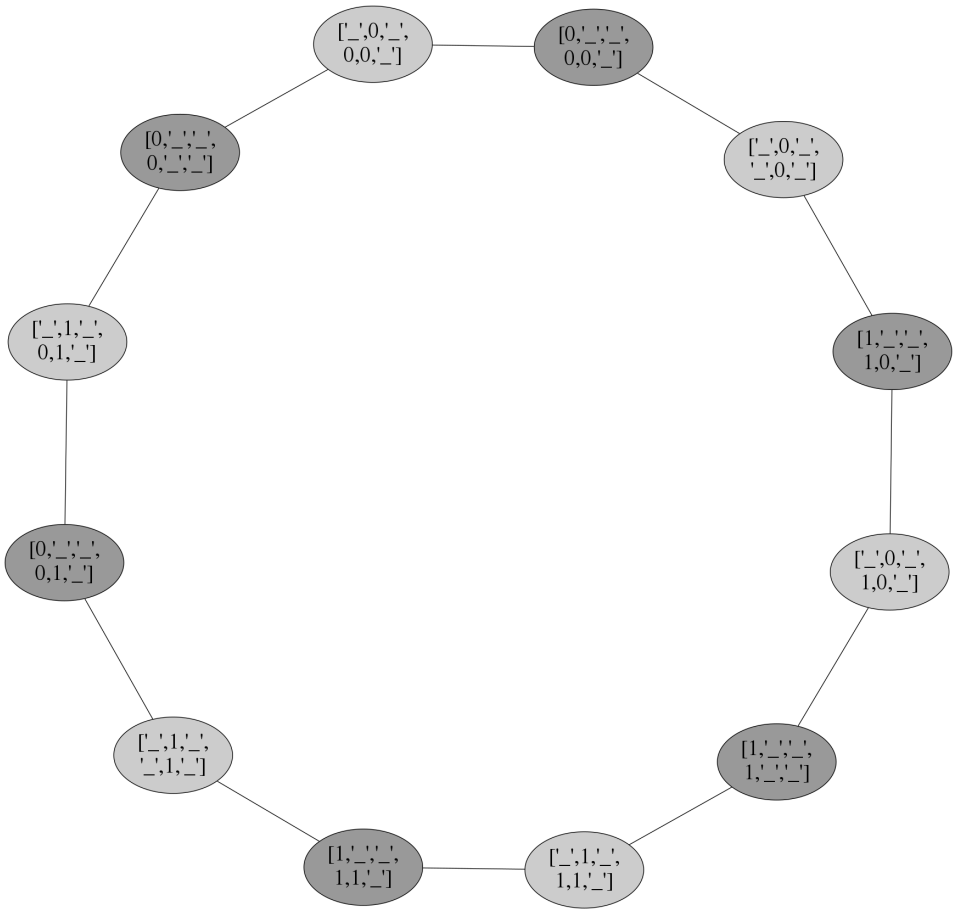

Figura 3.21: Proyección de la subdivisión cromática para tres procesos sobre los agentes a (gris oscuro) y b (gris claro).

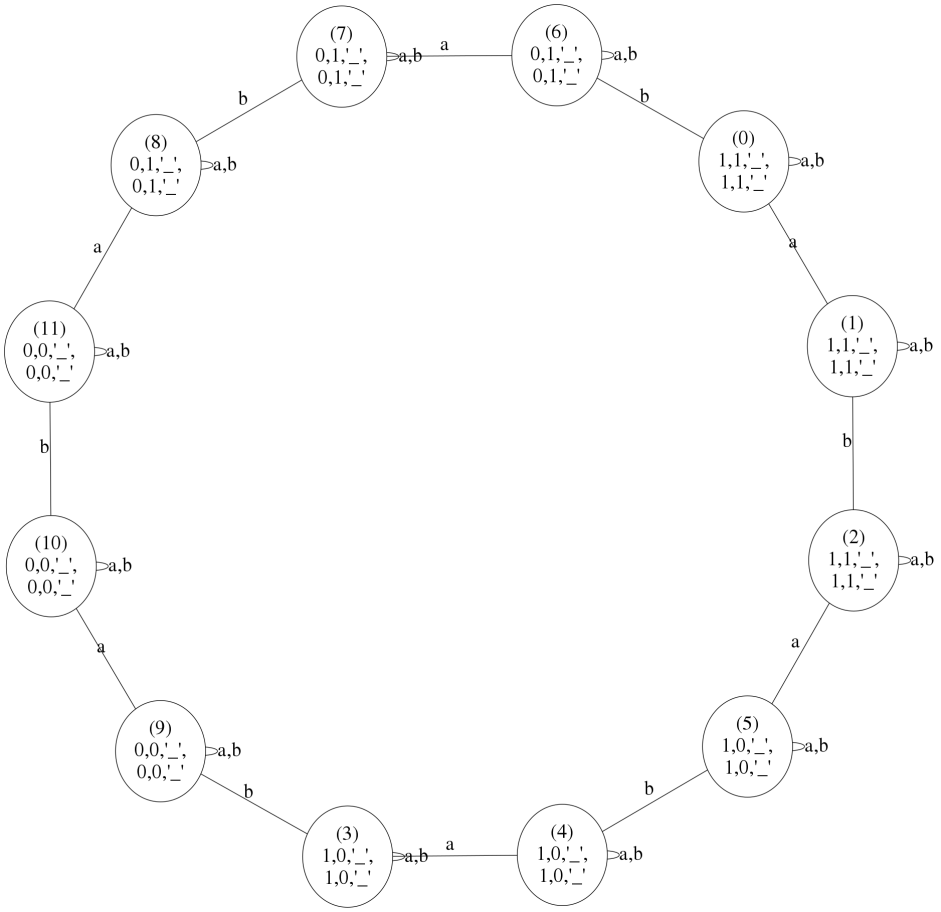

Figura 3.22: Modelo de Kripke correspondiente al complejo simplicial de la figura 3.21.

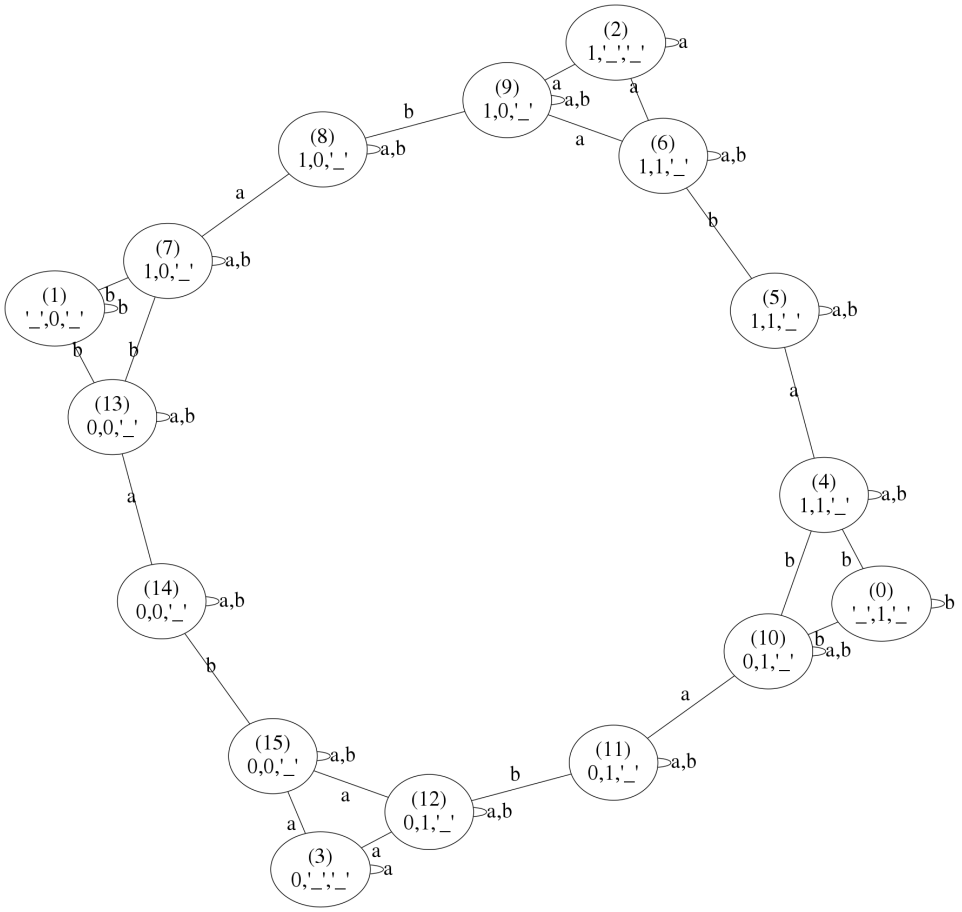

Figura 3.23: Modelos de Kripke correspondientes al complejo simplicial en la figura 3.21, para el escenario donde se supone que un proceso puede fallar del lado izquierdo y ambos del lado derecho.

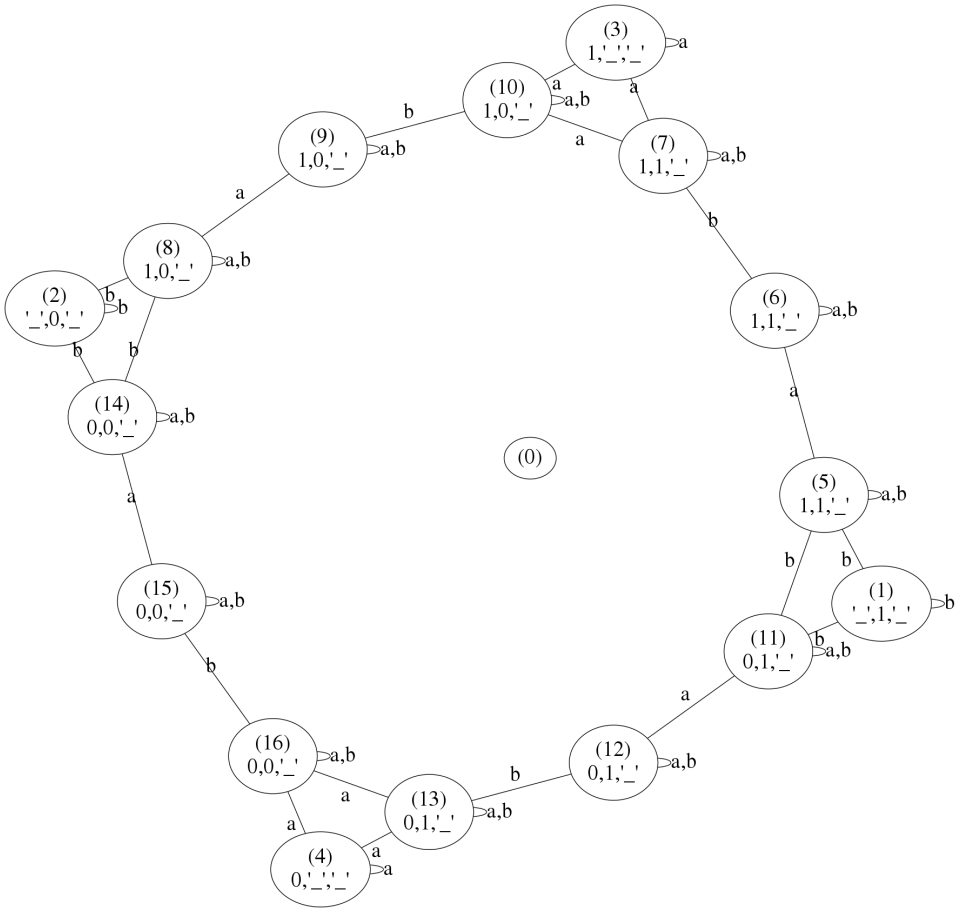

Figura 3.24: Modelos de Kripke correspondientes al complejo simplicial en la figura 3.21, para el escenario donde se supone que un proceso puede fallar del lado izquierdo y ambos del lado derecho.

Capítulo 4

Mapeos portadores y modelos de acciones extendidos

En este capítulo, trataremos complejos simpliciales puros, con las características mencionadas en 3.1 y los modelos epistémicos que pueden ser calculados por el algoritmo 8 a partir de las facetas de los complejos simpliciales mencionados, ya que analizamos la evolución del conocimiento representado en modelos de Kripke desde un enfoque de modelos de acciones. Presentamos una extensión de modelos de acciones que proveen la capacidad de agregar nuevas proposiciones a los modelos modelos de Kripke a lo largo de su evolución. Después discutimos una relación que percibimos entre los mapeos portadores y los modelos de acciones extendidos y en seguida mostramos la forma en que podemos construir un modelo de acciones extendido dada una subdivisión cromática. Después discutimos una diferencia importante que entre los mapeos portadores y los modelos de acciones extendidos: el hecho de que los mapeos portadores se pueden definir en cualquier dimensión del complejo simplicial mientras que los modelos de acciones extendidos se definen solamente sobre el conjunto total de agentes representados. Por último, presentamos una proyección sobre agentes de un modelo de Kripke, que resulta en la estructura de las definiciones del mapeo portador en dimensiones inferiores.

4.1. Modelos de acciones extendidos

Es importante tener un medio para modelar el cambio en los modelos con los que trabajamos. El enfoque topológico del cómputo distribuido utiliza mapeos portadores para modelar la evolución de los protocolos. Por otro lado, en el enfoque de la lógica, tenemos la lógica epistémica dinámica, cuyo propósito es modelar la evolución de conocimiento a través de transformaciones en los modelos de Kripke.

Para modelar cambio en lógica epistémica, partimos de los modelos de acciones, que operados con un modelo de Kripke mediante el producto modal restringido da lugar a un nuevo modelo de Kripke. Agregar proposiciones al conjunto de proposiciones con el que se trabaja es necesario para modelar la evolución de los modelos de nuestro interés ya que a lo largo de la ejecución de un protocolo los procesos que participan en el mismo acumulan información.

Tomando como punto de partida las definiciones de modelo de acciones y producto modal restringido de la subsección 2.1.6, encontramos que los modelos de acciones pueden modificar la estructura de un modelo de Kripke pero no pueden agregar proposiciones a la lógica, lo que desde nuestro punto de vista es necesario. Para mostrar una relación entre la lógica epistémica dinámica y los modelos de cómputo distribuido representados por complejos simpliciales proponemos extender a los modelos de acciones de la siguiente manera:

Def. Sea $\mathcal{L}$ el lenguaje de lógica epistémica sobre un conjunto de proposiciones P y un conjunto de agentes A y sea P' un conjunto de proposiciones disjunto con respecto a P. Un *modelo de acciones extendido* es una estructura $\mathbf{M} = \langle S, \sim, pre, L\rangle$ tal que

- S es el dominio de acciones
- $\sim \ = \bigcup \sim_a,\ a \in A$ con $\sim_a$ una relación de equivalencia en S
- $pre : S \longrightarrow \mathcal{L}$, una función de precondiciones
- $L : W \longrightarrow 2^{P'}$, una función de etiquetamiento

Def. Sean $\mathcal{M} = \langle W, R, L\rangle$ *y* $\mathcal{M}' = \langle W', R', L'\rangle$ modelos de Kripke y $\mathbf{M} = \langle S, \sim, pre, \mathbf{L}\rangle$ un modelo de acciones extendido. El *producto modal restringido extendido* $\mathcal{M}' = \mathcal{M} \bigotimes \mathbf{M}$ se define de la siguiente forma:

- $W' = \{(w, s) \mid w \in W, s \in S,\ y\ \mathcal{M}, w \vDash pre(s)\}$
- $((w_1, s_1), (w_2, s_2)) \in R' \Leftrightarrow (w_1, w_2) \in R\ y\ (s_1, s_2) \in\ \sim$
- $p \in L'((w, s)) \Leftrightarrow p \in L(w)\ o\ p \in \mathbf{L}(s)$

Se cambia la definición de los modelos de acciones en los modelos de acciones extendidos agregando en el cuarto punto una función de etiquetamiento que justamente consta de las proposiciones que se pretenden agregar en el etiquetamiento de los mundos del modelo $\mathcal{M}'$ cuando una acción en particular se aplica. Para lograr lo anterior, se cambia la definición de producto modal restringido en el producto modal restringido extendido en su tercer punto, construyendo el etiquetamiento del modelo de Kripke resultante, preservando el etiquetamiento que se tenía en el mundo del modelo operado pero también agregando el etiquetamiento del modelo de acciones extendido. Es importante notar que $\mathcal{M}'$ es un modelo que se puede analizar con el lenguaje de lógica epistémica proposicional sobre el conjunto de agentes A, pero el conjunto de proposiciones sobre el que está definido $\mathcal{M}'$ es $P \cup P'$.

4.2. Mapeos portadores y modelos de acciones extendidos

Como ya se mencionó, el enfoque topológico del cómputo distribuido usa los mapeos portadores para modelar la evolución de los complejos simpliciales a lo largo de la ejecución de un protocolo. Recordemos la definición de mapeos portadores:

Dados $\mathcal{A}$, $\mathcal{B}$ complejos simpliciales, un mapeo portador $\Phi : \mathcal{A} \to 2^{\mathcal{B}}$ manda simplejos $\sigma \in \mathcal{A}$ a un subcomplejo $\Phi(\sigma)$ de $\mathcal{B}$ tal que $\forall \sigma, \tau \in A$, si $\sigma \subseteq \tau$ entonces $\Phi(\sigma) \subseteq \Phi(\tau)$.

Desde el punto de vista de lógica, en particular de lógica epistémica, la evolución de los modelos de Kripke usa las lógicas epistémicas dinámicas. Por ejemplo: los anuncios públicos, las acciones epistémicas o los modelos de acciones presentados en [3]. Particularmente, como ya se mencionó en 4.1, este trabajo toma como punto de partida los modelos de acciones para definir los modelos de acciones extendidos, que es otra forma de lógica epistémica dinámica, que permite aumentar el conjunto de proposiciones con el que se trabaja.

En esta sección presentamos una relación entre los mapeos portadores y los modelos de acciones extendidos. En este capítulo trabajamos con modelos de Kripke correspondientes a las facetas de un complejo simplicial cromático y puro. Más precisamente, señalamos que de forma similar en la que un mapeo portador puede relacionar, desde el punto de vista de la estructura de las facetas del complejo simplicial, dos complejos simpliciales $\mathcal{A}$ y $\mathcal{B}$, los modelos de acciones extendidos pueden ser operados por medio del producto modal restringido extendido con el modelo de Kripke correspondiente a las facetas del complejo simplicial $\mathcal{A}$, $\mathcal{M}_{\mathcal{A}}$, para obtener el modelo de Kripke correspondiente a las facetas de $\mathcal{B}$, $\mathcal{M}_{\mathcal{B}}$.

4.3. Subdivisiones cromáticas y modelos de acciones extendidos

Las subdivisiones cromáticas de los complejos simpliciales son particularmente importantes para el modelo *Iterated Immediate Snapshots*. En este trabajo estudiamos protocolos particulares en este modelo en los que la primera iteración corresponde a la primera en los protocolos de información completa. Intuitivamente, una subdivisión cromática induce un complejo simplicial cromático $\mathcal{B}$ a partir de otro complejo simplicial $\mathcal{A}$ sustituyendo simplejos de $\mathcal{A}$ por "divisiones" del mismo. Por ejemplo, tomemos un complejo simplicial $\mathcal{F}$ con una sola faceta de dimensión 2 y su subdivisión cromática $\mathcal{S}$ después de una iteración de nuestros protocolos de estudio mostrados en la figura 4.1.

A continuación mostraremos una forma en la que podemos identificar los mundos de un modelo de Kripke construido a partir de las facetas de un complejo simplicial cromático puro que represente una ejecución acotada de los protocolos que estudiamos y en el que todos los agentes pueden tener más de una posible entrada. Enseguida mostraremos la forma en la que una subdivisión cromática se representaría como modelo de acciones extendido para el caso específico del mundo correspondiente a una faceta del complejo simplicial. Finalmente, presentamos la forma en que podemos construir un modelo de acciones extendido correspondiente a todas las facetas del complejo simplicial.

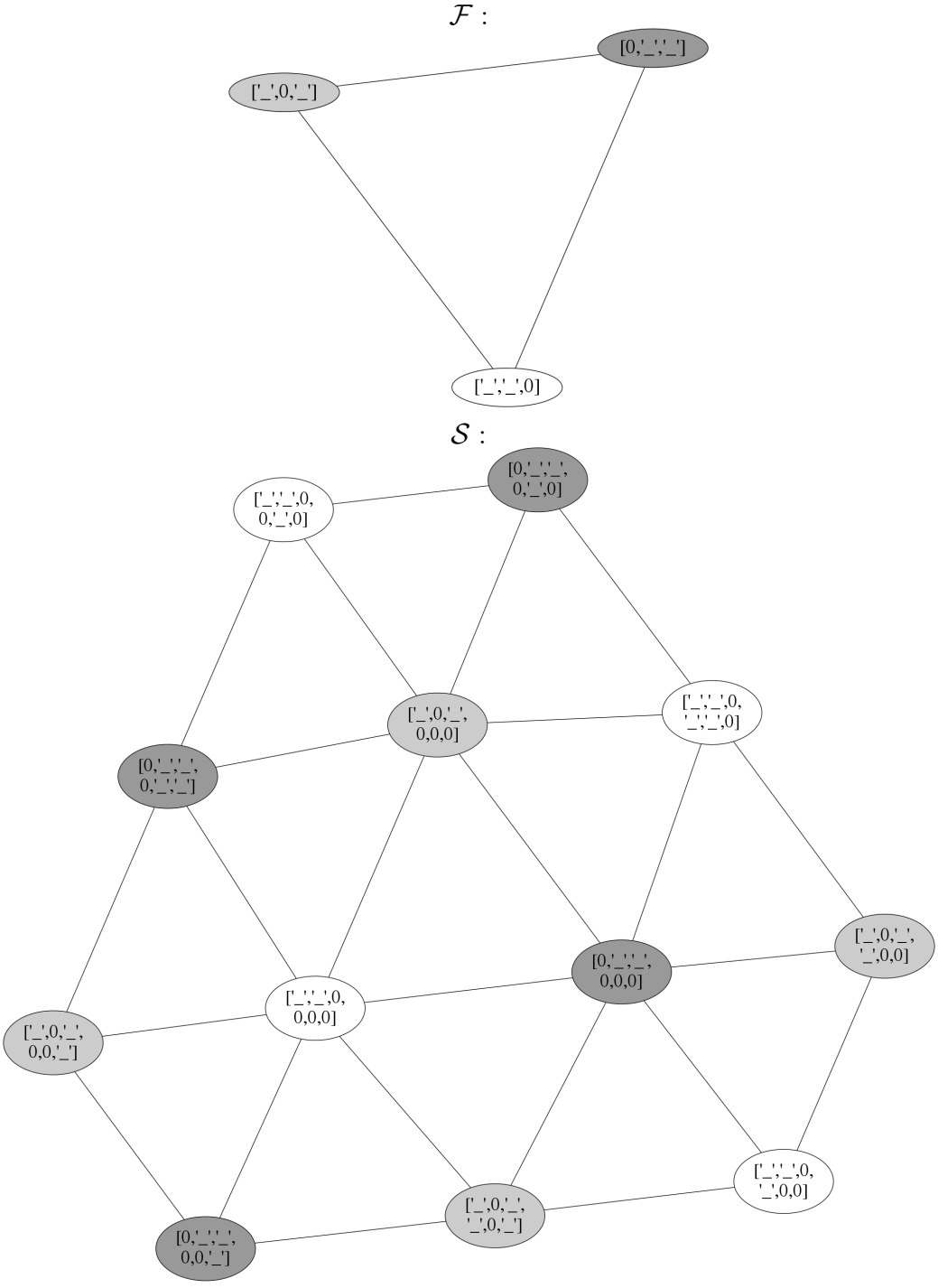

Figura 4.1: Complejo simplicial $\mathcal{F}$ de dimensión 2 con una sola faceta y su subdivisión cromática $\mathcal{S}$ después de la primera iteración de nuestros protocolos de estudio para tres procesos con entrada 0.

4.3.1. Identificación de mundos

En este punto, lo que pretendemos es mostrar una manera en la que podamos seleccionar cada mundo, correspondiente a una faceta del complejo simplicial para crear la estructura adecuada por medio de un modelo de acciones, faceta por faceta. Lo primero es mostrar que podemos identificar de alguna manera a los mundos para que al operarlos con el modelo de acciones extendido, podamos obtener el submodelo correspondiente al mundo "subdividido"[1]. Enunciamos el lema 4.3.1 que nos da una propiedad suficiente y necesaria para que podamos identificar a un mundo en un modelo de Kripke mediante una fórmula de lógica epistémica cierta a partir del mismo. Consideramos que dicho lema es elemental, sin embargo no tenemos conocimiento de que se haya presentado en algún otro lugar.

Lema 4.3.1. *$\mathcal{M} = \langle W, R, L\rangle$ es un modelo de Kripke mínimo con respecto a bisimulación si y solo si cada $w \in W$ puede identificarse con una fórmula de lógica epistémica ϕ tal que, $\mathcal{M}, w \models \phi$.*

Demostración.

$\Rightarrow$)

Si $W = \{w\}$, w puede identificarse con la formula $\top$. En caso de que $|W| \geq 1$. Supongamos por contradicción que existen dos mundos, $w_1, w_2 \in W$, que no pueden identificarse entre sí por ninguna fórmula de lógica epistémica ϕ. Entonces, $\mathcal{M}, w_1 \models \phi$ si y solo si $\mathcal{M}, w_2 \models \phi$. De lo anterior se sigue que w_1, w_2 están en la misma clase de equivalencia bajo bisimulación, lo cual contradice que $\mathcal{M}$ sea mínimo bajo bisimulación.

Por lo tanto, si $\mathcal{M} = \langle W, R, L\rangle$ es mínimo con respecto a bisimulación, cada $w \in W$ se puede identificar por una fórmula de lógica epistémica cierta en w.

$\Leftarrow$)

Por contradicción supongamos que existen dos mundos en $w_1, w_2 \in \mathcal{W}$ tales que no pueden identificarse por ninguna fórmula $\phi \in \mathcal{L}$ entre ellos. Esto significaría que existe una bisimulación en la que tanto w_1 como w_2 pertenecen a la misma clase de equivalencia. De lo anterior se sigue que $\mathcal{M}$ no es mínimo con respecto a bisimulación, lo cual es una contradicción ya que por hipótesis $\mathcal{M}$ es mínimo con respecto a bisimulación.

[1]Nos referimos a la subdivisión cromática en complejos simpliciales representada como modelo de Kripke.

□

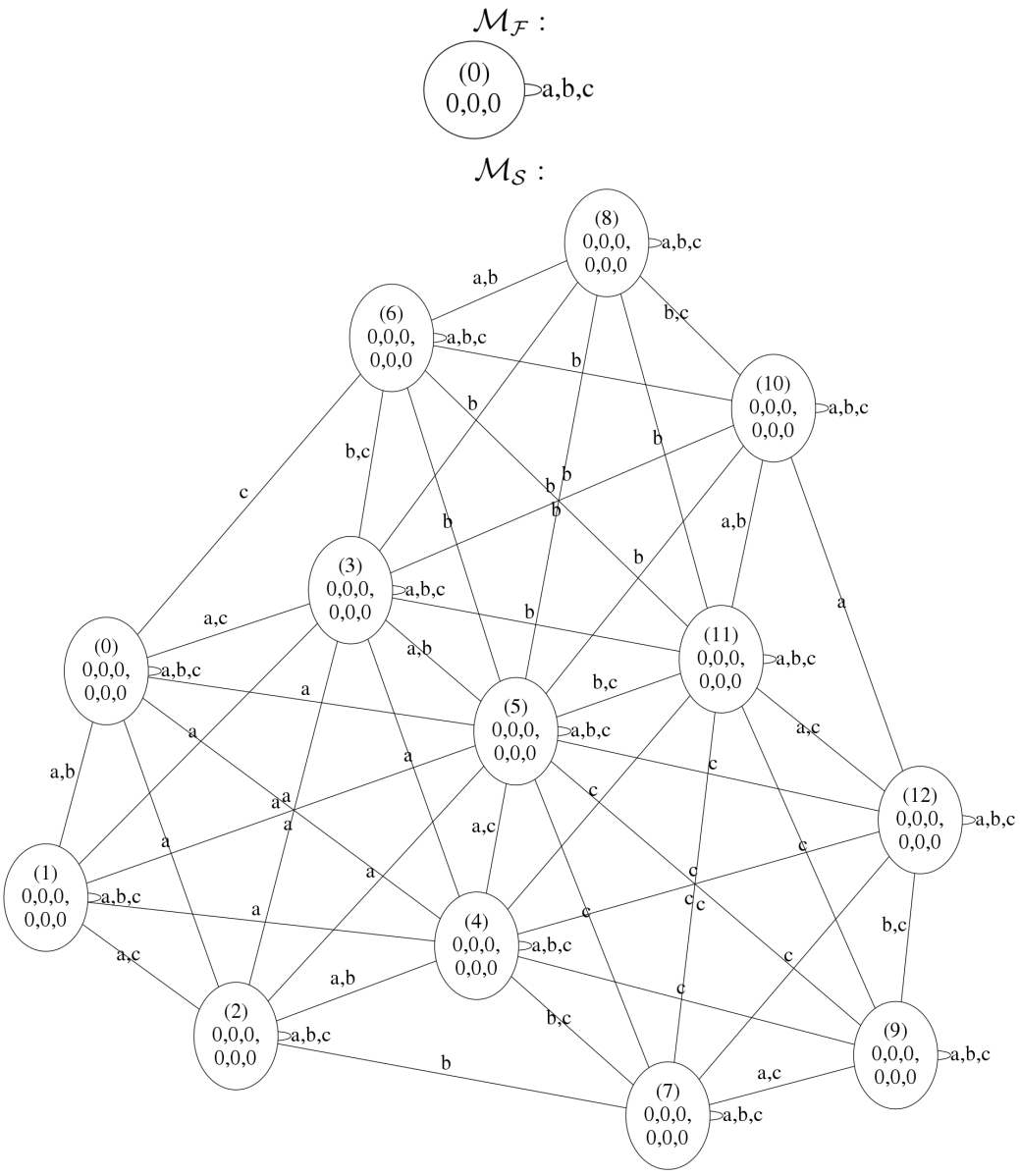

Figura 4.2: Modelos de Kripke $\mathcal{M}_{\mathcal{F}}$ y $\mathcal{M}_{\mathcal{S}}$ correspondientes a las facetas de los complejos simpliciales $\mathcal{F}$ y $\mathcal{S}$ de la figura 4.1.

Por ahora es suficiente con saber la existencia de las fórmulas con las que

podemos identificar los mundos. Agreguemos a nuestros complejos simpliciales adicionalmente a las mencionadas en 3.1 la restricción de que los procesos tienen al menos dos posibles entradas. La característica mencionada permite identificar fácilmente a los mundos que se generan en los modelos de Kripke correspondientes a facetas de los complejos simpliciales por medio de lo que ven con un valor distinto a $\perp$ como lo que conocen. y lo que ven como $\perp$ como lo que no conocen[2].

Consideremos el modelo de Kripke, $\mathcal{M}_{\mathcal{S}} = \langle W, R, L \rangle$ en la figura 4.2, que representa la subdivisión cromática para tres procesos después de la primera iteración para nuestros protocolos de estudio cuyos procesos tienen entrada 0. Notemos que $\mathcal{M}_{\mathcal{S}_2} \models C_{\{a,b,c\}}(p_{0,a,0}, p_{0,b,0}, p_{0,c,0})$, por lo que a partir de cada mundo, $w \in W$, $\mathcal{M}_{\mathcal{S}_2}, w \models K_a(p_{0,a,0}, p_{0,b,0}, p_{0,c,0}) \wedge K_b(p_{0,a,0}, p_{0,b,0}, p_{0,c,0}) \wedge K_c(p_{0,a,0}, p_{0,b,0}, p_{0,c,0})$. Esto sucede pues todos los procesos correspondientes a los agentes del modelo solamente pueden escribir en su memoria correspondiente el valor 0.

Consideremos ahora la figura 4.3, en la que se muestra el complejo simplicial correspondiente al caso en el que el proceso c puede escribir los valores $\{0, 1\}$ en su memoria y la figura 4.4, con el modelo de Kripke correspondiente a las facetas del complejo mencionado. Notemos que en los mundos $\{7, 9, 12, 20, 22, 25\}$, mostrados en el centro del modelo, los agentes $\{a, b\}$ ya no saben qué valor escribió el agente c. De manera similar, a medida que cada uno de los procesos tiene la posibilidad de escribir más de un valor en la memoria que le corresponde, el conocimiento de los agentes en el modelo se acerca a lo que el proceso correspondiente lee.

[2]Suponemos que $\perp$ es ajeno a los valores de entrada para los procesos.

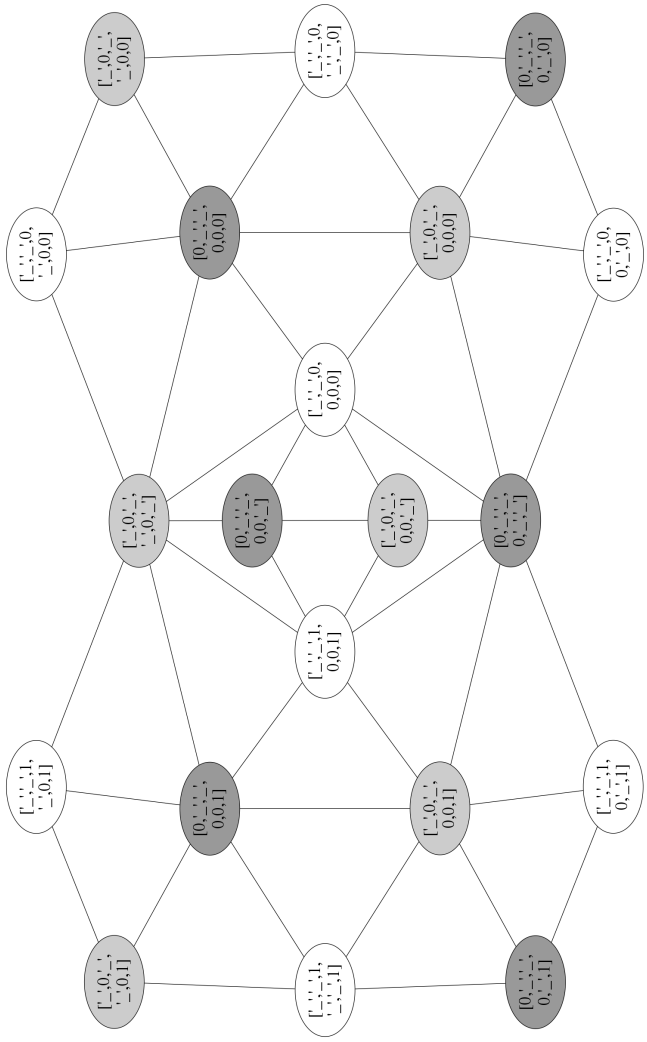

Figura 4.3: Subdivisión cromática después de la primera iteración del modelo *Iterated Immediate Snapshots* para tres procesos: a y b con entrada 0 y c con entrada en $\{0, 1\}$.

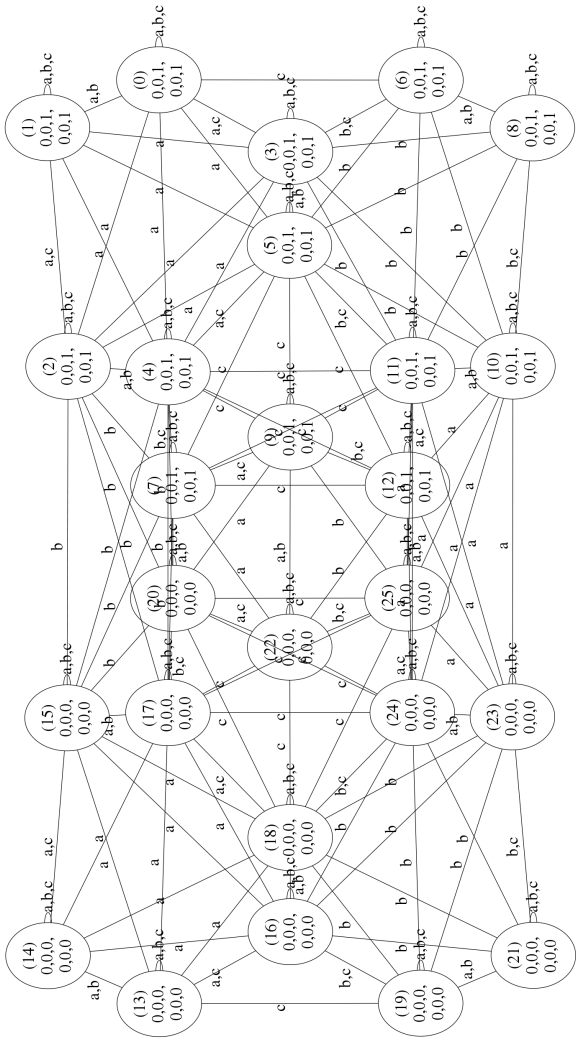

Figura 4.4: Modelo de Kripke correspondiente a las facetas del complejo simplicial de la figura 4.3.

Regresando a nuestros protocolos de estudio; Si suponemos que los procesos tienen entradas al menos binarias, podremos identificar a cada mundo con una fórmula de lógica epistémica que sea una conjunción del etiquetamiento del mundo y si cada uno de los agentes conoce o no el valor de la entrada de cada uno de los procesos. Primero construimos la fórmula a partir de una faceta del complejo simplicial y después con la información ya codificada en un modelo de Kripke.

Sea

$$\begin{aligned}\sigma = \{ & \\ & v(a_1, [v_{0,a_1,a_1}, \ldots, v_{0,a_1,a_n}, \ldots, v_{k,a_1,a_1}, \ldots, v_{k,a_1,a_n}]), \\ & v(a_2, [v_{0,a_2,a_1}, \ldots, v_{0,a_2,a_n}, \ldots, v_{k,a_2,a_1}, \ldots, v_{k,a_2,a_n}]), \\ & \qquad\qquad\qquad\qquad \vdots \\ & v(a_n, [v_{0,a_n,a_1}, \ldots, v_{0,a_n,a_n}, \ldots, v_{k,a_n,a_1}, \ldots, v_{k,a_n,a_n}]) \\ \} &\end{aligned}$$

una faceta de un complejo simplicial que representa k iteraciones de nuestros protocolos de estudio para n procesos. En esta representación las vistas de cada vértice tienen los valores que han leído hasta la k-ésima iteración. Las variables v_{0,a_i,a_j} son las entradas de los procesos por lo que solamente las v_{0,a_i,a_i} tienen valor distinto a $\perp$. Las variables $v_{it,a_i,a_j}, it > 0$ corresponden a los valores que el proceso a_i leyó del proceso a_j en la iteración it.

El mundo w_σ, que le corresponde a σ en el modelo de Kripke construido a partir de las facetas de $\mathcal{C}$, puede ser identificado por la fórmula de lógica epistémica construida por el algoritmo 10. La fórmula que se construye consta de la conjunción de las proposiciones que representan las entradas de los procesos, es decir $p_{0,a_1,a_1} \wedge p_{0,a_2,a_2} \cdots \wedge p_{0,a_n,a_n}$, y adicionalmente para todo valor en la vista del vértice, si el valor en la vista es diferente de $\perp$ se agrega una conjunción a la fórmula ϕ, de la forma $K_{a_i}(p_{it,a_j,v_{0,a_j,a_j}})$. Notemos que si $v_{it,a_i,a_j} \neq \perp$ entonces a_i leyó la entrada de a_j en la iteración it, pero por las características de nuestros protocolos de estudio $v_{it,a_i,a_j} = v_{0,a_j,a_j}$ ya que los procesos siempre escriben su entrada. Intuitivamente la fórmula puede identificar el mundo ya que cuando un proceso hace el snapshot puede distinguir entre los mundos donde un procesos escribe sus distintas entradas si y solo si el snapshot contempla la escritura del proceso mencionado y además, a que la fórmula incluye la conjunción de las proposiciones que etiquetan al mundo w_σ.

Argumentemos que el algoritmo 10 es correcto. En la línea 2 se inicializa ϕ a $\top$ que es el neutro en la conjunción. En el ciclo de las lineas 3-5 se agrega a ϕ las conjunciones con las proposiciones correspondientes a las entradas de cada uno de los procesos. En el ciclo de las lineas 8-14 se examina si el valor del proceso a_j en una iteración it desde el punto de vista de un proceso a_i es distinto de $\perp$ lo que implica que el proceso a_i leyó la entrada del proceso a_j en la iteración it. Si dicha condición se cumple se agrega a ϕ una conjunción con $K_{a_i}(p_{it,a_j,v_{it,a_i,a_j}})$, en caso contrario con $\neg K_{a_i}(p_{it,a_j,v_{0,a_j,a_j}})$. Recordemos

Algoritmo 10 Algoritmo para calcular una fórmula que identifica a w_σ.

```
1: function PHI(σ, k, n)
2:     φ ← ⊤
3:     for all i ∈ {1, ..., n} do                          ▷ Entradas
4:         φ ← φ ∧ p_{0,a_i,v_{0,a_i,a_i}}
5:     end for
6:     for all it ∈ {1, ..., k} do                         ▷ Iteraciones
7:         for all i ∈ {1, ..., n} do                      ▷ Procesos
8:             for all j ∈ {1, ..., n} do                  ▷ Vértice
9:                 if v_{it,a_i,a_j} ≠ ⊥ then
10:                    φ ← φ ∧ K_{a_i}(p_{it,a_j,v_{it,a_i,a_j}})
11:                else
12:                    φ ← φ ∧ ¬K_{a_i}(p_{it,a_j,v_{0,a_j,a_j}})
13:                end if
14:            end for
15:        end for
16:    end for
17:    return φ
18: end function
```

que si $v_{it,a_i,a_j} \neq \bot$ entonces $v_{it,a_i,a_j} = v_{0,a_j,a_j}$. Esto se hace para cada proceso a_i en el ciclo de las lineas 7-15 y todo lo anterior para cada iteración en el ciclo de las lineas 6-16. De lo anterior podemos decir que *phi* es la fórmula que se describió para identificar al mundo w_σ salvo por el $\top$ al principio, pero como $\top$ es el neutro para la conjunción ϕ también identifica al mundo w_σ.

Similarmente, si ya se ha construido $\mathcal{MC} = \langle W_\mathcal{C}, R_\mathcal{C}, L_\mathcal{C} \rangle$, el modelo de Kripke sobre los agentes en el conjunto $A = \{a_1, a_2, \ldots, a_n\}$ y correspondiente al complejo simplicial $\mathcal{C}$, podemos construir una fórmula con con la misma estructura que en el algoritmo 10 usando el algoritmo 11.

Argumentemos que el algoritmo 11 es correcto. Tenemos que ϕ se inicializa a $\top$ en la línea 2. Nuevamente notemos que $\top$ es el neutro de la conjunción. En el ciclo de las lineas 3-5 se agregan a ϕ las conjunciones de las entradas de cada proceso. En el ciclo de las lineas 8-15 se evalúa si en un proceso a_i conoce el valor que escribió el proceso a_j en una iteración it. Lo anterior se ejecuta para cada proceso a_i en el ciclo de las lineas 8-15. Finalmente todo lo anterior para todo it en el ciclo de las lineas 6-17. Por lo anterior la formula que se construye tiene la estructura que buscamos.

Algoritmo 11 Algoritmo para calcular una fórmula que identifica a $w \in W$ del modelo de Kripke $\mathcal{M}$.

```
1: function PHI(w ∈ W_C, k, n, M_C = ⟨W_C, R_C, L_C⟩)
2:     φ ← ⊤
3:     for all p_{0,a,v} ∈ L(w) do
4:         φ ← φ ∧ p_{0,a,v}
5:     end for
6:     for all it ∈ {1, ..., k} do                          ▷ Iteraciones
7:         for all a_i, i ∈ {1, ..., n} do                  ▷ Procesos
8:             for all a_j, j ∈ {1, ..., n} do
9:                 Sea v tal que p_{0,a_j,v} ∈ L(w)
10:                if M_C, w ⊨ K_a(p_{k,a_j,v}) then
11:                    φ ← φ ∧ K_{a_i}(p_{k,a_j,v})
12:                else
13:                    φ ← φ ∧ ¬K_{a_i}(p_{k,a_j,v})
14:                end if
15:            end for
16:        end for
17:    end for
18:    return φ
19: end function
```

Tanto el algoritmo 10 como el 11 toman como entrada

- k, el número de iteraciones del protocolo representadas
- n, el número de procesos que participan en el protocolo

Para el algoritmo 10 se tiene como entrada adicional σ, el simplejo que corresponde al mundo w_σ en el modelo de Kripke construido a partir del conjunto de facetas de $\mathcal{C}$ del que σ es un elemento.

Para el algoritmo 11 se toman como entradas adicionales el modelo de Kripke $\mathcal{M}_\mathcal{C} = \langle W, R, L\rangle$ correspondiente al complejo $\mathcal{C}$ y un mundo $w \in W$ que será al que se identificará por la fórmula ϕ.

4.3.2. Subdivisión cromática de una faceta y su modelo de acciones extendido

En esta sección comenzaremos por ejemplificar los modelos de acciones correspondientes a las subdivisiones cromáticas para dos y tres procesos. Luego explicaremos por qué los modelos de acciones extendidos expuestos en los ejemplos tienen la estructura propuesta y por último daremos una generalización para la construcción de los modelos de acciones extendidos dada la subdivisión cromática de una faceta de un complejo simplicial.

Primero tomemos como ejemplo la subdivisión cromática para dos procesos en el modelo *Iterated Immediate Snapshots*. La figura 4.5 muestra el complejo simplicial $\mathcal{F}_1$, correspondiente al caso en el que ambos procesos tienen como entrada 0 antes de comenzar con la ejecución del protocolo y el complejo simplicial $\mathcal{S}_1$ correspondiente a la subdivisión de $\mathcal{F}_1$ después de una iteración del protocolo. La figura 4.6 muestra el modelo de Kripke $\mathcal{M}_{\mathcal{F}_1} = \langle W_{\mathcal{F}_1}, R_{\mathcal{F}_1}, L_{\mathcal{F}_1} \rangle$ y $\mathcal{M}_{\mathcal{S}_1} = \langle W_{\mathcal{S}_1}, R_{\mathcal{S}_1}, L_{\mathcal{S}_1} \rangle$, correspondientes a las facetas de $\mathcal{F}_1$ y $\mathcal{S}_1$ respectivamente.

Un modelo de acciones extendido $\mathbf{M}_{\mathcal{S}_1} = \langle S_1, \sim_1, Pre_1, L_1 \rangle$ tal que $\mathcal{M}_{\mathcal{F}_1} \bigotimes \mathbf{M}_{\mathcal{S}_1} = \mathcal{S}_1$ es mostrado en la figura 4.7. La función de precondición Pre_1 está definida como sigue:

$Pre_1(s) = true \ \forall s \in S_1$.

Observemos que salvo por $L_{\mathcal{S}_1}$ de $\mathcal{M}_{\mathcal{S}_1}$, y Pre_1 y L_1 de $\mathbf{M}_{\mathcal{S}_1}$, $\mathcal{M}_{\mathcal{S}_1}$ y $\mathbf{M}_{\mathcal{S}_1}$ son dos digráficas isomorfas.

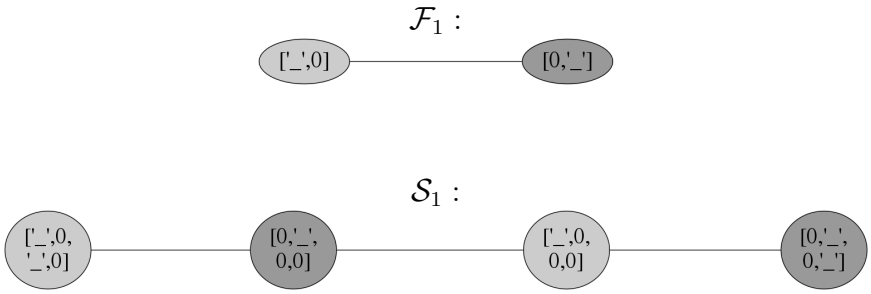

Figura 4.5: Complejo simplicial $\mathcal{F}_1$ de dimensión 1 con una sola faceta y su subdivisión cromática $\mathcal{S}_1$ después de la primera iteración de nuestros protocolos de estudio de dos procesos con entrada 0.

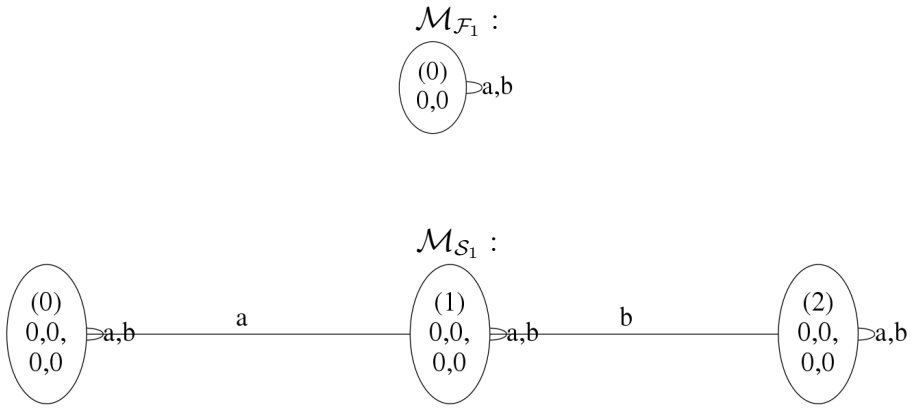

Figura 4.6: Modelos de Kripke $\mathcal{M}_{\mathcal{F}_1}$ y $\mathcal{M}_{\mathcal{S}_1}$ correspondientes a las facetas de los complejos simpliciales $\mathcal{F}_1$ y $\mathcal{S}_1$ de la figura 4.5.

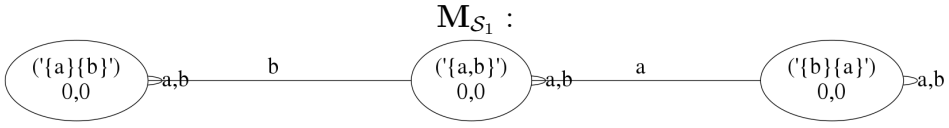

Figura 4.7: Modelo de acciones extendido $\mathbf{M}_{\mathcal{S}_1}$.

Ahora tomemos como ejemplo la subdivisión cromática para tres procesos en el modelo *Iterated Immediate Snapshots*. La figura 4.8 muestra el complejo simplicial $\mathcal{F}_2$ correspondiente al caso en el que los tres procesos tienen como entrada 0 antes de comenzar con la ejecución del protocolo y el complejo simplicial $\mathcal{S}_2$ correspondiente a la subdivisión de $\mathcal{F}_2$ después de una iteración del protocolo. La figura 4.9 muestra el modelo de Kripke $\mathcal{M}_{\mathcal{F}_2} = \langle W_{\mathcal{F}_2}, R_{\mathcal{F}_2}, L_{\mathcal{F}_2} \rangle$ y $\mathcal{M}_{\mathcal{S}_2} = \langle W_{\mathcal{S}_2}, R_{\mathcal{S}_2}, L_{\mathcal{S}_2} \rangle$, correspondientes a las facetas de $\mathcal{F}_2$ y $\mathcal{S}_2$ respectivamente.

Un modelo de acciones extendido $\mathbf{M}_{\mathcal{S}_2} = \langle S_2, \sim_2, Pre_2, L_2 \rangle$ tal que $\mathcal{M}_{\mathcal{F}_2} \bigotimes \mathbf{M}_{\mathcal{S}_2} = \mathcal{S}_2$ es mostrado en la figura 4.10. La función de precondición está definida como sigue:

$Pre_2(s) = true \ \forall s \in S_2$.

Recordemos que los modelos de acciones extendidos se operan con modelos de Kripke a través del producto modal restringido extendido. La aplicación del producto modal restringido extendido sobre $\mathcal{M}_1$ y $\mathbf{M}$ hace que $W_2 \subseteq W_1 \times S$. Observemos que los modelos de Kripke $\mathcal{M}_{\mathcal{F}_1}$ y $\mathcal{M}_{\mathcal{F}_2}$ constan de un solo mundo, de tal forma que los modelos de acciones extendidos $\mathbf{M}_{\mathcal{S}_1}$

$\mathbf{M}_{\mathcal{S}_2}$ deben tener la misma estructura en cuanto a mundos y relaciones que $\mathcal{S}_1$ y $\mathcal{S}_2$ respectivamente. Además como $\mathcal{M}_{\mathcal{F}_1}$ y $\mathcal{M}_{\mathcal{F}_2}$ tienen un solo mundo, podemos tener una precondición constante *true* en ambos modelos de acciones extendidos. Notemos además que la representación de los modelos de acciones extendidos en las figuras 4.7 y 4.10 usa como identificador de los puntos de acción extendido del modelo, las diferentes formas de concurrencia de los procesos correspondientes a los agentes en el complejo simplicial correspondiente.

4.3.2.1. Generalización

Consideremos a $\mathcal{F}_n$, una faceta de un complejo simplicial que modela k iteraciones de nuestros protocolos de estudio y a $\mathcal{S}_n$, la subdivisión cromática de $\mathcal{F}_n$. Además consideremos a $\mathcal{M}_{\mathcal{F}_n} = \langle\{\sigma_{\mathcal{F}_n}\}, R_{\mathcal{F}_n}, L_{\mathcal{F}_n}\rangle$ y a $\mathcal{M}_{\mathcal{S}_n} = \langle W_{\mathcal{S}_n}, R_{\mathcal{S}_n}, L_{\mathcal{S}_n}\rangle$, los modelos de Kripke correspondientes a los conjuntos de facetas de $\mathcal{F}_n$ y $\mathcal{S}_n$ respectivamente sobre el conjunto de agentes A_n.

Podemos generalizar la construcción de modelos de acciones extendidos $\mathbf{M}_{\mathcal{S}_n}$ tales que $\mathbf{M}_{\mathcal{S}_n} \bigotimes \mathcal{M}_{\mathcal{F}_n} = \mathcal{M}_{\mathcal{S}_n}$ construyendo a $\mathbf{M}_{\mathcal{S}_n} = \langle S, \sim, pre, L\rangle$ como sigue:

- S es un conjunto de puntos de acción tal que $\mid S \mid \;=\; \mid W_{\mathcal{S}_n} \mid$
- $f : W_{\mathcal{S}_n} \longrightarrow S$ es una función que biyecta los conjuntos $W_{\mathcal{S}_n}$ y S
- $(f(w_1), Ags, f(w_2)) \in \;\sim\; \Leftrightarrow (w_1, Ags, w_2) \in R_{\mathcal{S}_n}$
- $pre(s) = \top \; \forall s \in S$
- $L(s) = \{p_{k+1,a,v} \mid a \in A_n,\ p_{0,a,v} \in L_{\mathcal{F}_n}(\sigma_{\mathcal{F}_n})\}$

De esa forma estamos copiando la estructura de $\mathcal{M}_{\mathcal{S}_n}$ a $\mathbf{M}_{\mathcal{S}_n}$ añadiendo el etiquetamiento adecuado y una función de precondición suficiente por ser $\sigma_{\mathcal{F}_n}$ el único mundo de $\mathcal{M}_{\mathcal{F}_n}$.

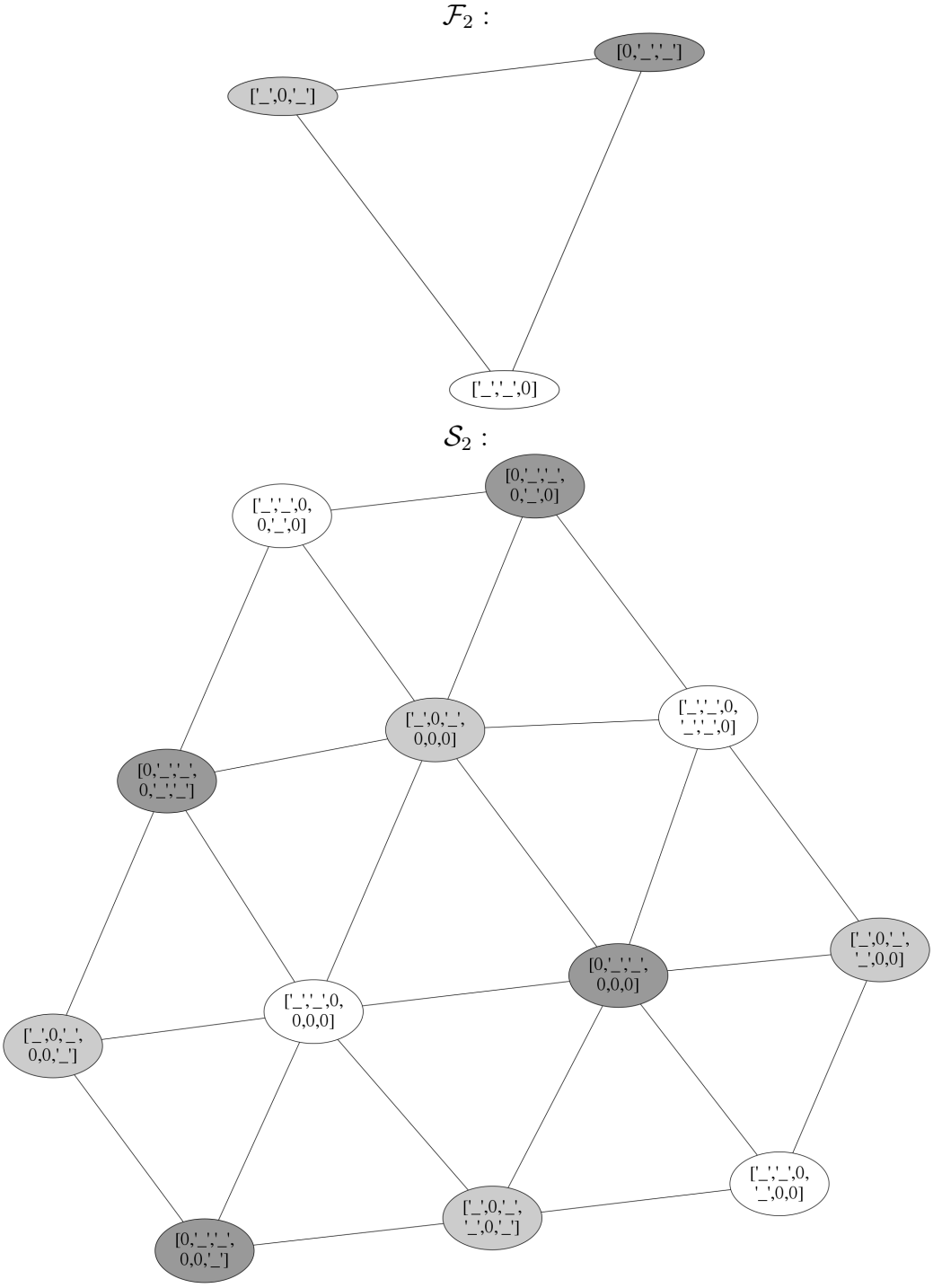

Figura 4.8: Complejo simplicial $\mathcal{F}_2$ de dimensión 2 con una sola faceta y su subdivisión cromática $\mathcal{S}_2$ después de la primera iteración del modelo *Iterated Immediate Snapshots* para tres procesos con entrada 0.

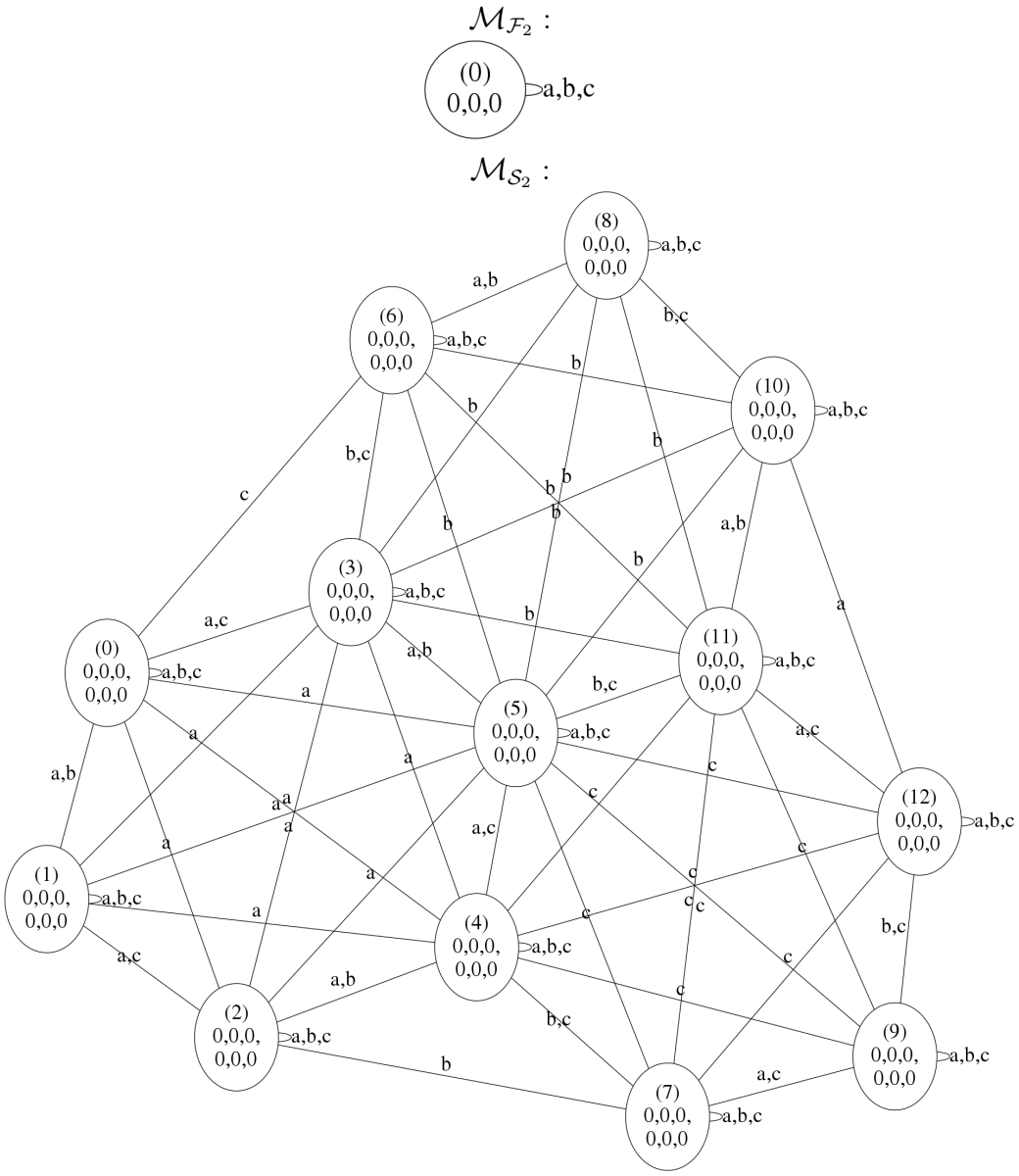

Figura 4.9: Modelos de Kripke $\mathcal{M}_{\mathcal{F}_2}$ y $\mathcal{M}_{\mathcal{S}_2}$ correspondientes a las facetas de los complejos simpliciales $\mathcal{F}_2$ y $\mathcal{S}_2$ de la figura 4.8.

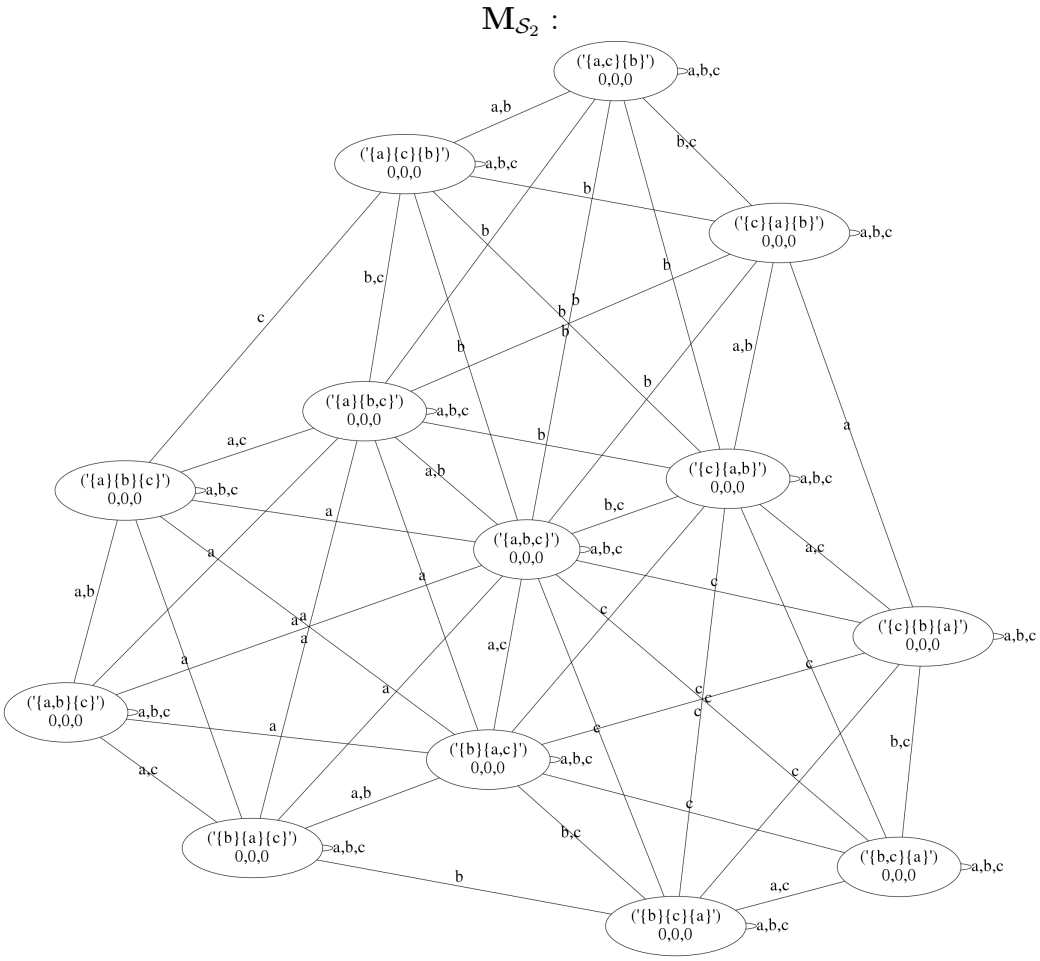

Figura 4.10: Modelo de acciones extendido $\mathbf{M}_{\mathcal{S}_2}$.

4.3.3. Subdivisión cromática de las facetas de un complejo simplicial y su modelo de acciones extendido

A partir de ahora pensemos en un complejo simplicial $\mathcal{C}$ que contenga todas las facetas correspondientes a las posibles ejecuciones del protocolo después de k iteraciones y el modelo de Kripke $\mathcal{M}_{\mathcal{C}} = \langle W_{\mathcal{C}}, R_{\mathcal{C}}, L_{\mathcal{C}} \rangle$ correspondiente a las facetas de $\mathcal{C}$. Similarmente consideremos a $\mathcal{S}$, la subdivisión cromática de $\mathcal{C}$, y $\mathcal{M}_{\mathcal{S}} = \langle W_{\mathcal{S}}, R_{\mathcal{S}}, L_{\mathcal{S}} \rangle$, el modelo de Kripke correspondiente

a las facetas de $\mathcal{S}$.

En seguida analizaremos cómo juntar las ideas en 4.3.1 y 4.3.2.1 para construir un modelo de acciones extendido $\mathbf{M}_{\mathcal{S}}$ que, operado con el modelo de Kripke $\mathcal{M}_{\mathcal{C}}$ mediante el producto modal restringido extendido, dé como resultado $\mathcal{M}_{\mathcal{S}}$.

En 4.3.2.1 se presentó el procedimiento para construir un modelo de acciones extendido que operado con un modelos de Kripke de un solo mundo genera el modelo de Kripke que representa las facetas de la subdivisión cromática de una sola faceta. Necesitamos construir un modelo de acciones extendido $\mathbf{M}_{\mathcal{S}} = \langle S, \sim, Pre, L\rangle$ tal que $\mathcal{M}_{\mathcal{S}} = \mathcal{M}_{\mathcal{C}} \bigotimes \mathbf{M}_{\mathcal{S}}$.

Tomemos en cuenta que lo que queremos es modelar las subdivisiones cromáticas de $\mathcal{C}$ en cada iteración de nuestros protocolos de estudio. En 4.3.1 mostramos una forma de identificar los mundos de un modelo de Kripke reducido bajo bisimulación y en 4.3.2.1 se describió como construir un modelo de acciones extendido para una sola faceta de $\mathcal{C}$.

El modelo de acciones extendido $\mathbf{M}_{\mathcal{S}}$ debe corresponder en estructura con $\mathcal{M}_{\mathcal{S}}$, salvo por Pre, L y $L_{\mathcal{S}}$.

Esto lo logramos construyendo un submodelo de acciones extendido para cada una de las facetas de $\mathcal{C}$ usando como identificador, adicionalmente al propuesto en 4.3.2, la fórmula de precondición que identifica a la faceta en $\mathcal{C}$. Los identificadores de los puntos de acción extendidos serían de la forma $\phi_\sigma - Conc$ donde $Conc$ define el orden y concurrencia de los procesos por ejemplo '$\{a\}\{b\}\{c\}$', '$\{a,b\}\{c\}$', o '$\{a,b,c\}$'.[3]

Con lo anterior tendríamos tantos submodelos de acciones extendidos como facetas en C, pero todos ellos desconectados entre sí. Para completar a $\mathbf{M}_{\mathcal{S}}$ presentamos los algoritmos 12 y 13.

El algoritmo 12 calcula el conocimiento que obtiene un agente después de la aplicación de una acción. En dicho algoritmo supondremos que $Conc$ es una lista de conjuntos de agentes a la que le podemos aplicar las funciones *valor* y *siguiente*. La función *valor* obtiene el conjunto de agentes que se tienen en la cabeza de la lista, mientras siguiente obtiene la cola de la lista.

Como entrada recibe:

- $\mathbf{M}_\sigma = \langle S_\sigma, \sim_\sigma, Pre_\sigma, L_\sigma\rangle$, el modelo de acciones
- $Id = \phi - Conc$, el identificador del punto de acción

[3]El apóstrofe se agrega por limitaciones del lenguaje Prolog.

- a, el agente del cual se quiere calcular el nuevo conocimiento

Como resultado obtiene:

- FK, el conjunto de proposiciones nuevas que conocerá el agente a después de aplicar la acción definida por el punto de acción identificado por Id en S_σ

Algoritmo 12 Algoritmo para calcular el nuevo conocimiento que tendrá el agente a después de aplicar la acción definida por el punto de acción identificado por Id.

```
1: function CálculoFK(M_σ = ⟨S_σ, ∼_σ, Pre_σ, L_σ⟩, Id = φ − Conc, a)
2:     l ← Conc
3:     Agentes ← ∅
4:     repeat
5:         Agentes ← Agentes ∪ valor(l)
6:         l ← siguiente(l)
7:     until a ∈ Agentes
8:     return FK = {p_{i,ag,v} | ag ∈ Agentes ∧ p_{0,ag,v} ∈ L_σ(Id)}
9: end function
```

El algoritmo 12 simplemente recorre la lista $Conc$ hasta encontrar al conjunto que contiene al agente a. Por la semántica de $Conc$, al terminar la iteración el agente a lee los valores escritos por todos los agentes que ejecutaron antes que él y al mismo tiempo, que son justamente los que se agregan al conjunto $Agentes$ en el algoritmo y por tanto FK es el conjunto de proposiciones nuevas que el agente a conocerá después de aplicar el punto de acción identificado por Id.

Para construir el modelo de acciones extendido $\mathbf{M}_\mathcal{S}$ presentamos el algoritmo 13 que usa como subrutina el algoritmo 12. Intuitivamente el algoritmo crea un modelo de acciones extendido que en principio es la unión de los submodelos de acciones extendidos para cada faceta de $\mathcal{C}$. Después relaciona los submodelos de acciones extendidos cuyas facetas correspondientes son adyacentes en $\mathcal{M}_\mathcal{C}$. Por último, añade las relaciones que hacen falta entre puntos de acciones de diferentes submodelos de acciones extendidos.

Como entrada recibe:

- $\mathcal{M}_\mathcal{C}$, el modelo de Kripke correspondiente a $\mathcal{C}$

- $SubModAccExt$, los submodelos de acciones extendidos correspondientes a las subdivisiones de las facetas de $\mathcal{C}$

Como resultado obtiene:

- $\mathbf{M}_{\mathcal{S}}$, el modelo de acciones extendido tal que $\mathcal{M}_{\mathcal{C}} \bigotimes \mathbf{M}_{\mathcal{S}} = \mathcal{M}_{\mathcal{S}}$

Algoritmo 13 Algoritmo para construir el modelo de acciones extendido $\mathbf{M}_{\mathcal{S}}$ a partir del modelo de Kripke $\mathcal{M}_{\mathcal{C}}$ y los submodelos de acciones extendidos $\mathbf{M}_{\sigma}$ $\forall \sigma \in facetas(\mathcal{C})$.

function ConstrucciónMAE($\mathcal{M}_{\mathcal{C}}, SubModAccExt$)
  $S_{\mathcal{S}} \leftarrow \bigcup_{\sigma \in facetas(\mathcal{C})} S_{\sigma}$
  $\sim_{\mathcal{S}} \leftarrow \bigcup_{\sigma \in facetas(\mathcal{C})} \sim_{\sigma}$
  $Pre_{\mathcal{S}} \leftarrow \bigcup_{\sigma \in facetas(\mathcal{C})} Pre_{\sigma}$
  $L_{\mathcal{S}} \leftarrow \bigcup_{\sigma \in facetas(\mathcal{C})} L_{\sigma}$
  for all $\mathbf{M}_{\sigma_w} = \langle S_{\sigma_w}, \sim_{\sigma_w}, Pre_{\sigma_w}, L_{\sigma_w} \rangle \in SubModAccExt$ **do**
    Sea $s \in S_{\sigma}$ un punto de acción de $\mathbf{M}_{\sigma}$
    $\phi \leftarrow Pre(s)$
    Sea $w \in W_{\mathcal{C}}$ el mundo identificado por ϕ en $\mathcal{M}_{\mathcal{C}}, w$
    for all $a \in Agentes$ **do** ▷ Conocimiento por agente en $\mathcal{M}_{\mathcal{C}}$
      $K_{a_w} = \{p_{i,ag,v} | \mathcal{M}_{\mathcal{C}}, w \models K_a(p_{i,ag,v})\}$
    end for
    $Vecindad = \{w' \in W_{\mathcal{C}} | (w, w') \in R_{\mathcal{C}}\} - \{w\}$
    for all $w' \in Vecindad$ **do**
      for all $a \in Agentes$ **do** ▷ Conocimiento por agente en $\mathcal{M}_{\mathcal{C}}, w'$
        $K_{a_{w'}} = \{p_{i,ag,v} | \mathcal{M}_{\mathcal{C}}, w' \models K_a(p_{i,ag,v})\}$
      end for
      Sea $\mathbf{M}_{\sigma_{w'}} \in SubModAccExt$ el correspondiente a w'
      for all $(s_{\sigma_w}, s_{\sigma_{w'}}) \in S_{\sigma_w} \times S_{\sigma_{w'}}$ **do**
        $Label = \emptyset$
        for all $a \in Agentes$ **do** ▷ Nuevo conocimiento
          $FK_{a,s_{\sigma_w}} =$ CálculoFK($\mathbf{M}_{\sigma_w}, s_{\sigma_w}, a$)
          $FK_{a,s_{\sigma_{w'}}} =$ CálculoFK($\mathbf{M}_{\sigma_{w'}}, s_{\sigma_{w'}}, a$)
          if $K_{a_w} \cup FK_{a,s_{\sigma_w}} = K_{a_{w'}} \cup FK_{a,s_{\sigma_{w'}}}$ **then**
            $Label \leftarrow Label \cup \{a\}$
          end if
        end for
        if $Label \neq \emptyset$ **then**
          $\sim_{\mathcal{S}} \leftarrow \sim_{\mathcal{S}} \cup \{(s_{\sigma_w}, Label, s_{\sigma_{w'}}), (s_{\sigma_{w'}}, Label, s_{\sigma_w})\}$
        end if
      end for
    end for
  end for
  return $\mathbf{M}_{\mathcal{S}} = \langle S_{\mathcal{S}}, \sim_{\mathcal{S}}, Pre_{\mathcal{S}}, L_{\mathcal{S}} \rangle$
end function

4.4. Tratamiento de dimensiones inferiores

En esta sección, discutimos una diferencia que hay entre los mapeos portadores y los modelos de acciones extendidos. En seguida presentamos las proyecciones de modelos de Kripke sobre un conjunto de agentes del mismo. Por último comentamos cómo las proyecciones obtienen las definiciones de los mapeos portadores en dimensiones inferiores.

4.4.1. Diferencia entre mapeos portadores y modelos de acciones extendidos

Como se mencionó anteriormente, existe una diferencia a primera vista importante entre los mapeos portadores y los modelos de acciones extendidos. Mientras que los mapeos portadores se pueden definir para cualquier simplejo de un complejo simplicial y por ello en cualquier dimensión del mismo, los modelos de acciones extendidos, al ser una lógica epistémica dinámica, buscan que los modelos de Kripke sean modelos epistémicos, lo que implica que los modelos de acciones extendidos están definidos solamente sobre el conjunto total de agentes del modelo, correspondientes a una sola dimensión en el complejo simplicial.

Recordemos que un mapeo portador $\Phi : A \to B$ manda cada simplejo $\sigma \in \mathcal{A}$ a un subcomplejo $\Phi(\sigma)$ de $\mathcal{B}$ tal que $\forall\, \sigma, \tau \in A,\ si\ \sigma \subseteq \tau$, entonces $\Phi(\sigma) \subseteq \Phi(\tau)$.

Como los mapeos portadores son monótonos, la información que existe en dimensiones inferiores del complejo simplicial debería poder ser obtenida de alguna forma desde modelo de Kripke correspondiente a las facetas del complejo. Esta idea la concretamos en 4.4.2, donde presentamos una proyección de modelos de Kripke sobre agentes, misma que podemos usar para obtener la definición de los mapeos portadores en dimensiones inferiores para un conjunto de procesos.

4.4.2. Proyecciones sobre agentes

Proponemos una proyección sobre un subconjunto de agentes de un modelo de Kripke que tiene como contraparte la proyección de complejos simpliciales presentada en 3.3. Sea $\mathcal{M} = \langle W, R, L\rangle$ un modelo de Kripke que represente las facetas de un complejo simplicial con las características en 3.1. Intuitivamente el proceso consiste en cambiar el etiquetamiento de los

mundos del modelo de Kripke de tal forma que las proposiciones correspondientes a la escritura de la última iteración en agentes no proyectados cambie de $p_{k,ag,v}$ a $p_{k,ag,\perp}$, donde k es el número de la última iteración, ag es cualquier agente no proyectado y v es la entrada del agente ag en cada mundo. Luego debemos eliminar a los agentes no proyectados de R. Por último debemos hacer una reducción del modelo de Kripke mediante bisimulación para eliminar la redundancia que se crea en el modelo de Kripke al cambiar el etiquetamiento de los mundos.

Presentamos el algoritmo 14, que formaliza la intuición de proyección de modelos de Kripke sobre un subconjunto de agentes del mismo.

Como entrada recibe:

- $\mathcal{M}$, el modelo de Kripke a proyectar
- $Agentes$, el subconjunto de agentes sobre los que se proyectará $\mathcal{M}$

Como resultado obtiene:

- $\mathcal{M}_{Agentes}$, el modelo de Kripke $\mathcal{M}$ proyectado sobre $Agentes$

Argumentemos que el algoritmo 14 es correcto. Podemos observar que en el ciclo de las líneas 6-12 se hace el cambio en el etiquetamiento para representar que los agentes $a \notin Agentes$ no escribieron en la última iteración y que en la línea 13 se agrega el etiquetamiento modificado para el mismo mundo que estaba originalmente en L a L'. Como L' se inicializa a $\emptyset$ en la línea 2, al final del ciclo de las lineas 6-12 el etiquetamiento representado en L' contiene el nuevo etiquetamiento de los mundo de $\mathcal{M}$ en la forma deseada. Respecto a la relación de accesibilidad tenemos que en el ciclo de las lineas 16-20 se procesa a cada tripleta en la relación agregando a R' únicamente a las relaciones (w_1, Ags', w_2) tales que $(w_1, Ags, w_2) \in R \ \wedge \ (Ags' \leftarrow Ags \cap Agentes) \neq \emptyset$ y como R' se inicializa a $\emptyset$ en la linea 15, al final del ciclo de las lineas 16-20 $R' \subseteq R$ contendrá a todas las relaciones de accesibilidad como las deseamos. Finalmente en la línea 21 se procesa el modelo de Kripke $\langle W, R', L' \rangle$ bajo bisimulación para eliminar la redundancia creada por el reetiquetamiento de los mundos y el resultado se guarda en $M_{Agentes}$, por lo que al final de la ejecución $\mathcal{M}_{Agentes}$ tendrá la proyección de $\mathcal{M}$ sobre $Agentes$.

Algoritmo 14 Algoritmo para proyectar un modelo de Kripke $\mathcal{M}$ sobre $Agentes$.

1: **function** PROYECCIÓNMK($\mathcal{M} = \langle W, R, L\rangle, Agentes$)
2: $L' \leftarrow \emptyset$
3: **for all** $(id, Et) \in L$ **do** ▷ Cambio de etiquetado
4: $k \leftarrow$ el número de la última iteración que representa $\mathcal{M}$
5: $Et' \leftarrow \emptyset$
6: **for all** $p_{i,a,v} \in Et$ **do**
7: **if** $i = k \wedge a \notin Agentes$ **then**
8: $Et' \leftarrow Et' \cup \{p_{k,a,\perp}\}$
9: **else**
10: $Et' \leftarrow Et' \cup \{p_{i,a,v}\}$
11: **end if**
12: **end for**
13: $L' \leftarrow L' \cup \{(id, Et')\}$
14: **end for**
15: $R' \leftarrow \emptyset$
16: **for all** $(w_1, Ags, w_2) \in R$ **do** ▷ Cambio en la relación
17: $Ags' \leftarrow Ags \cap Agentes$
18: **if** $Ags' \neq \emptyset$ **then** $R' \leftarrow R' \cup \{(w_1, Ags', w_2)\}$
19: **end if**
20: **end for**
21: $\mathcal{M}_{Agentes} \leftarrow$ MINBISIMULACIÓN($\langle W, R', L'\rangle$)
22: **return** $\mathcal{M}_{Agentes}$
23: **end function**

4.4.3. Definiciones de mapeos portadores en dimensiones inferiores

Ahora mostraremos cómo la proyección presentada en 4.4.2 nos permite obtener la estructura de los mapeos portadores en dimensiones inferiores.

Tomemos en cuenta el complejo simplicial que corresponde todas las ejecuciones de uno de nuestros protocolos de estudio cuyos procesos tienen entradas binarias y el modelo de Kripke correspondiente a sus facetas. EL complejo y el modelo de Kripke mencionados son muy grandes. Solamente para dimensionar su tamaño están representados en las figuras 4.11 y 4.12 respectivamente.

La proyección del complejo simplicial de la figura 4.11 dado el supuesto de que solamente los procesos a y b, coloreados con gris oscuro y gris claro respectivamente, siguen vivos se muestra en la figura 4.13, y el modelo de Kripke que le corresponde a sus facetas de la proyección se muestra en la figura 4.14.

Por otro lado, la proyección del modelo de Kripke en la figura 4.12 sobre los agentes a y b se muestra en la figura 4.15. Las clases de equivalencia que resultan de la bisimulación del modelo en la figura 4.12 después del reetiquetamiento de los mundos están dadas por la siguiente partición:
$\{103, 102, 101, 99, 97, 90, 89, 88, 86, 84\}$
$\{100, 96, 94, 87, 83, 81\}$
$\{98, 95, 93, 92, 91, 85, 82, 80, 79, 78\}$
$\{77, 76, 75, 73, 71, 64, 63, 62, 60, 58\}$
$\{74, 70, 68, 61, 57, 55\}$
$\{72, 69, 67, 66, 65, 59, 56, 54, 53, 52\}$
$\{51, 50, 49, 47, 45, 38, 37, 36, 34, 32\}$
$\{48, 44, 42, 35, 31, 29\}$
$\{46, 43, 41, 40, 39, 33, 30, 28, 27, 26\}$
$\{25, 24, 23, 21, 19, 12, 11, 10, 8, 6\}$
$\{22, 18, 16, 9, 5, 3\}$
$\{20, 17, 15, 14, 13, 7, 4, 2, 1, 0\}$

Observemos que los modelos de Kripke en las figuras 4.14 y 4.15 son isomorfos[4]. Además ambos corresponden a la subdivisión cromática para dos procesos con entradas en $\{0, 1\}$ salvo por la memoria que correspondería al

[4]La única diferencia entre los modelos son los identificadores, pues en la figura 4.15, al ser una reducción bajo bisimulación, el identificador es la clase de equivalencia calculada.

proceso, o etiquetamiento del agente, c. Lo anterior describe la estructura inducida por la función de mapeo portador aplicada a los simplejos de dimensión de dimensión 1, cuyos vértices corresponden a los procesos a y b en el complejo de entrada. Es decir: si $\Delta : \mathcal{C} \longrightarrow 2^{\mathcal{C}'}$ es un mapeo portador y $\mathcal{M}_{\mathcal{C}'}$ es el modelo de Kripke correspondiente a las facetas de $\mathcal{C}'$, entonces ProyecciónMK$(\mathcal{M}, Agentes)$ es isomorfo al modelo de Kripke correspondiente a las facetas de $\bigcup_{\sigma \in \mathcal{C} \wedge colores(\sigma) = Agentes} \Delta(\sigma)$.

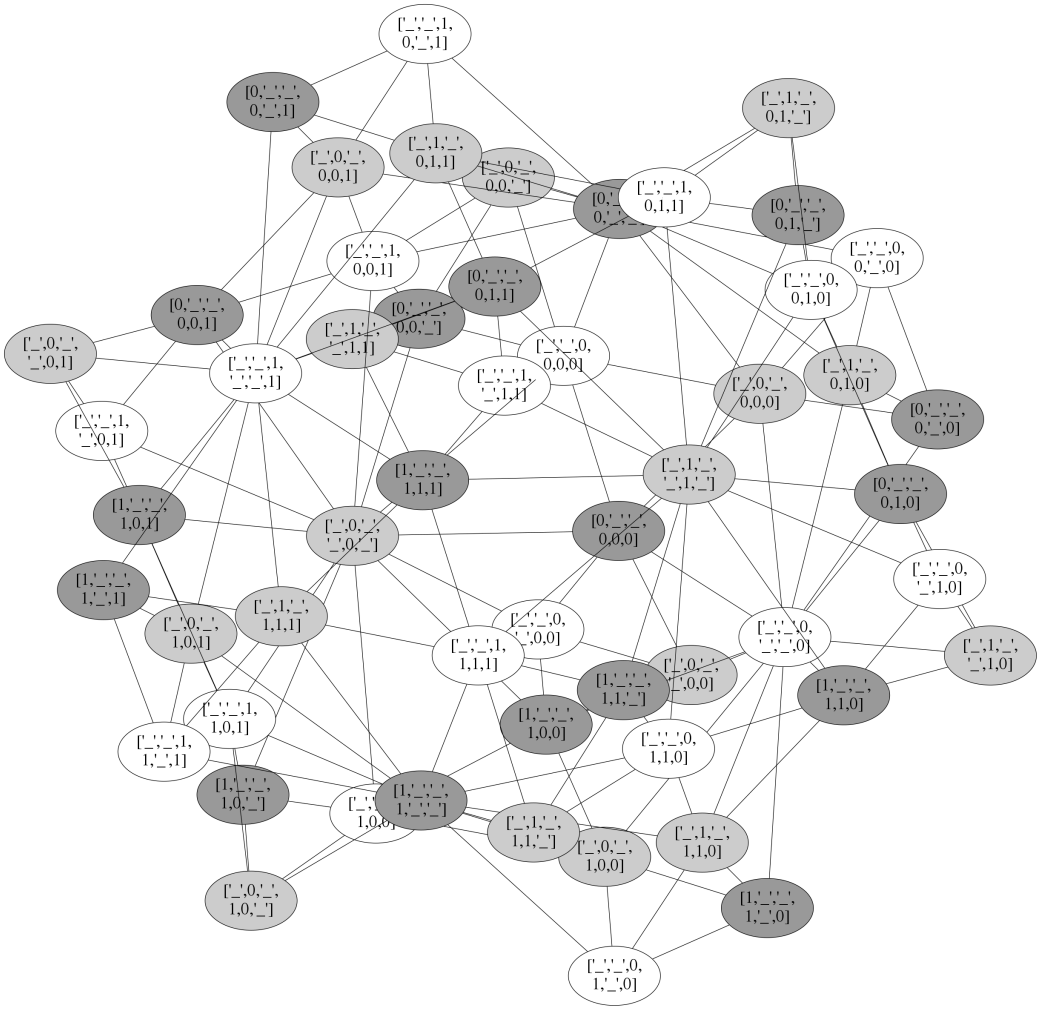

Figura 4.11: Complejo simplicial de la subdivisión crómatica para tres procesos con entrada binaria.

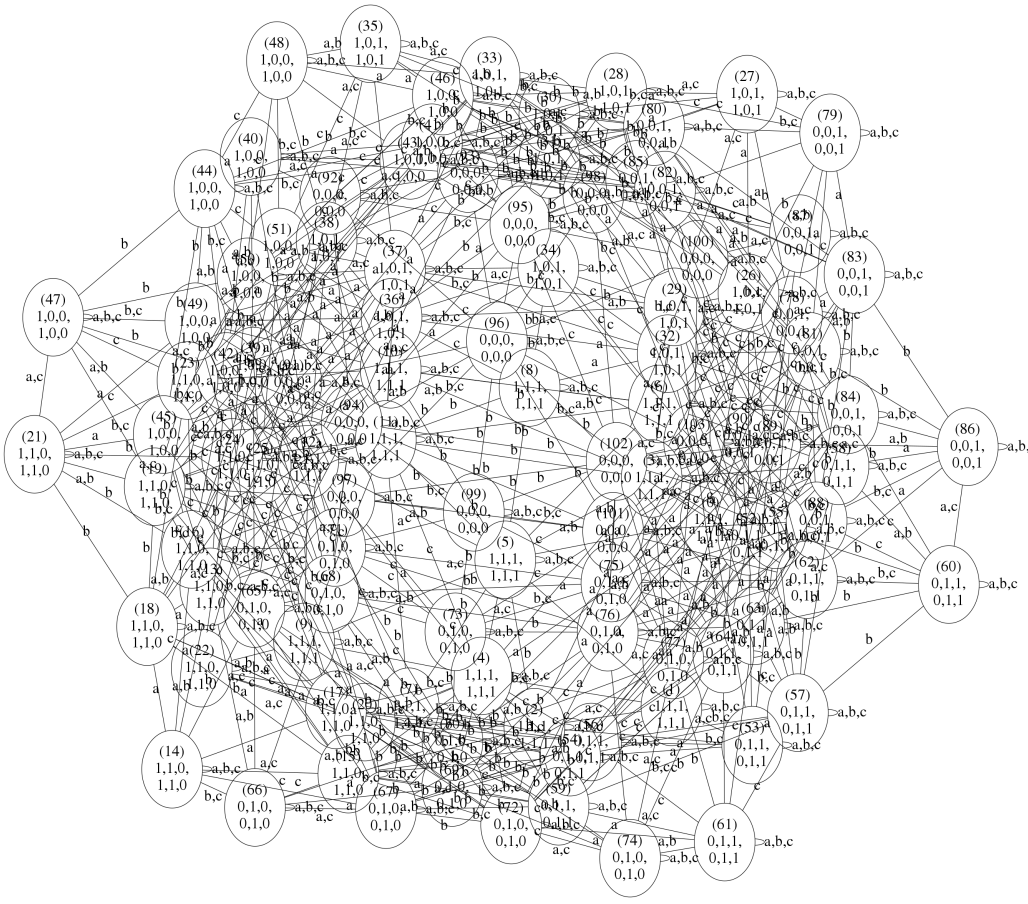

Figura 4.12: Modelo de Kripke correspondiente al complejo simplicial en 4.11.

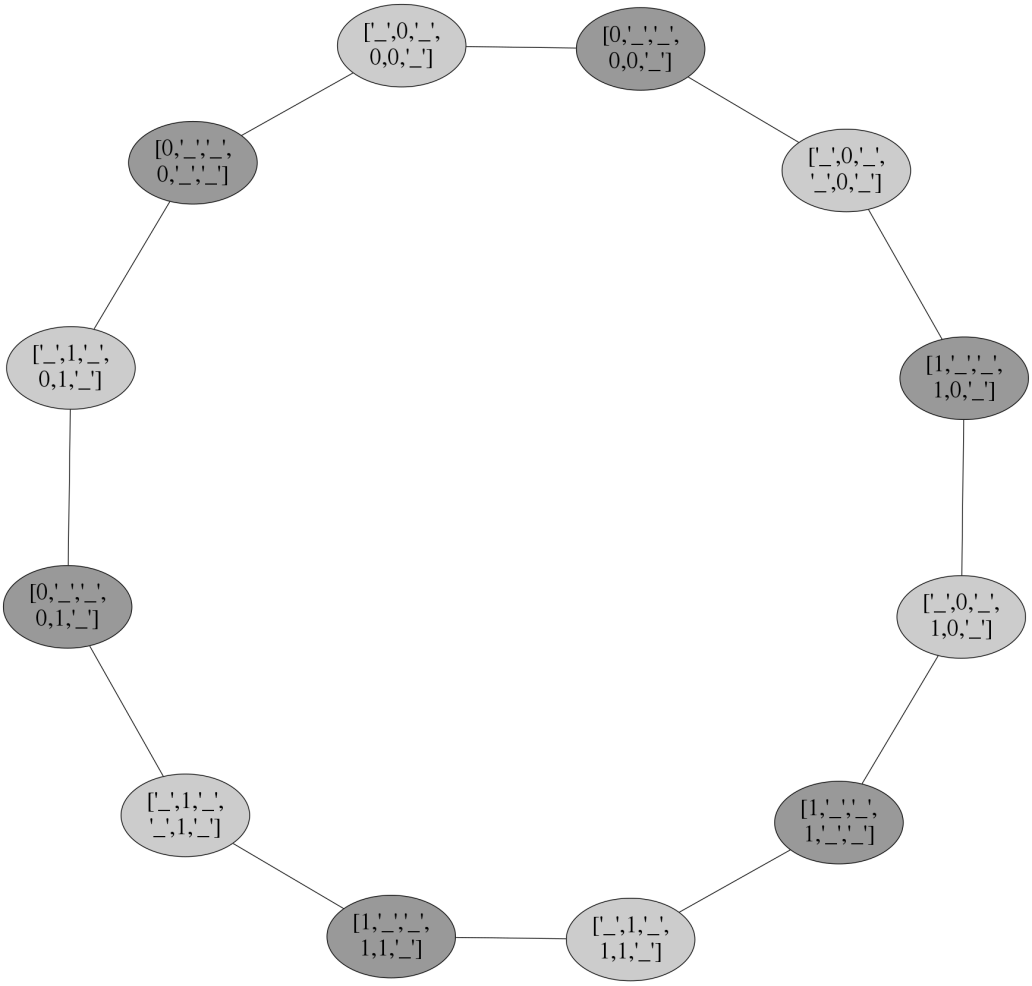

Figura 4.13: Proyección del complejo simplicial de la figura 4.11 sobre los procesos a y b.

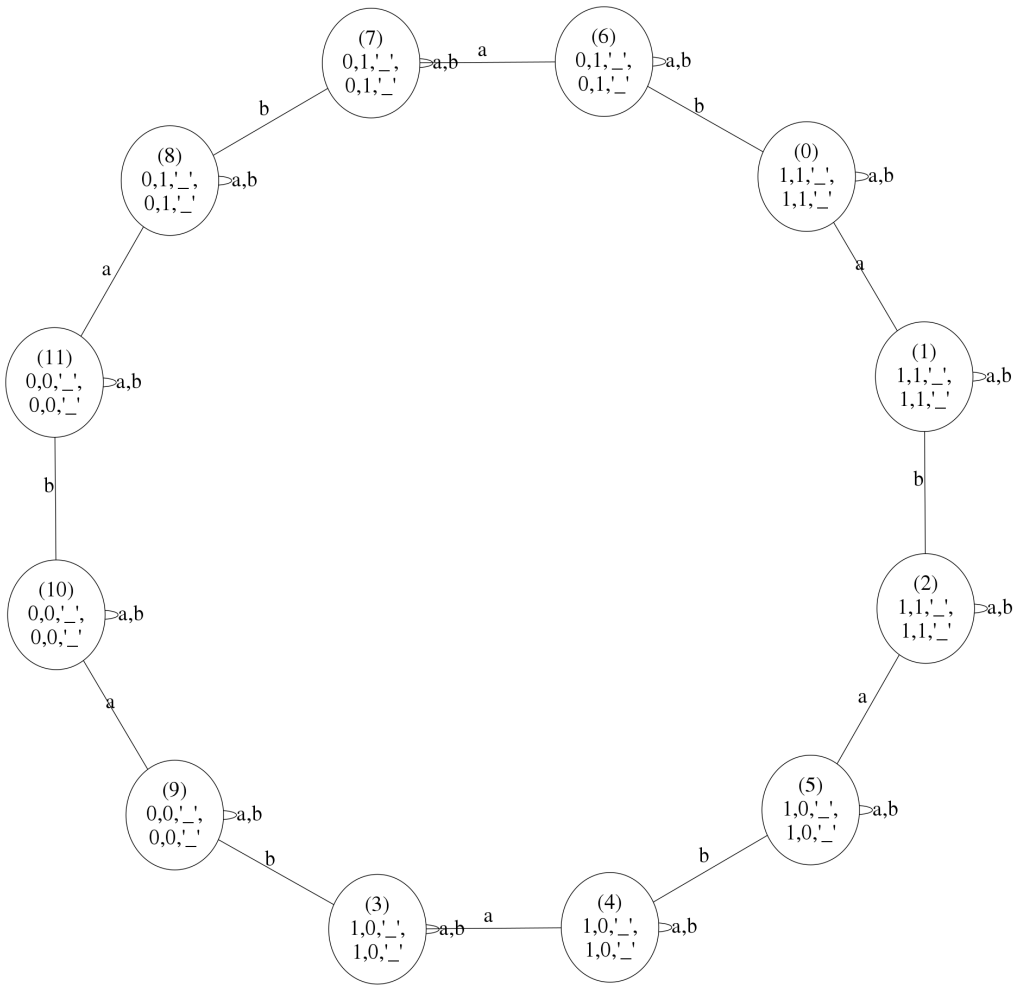

Figura 4.14: Modelo de Kripke correspondiente a las facetas del complejo simplicial de la figura 4.13.

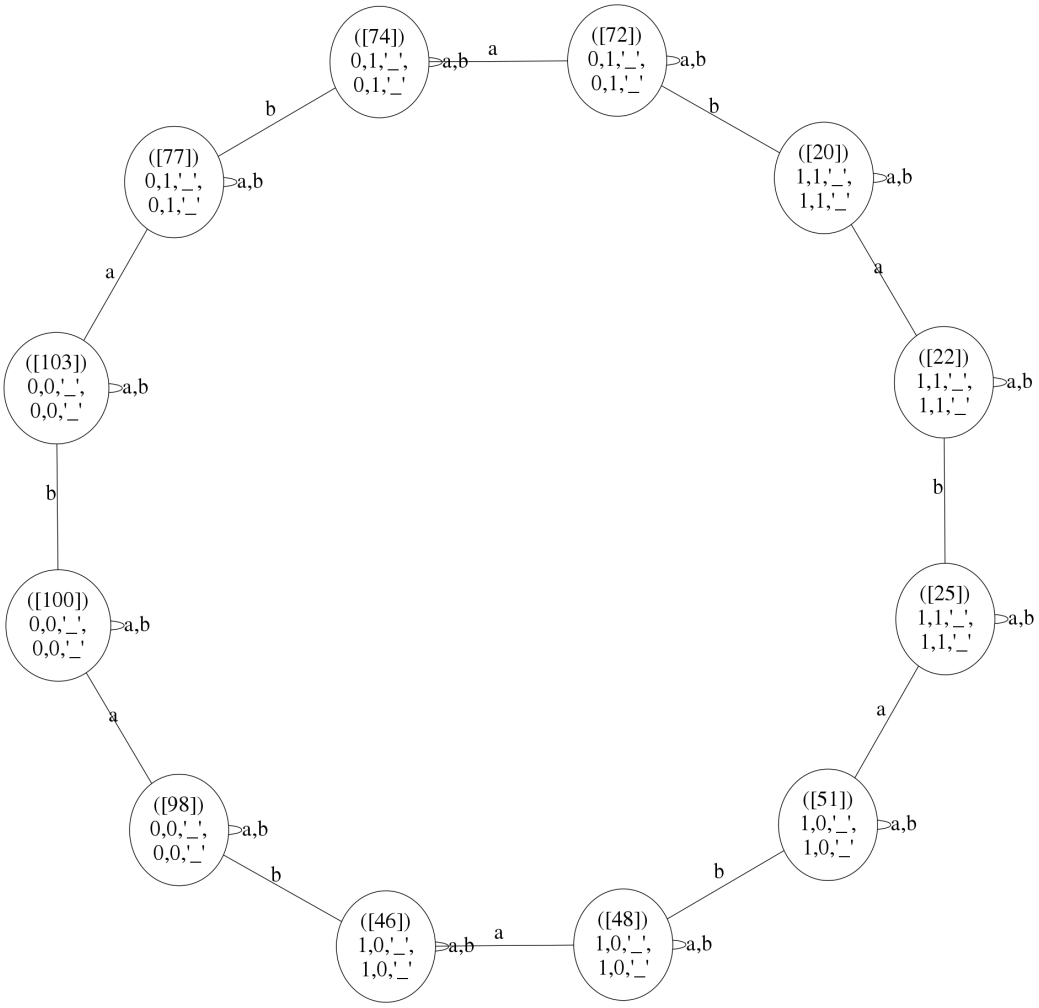

Figura 4.15: Proyección del modelo de Kripke en la figura 4.12 sobre los agentes a y b.

Capítulo 5

Conclusiones

En este trabajo mostramos una relación entre el enfoque de las lógicas modales y el enfoque topológico del cómputo distribuido. Por simplicidad estudiamos un tipo particular de protocolos en el modelo *Iterated Immediate Snapshots*. En dichos protocolos, los procesos escriben únicamente su entrada en cada iteración, pero tienen memoria local infinita donde guardan lo que leen en todas las iteraciones que ejecutan. Analizamos ejecuciones acotadas de nuestros protocolos de estudio representadas en modelos de Kripke, desde el enfoque de las lógicas modales, y en complejos simpliciales, desde el enfoque topológico del cómputo distribuido. Propusimos algoritmos para transformar nuestras representaciones en modelos de Kripke a complejos simpliciales y viceversa. Luego analizamos una relación entre los mapeos portadores y los modelos de acciones extendidos desde la forma en que ambos definen la evolución de las estructuras correspondientes.

Más precisamente, diseñamos algoritmos para transformar modelos de Kripke a conjuntos de simplejos y viceversa. Expresamos los algoritmos desde el punto de vista de las lógicas modales. A pesar de que en dichos algoritmos el cálculo del etiquetamiento de los mundos, en los modelos de Kripke, y el cálculo de vistas de los vértices, en los simplejos, se diseñaron específicamente para nuestras representaciones en las estructuras mencionadas, las transformaciones se pueden adecuar para representar protocolos de información completa representando la información leída en la iteración i como un contenedor de i niveles, donde el contenedor de nivel 0 guarda la entrada de un proceso y uno de nivel $k+1$ guarda la referencia a otro de nivel k.

Después de que analizamos la forma en que los mapeos portadores y los modelos de acciones extendidos definen la evolución en las estructuras co-

rrespondientes a cada enfoque, encontramos una diferencia entre los mapeos portadores y los modelos de acciones extendidos. Los modelos de acciones extendidos describen la evolución de los modelos de Kripke sobre el conjunto total de agentes mientras que los mapeos portadores pueden definirse sobre cualquier simplejo. Propusimos proyecciones sobre modelos de Kripke para recuperar la definición de los mapeos portadores sobre simplejos que no representan a todos los procesos, tomando como punto de partida que los mapeos portadores preservan contención.

Un observación que obtuvimos es que los modelos de acciones extendidos podrían no ser una lógica epistémica dinámica adecuada para calcular modelos de Kripke correspondientes a subdivisiones cromáticas de complejos simpliciales, pues su tamaño y estructura corresponde al del modelo de Kripke que se quiere obtener. Consideramos que un trabajo a futuro importante es proponer una lógica epistémica dinámica que no aumente el tamaño de sus representación al mismo ritmo que los modelos que se quieren estudiar, ya que ello resultaría en una lógica epistémica dinámica más práctica en su uso.

También dejamos como trabajo futuro diseñar una lógica modal, con una semántica similar a la lógica epistémica proposicional, que permita que las relaciones de accesibilidad de los agentes en los modelos de Kripke pierdan la propiedad de ser reflexivas. Si diseñamos una lógica modal que permita dicha característica, podríamos verificar modelos correspondientes a cualquier conjunto de simplejos del complejo simplicial. Lo anterior es de utilidad para estudiar modelos de cómputo cuyas ejecuciones se representan en complejos simpliciales no necesariamente puros.

Apéndice A

Implementación

En este apéndice, ofrecemos una versión literaria de los predicados en la implementación, en lenguaje Prolog, que desarrollamos en este trabajo. En la sección A.1 describimos la representación de las estructuras con las que trabajamos. En la sección A.2 presentamos el predicado para proyectar complejos simpliciales sobre procesos vivos. En la sección A.3 exponemos el predicado de transformación de conjuntos de simplejos a modelos de Kripke. En la sección A.4 mostramos el predicado para transformar un modelo de Kripke a un conjunto de simplejos. En la sección A.5 mencionamos el predicado para calcular la proyección de un modelos de Kripke sobre un conjunto de agentes. En la sección A.6 exhibimos el verificador de modelos epistémicos que implementamos. Por último, en la sección A.7 enunciamos los predicados generadores de gráficas para representar complejos simpliciales y modelos de Kripke y modelos de acciones extendidos con los que generamos la mayoría de las figuras de este trabajo.

A.1. Representación

A.1.1. Complejos simpliciales

Recordemos que los complejos simpliciales son un conjunto de simplejos y que cada simplejo es un conjunto. Por lo anterior, representamos los complejos simpliciales como listas de listas de vértices, donde cada una de ellas es un conjunto.

Los vértices de los complejos simpliciales que tratamos se identifican por

su color y su vista, de tal forma que representamos su estructura con el término `v(Color,Vista)`. Los colores pueden ser cualquier conjunto de átomos en Prolog. Por ejemplo, en este trabajo usamos el conjunto $\{a, b, c\}$ para denotar que un complejo simplicial se colorea con los procesos a, b y c. Las vistas en nuestra implementación son listas de lo que un proceso ve, representadas de una forma conveniente[1]. Por ejemplo en el complejo simplicial de los niños lodosos de la figura A.1 la vista `[0,'_',0,'_']` corresponde a que el proceso a tiene como entrada el valor 0 y después de la iteración ve que él escribió su entrada, pero no vio nada escrito en la memoria correspondiente al proceso b. El átomo `'_'` en esta representación corresponde a la ausencia de información.

La siguiente lista representa el complejo simplicial de la figura A.1:

```
[
    [],
    [v(b,['_',0,0,0])],
    [v(a,[0,'_',0,0])],
    [v(a,[0,'_',0,'_']),v(b,['_',0,0,0])],
    [v(a,[0,'_',0,0]),v(b,['_',0,0,0])],
    [v(a,[0,'_',0,0]),v(b,['_',0,'_',0])],
    [v(a,[0,'_',0,'_'])],
    [v(b,['_',1,0,1])],
    [v(a,[0,'_',0,1])],
    [v(a,[0,'_',0,'_']),v(b,['_',1,0,1])],
    [v(a,[0,'_',0,1]),v(b,['_',1,0,1])],
    [v(a,[0,'_',0,1]),v(b,['_',1,'_',1])],
    [v(b,['_',0,1,0])],
    [v(a,[1,'_',1,0])],
    [v(b,['_',0,'_',0])],
    [v(a,[1,'_',1,'_']),v(b,['_',0,1,0])],
    [v(a,[1,'_',1,0]),v(b,['_',0,1,0])],
    [v(a,[1,'_',1,0]),v(b,['_',0,'_',0])],
    [v(a,[1,'_',1,'_'])],
    [v(b,['_',1,1,1])],
    [v(a,[1,'_',1,1])],
    [v(b,['_',1,'_',1])],
    [v(a,[1,'_',1,'_']),v(b,['_',1,1,1])],
```

[1] Lo conveniente depende de cómo se calcule el etiquetamiento de los modelos de Kripke.

```
    [v(a,[1,'_',1,1]),v(b,['_',1,1,1])],
    [v(a,[1,'_',1,1]),v(b,['_',1,'_',1])]
]
```

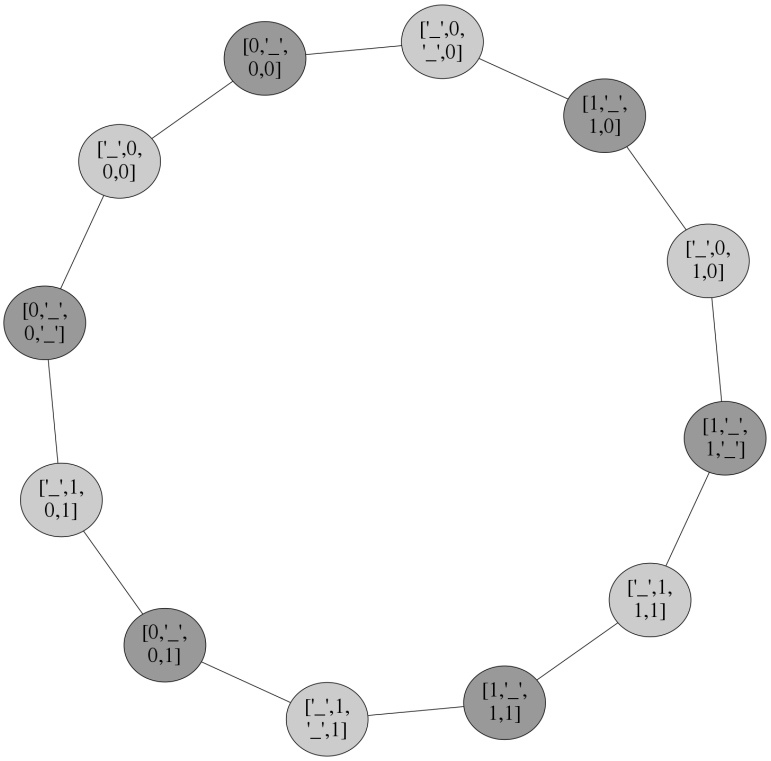

Figura A.1: Complejo simplicial para la subdivisión cromática de 2 procesos después de una iteración.

A.1.2. Modelos de Kripke

Recordemos que un modelo de Kripke $\mathcal{M} = \langle W, R, L \rangle$ sobre el conjunto de proposiciones P y los agentes *Agentes* es una estructura que consta de 3 elementos: W, un conjunto no vacío de mundos, $R \subseteq W \times 2^{Agentes} \times W$, la relación de accesibilidad de los agentes a los mundos y $L : W \longrightarrow 2^P$, una función de etiquetamiento.

En nuestra implementación, representamos los modelos de Kripke con el término $model(Worlds, Edges)$ donde *Worlds* es una lista de mundos y

su respectivo etiquetamiento y *Edges* es una lista de flechas entre mundos etiquetadas por un conjunto no vacío de agentes englobados.

Cada elemento de la lista *Worlds* es representado con el término $w(Id, Label)$. El *Id* es un identificador del mundo y *Label* es el etiquetamiento del mundo. En nuestro caso, al estudiar un modelo iterado con memoria compartida, representamos el etiquetamiento con el estado de la memoria representado como una lista de tamaño $(k+1) \times n$ donde k es el número de iteraciones ejecutadas y n es el número de agentes que participan en el protocolo. Los primeros n elementos de la lista corresponderán a los valores de entrada de cada uno de los agentes antes de iniciar la ejecución del protocolo y las siguientes corresponderán a los valores escritos en memoria compartida.

Cada flecha es representada por el término $edge(IdS, Ags, IdT)$, donde IdS corresponde al identificador del mundo de origen, IdT al mundo de destino y $Ags \subseteq Agentes$ es el conjunto de agentes con el que está etiquetada la arista.

Por ejemplo, la siguiente representación corresponde al modelo de Kripke de la figura A.2:

```
model(
    [
        w(11,[0,0,0,0]),
        w(10,[0,0,0,0]),
        w(9,[0,0,0,0]),
        w(8,[0,1,0,1]),
        w(7,[0,1,0,1]),
        w(6,[0,1,0,1]),
        w(5,[1,0,1,0]),
        w(4,[1,0,1,0]),
        w(3,[1,0,1,0]),
        w(2,[1,1,1,1]),
        w(1,[1,1,1,1]),
        w(0,[1,1,1,1])
    ],
    [
        edge(11,[a,b],11),
        edge(11,[b],10),
        edge(11,[a],8),
        edge(10,[b],11),
```

```
        edge(10,[a,b],10),
        edge(10,[a],9),
        edge(9,[a],10),
        edge(9,[a,b],9),
        edge(9,[b],3),
        edge(8,[a],11),
        edge(8,[a,b],8),
        edge(8,[b],7),
        edge(7,[b],8),
        edge(7,[a,b],7),
        edge(7,[a],6),
        edge(6,[a],7),
        edge(6,[a,b],6),
        edge(6,[b],0),
        edge(5,[a,b],5),
        edge(5,[b],4),
        edge(5,[a],2),
        edge(4,[b],5),
        edge(4,[a,b],4),
        edge(4,[a],3),
        edge(3,[b],9),
        edge(3,[a],4),
        edge(3,[a,b],3),
        edge(2,[a],5),
        edge(2,[a,b],2),
        edge(2,[b],1),
        edge(1,[b],2),
        edge(1,[a,b],1),
        edge(1,[a],0),
        edge(0,[b],6),
        edge(0,[a],1),
        edge(0,[a,b],0)
    ]
)
```

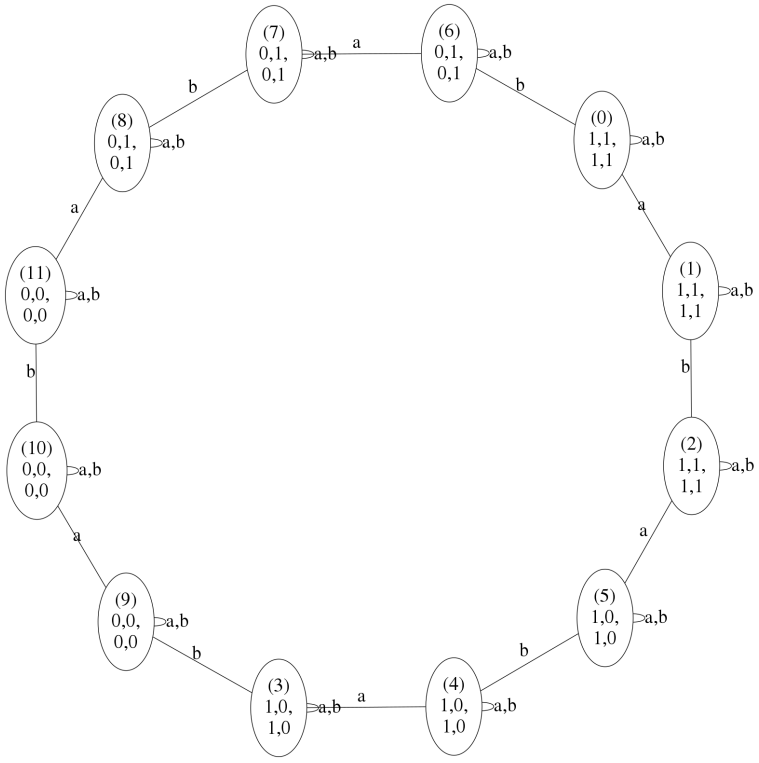

Figura A.2: Modelo de Kripke para la subdivisión cromática de 2 procesos después de una iteración.

A.1.3. Modelos de acciones extendidos

Recordemos que los modelos de acciones extendidos sobre el conjunto de agentes *Agentes* son una estructura $\mathbf{M} = \langle S, \sim, pre, L \rangle$, donde S es un conjunto no vacío de puntos de acciones, $\sim \ \subseteq S \times 2^{Agentes} \times S$ es la relación de indistinguibilidad de acciones de los agentes, *pre* es la función de precondición y L es la función de etiquetamiento.

Representamos los modelos de acciones extendidos con el término `extended_action_model(EAPs,Es)`. La variable `EAPs` es una lista de elementos englobados por el término `eap(Id,Pre,Label)`, donde `Id` representa el identificador del punto de acción, `Pre` su función de precondición y `Label` su etiquetamiento. `Es` es una lista de arcos entre puntos de acciones etiquetados por un conjunto de agentes $A \subseteq Agentes$. Por ejemplo, la siguiente representación corresponde al modelo de acciones extendido de la figura A.3,

cuya función de precondición pre está definida como sigue:
$pre('ba') = pre('a, b') = pre('ab') = true$

```
extended_action_model(
    [
        eap('{b}{a}',1,[0,0]),
        eap('{a,b}',1,[0,0]),
        eap('{a}{b}',1,[0,0])
    ],
    [
        edge('{a}{b}',[a,b],'{a}{b}'),
        edge('{a,b}',[a,b],'{a,b}'),
        edge('{b}{a}',[a,b],'{b}{a}'),
        edge('{a}{b}',[b],'{a,b}'),
        edge('{a,b}',[b],'{a}{b}'),
        edge('{b}{a}',[a],'{a,b}'),
        edge('{a,b}',[a],'{b}{a}')
    ]
)
```

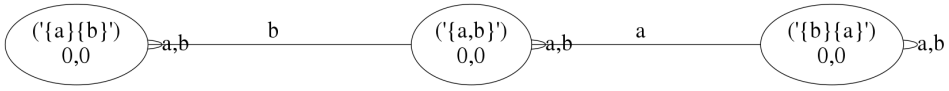

Figura A.3: Ejemplo de modelo de acciones extendido.

Notemos que la representación gráfica del modelo de acciones extendido no muestra la función de precondición. Eso se decidió debido a que la fórmula de precondición de los mundos puede ser arbitrariamente grande, lo que saturaría la imagen.

A.2. Proyecciones de complejos simpliciales sobre procesos vivos

Para obtener las proyecciones de un complejo simplicial de n procesos sobre algún subconjunto, como se presentó en la sección 3.3, se implementó el predicado `proy_complex(Complex,ProcsCols,NComplex)`. La variable `Complex` denota el complejo simplicial a proyectar. La variable `ProscCols`

representa una lista de elementos de la forma `Proc=>Col`, donde `Proc` es un proceso y `Col` es la posición en la que `Proc` escribe módulo n. Finalmente, la variable `NComplex` denota el complejo proyectado. Por ejemplo, para obtener el complejo simplicial de la figura A.5 a partir del complejo simplicial de la figura A.4 podemos llamar al predicado `proy_complex` de la siguiente forma:

```
proy_complex(Complex000,[a=>0,b=>1],Complex00_)
```

En la llamada anterior la variable `Complex000` debe estar unificada con una representación del complejo simplicial de la figura A.4. El predicado calculará una representación del complejo simplicial proyectado y la unificará con la variable `Complex00_`.

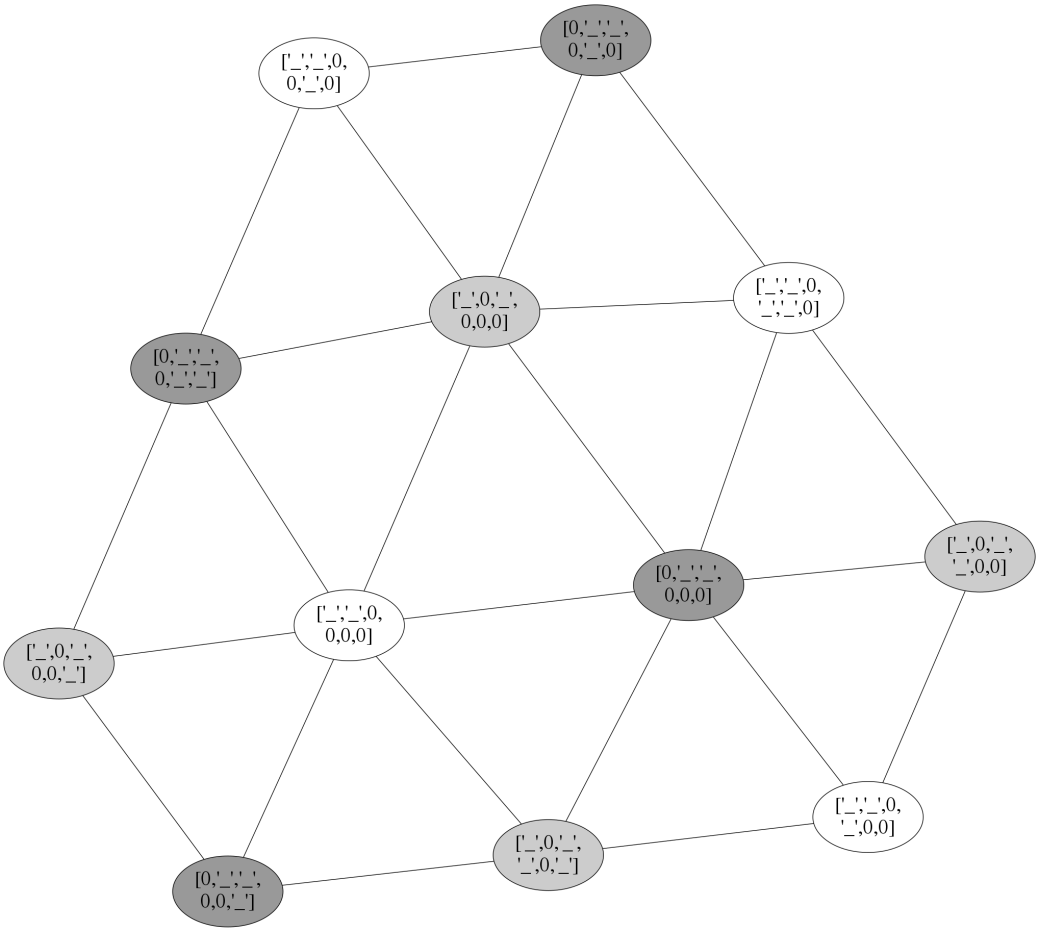

Figura A.4: Complejo simplicial para una iteración del modelo *Iterated Immediate Snapshots* de tres procesos con entrada 0.

Figura A.5: Proyección del complejo simplicial en la figura A.4 sobre los agentes a y b.

A.3. Conjuntos de simplejos a modelos de Kripke

Para transformar conjuntos de simplejos a modelos de Kripke implementamos el predicado `simplices_to_model(Simplices,Model)`, donde `Simplices` es una lista de simplejos y `Model` es el modelo de Kripke calculado a partir del algoritmo 8.

Muchas veces es deseable obtener el modelo de Kripke correspondiente a las facetas de un modelo de Kripke. Implementamos también el predicado `complex_to_model(Complex,Model)` para obtener la representación del conjunto de facetas de un complejo simplicial. El predicado `complex_to_model` calcula las facetas del complejo simplicial y llama a `simplices_to_model` para calcular el modelo de Kripke que le corresponde a las facetas.

Por ejemplo, para calcular el modelo de Kripke de la figura A.2 correspondiente a las facetas del complejo simplicial de la figura A.1 podemos llamar al predicado `complex_to_model(Complex,Model)` con la variable `Complex` unificada con la representación mostrada en la subsección A.1.1 y el predicado `complex_to_model` calculará una representación del modelo de Kripke, la unificará la variable `Model` y se evaluará a *true*.

A.4. Modelos de Kripke a Conjuntos de simplejos

Para calcular los simplejos correspondientes a los mundos de un modelo de Kripke y el complejo mínimo que los contiene podemos utilizar el predicado
`model_to_simplices_and_complex(Model,Simplices,Complex)`. A partir de la variable `Model` unificada con una representación del modelo de Kripke, dicho predicado calcula los simplejos correspondientes a los mundos del modelo de Kripke y los unifica la variable `Simplices`. Posteriormente calcula el complejo simplicial mínimo que contiene a los simplejos y lo unifica con la variable `Complex`.

Además, se definieron los predicados `model_to_simplices(Model,Simplices)` y `model_to_complex(Model,Complex)` para obtener únicamente el conjunto de simplejos o el complejo simplicial mínimo que lo contiene.

Por ejemplo, para obtener el complejo simplicial de la figura A.1, con la

variable `Model` unificada con una representación del modelo de Kripke de la figura A.2, podemos llamar al predicado de la siguiente forma:

```
model_to_complex(Model,Complex)
```

El predicado `model_to_complex` llamará al predicado `model_to_simplices_and_complex` y unificará el complejo simplicial calculado con la variable `Complex`.

A.5. Proyecciones de modelos de Kripke sobre agentes

Para calcular las proyecciones sobre agentes de los modelos de Kripke, se implementó el predicado `model_proy(Model,AgentsWithPosition,NewModel)`. A partir de la variable `Model` unificada con alguna representación del modelo de Kripke de la forma expuesta en A.1.2 y la variable `AgentsWithPosition` unificada con una lista que contiene a los agentes sobre los que se está proyectando así como su columna correspondiente módulo n, el predicado construye la representación del modelo de Kripke y la unifica con la variable `NewModel`.

Por ejemplo, si la variable `Model` está unificada con una representación del modelo de Kripke de la figura A.6, que es un modelos de Kripke muy grande, y la variable `AgentsWithPosition` está unificada con la lista `[a=>0,b=>1]`. Al llamar al predicado como sigue:
`model_proy(Model,AgentsWithPosition,NewModel)` en el intérprete de Prolog se calcula la representación del modelo de Kripke en la figura A.7 y la unifica con la variable `NewModel`.

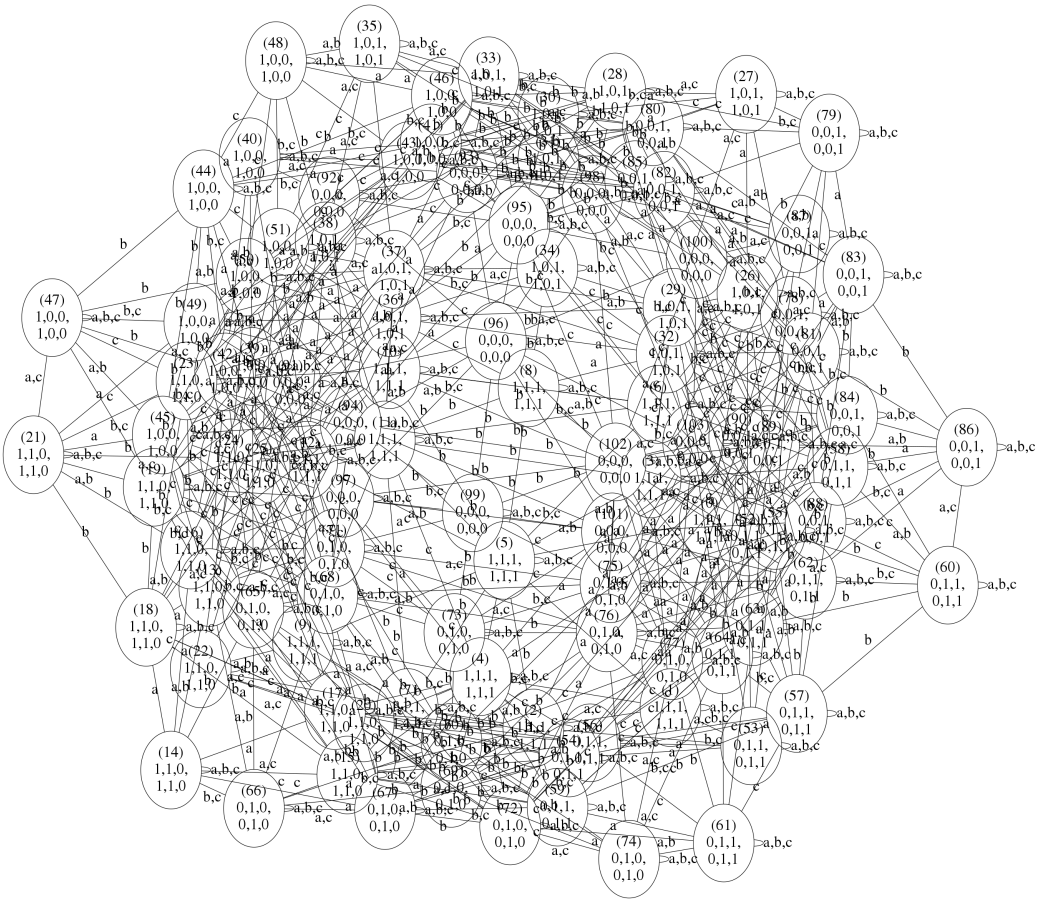

Figura A.6: Modelo de Kripke correspondiente al complejo simplicial en la figura4.11.

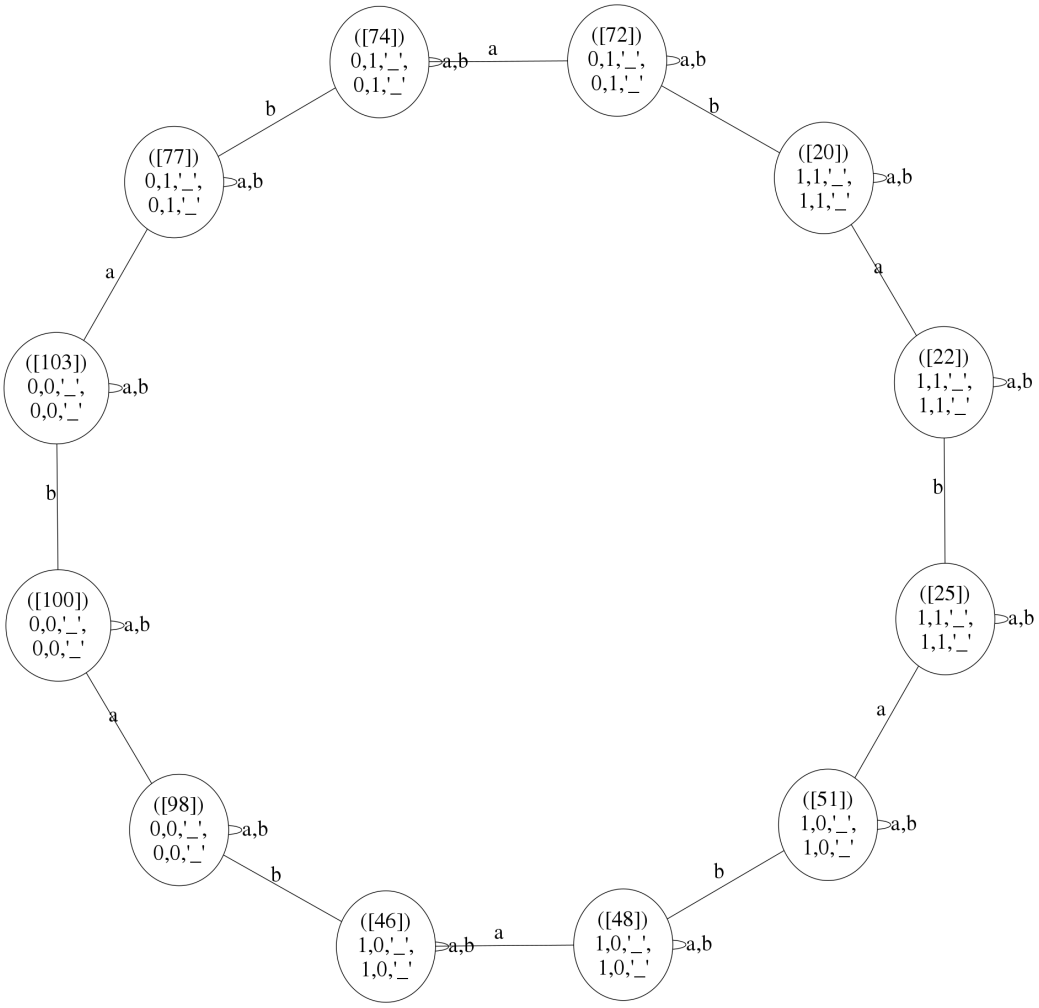

Figura A.7: Proyección del modelo de Kripke en la figura A.6 sobre los agentes a y b.

A.6. Verificador de modelos epistémicos

Debido a que la lógica epistémica es de particular importancia, en este trabajo implementamos un verificador de modelos epistémicos.

Consideremos un modelo epistémico sobre un conjunto de agentes *Agentes*.

La sintaxis con la que trabaja el verificador es la siguiente:

- $c := 0 \mid 1$
- $p := p(Index) \mid p(Index, Value) \mid p(NAgents, Round, CAgent, Value)$
- $\varphi := c \mid p \mid not\ \varphi \mid \varphi\ and\ \varphi \mid \varphi\ or\ \varphi \mid \varphi\ =>\ \varphi \mid \varphi\ <=>\ \varphi \mid k(a, \varphi) \mid e(A, \varphi) \mid c(A, \varphi) \mid d(A, \varphi)$

En la sintaxis anterior c corresponde a las constantes 0 para $\bot$ y 1 para $\top$. La p corresponde a los posibles átomos proposicionales. Las fórmulas $not\ \varphi$, $\varphi\ and\ \varphi$, $\varphi\ or\ \varphi$, $\varphi\ =>\ \varphi$, $\varphi\ <=>\ \varphi$ corresponden a las de lógica proposicional. La fórmula $k(a, \varphi)$ representa $K_a(\varphi)$ con $a \in Agentes$. La fórmula $e(A, \varphi)$ representa $E_A(\varphi)$, con $A \subseteq Agentes$. La fórmula $c(A, \varphi)$ representa $C_A(\varphi)$. Finalmente la fórmula $d(A, \varphi)$ representa $D_A(\varphi)$ con $a \in Agentes$.

La representación del etiquetamiento de mundos consiste únicamente en los estados de memoria vistos como un arreglo unidimensional. Debido a lo anterior tenemos distintas representaciones de átomos proposicionales. El átomo $p(Index)$ se evalúa a verdadero si y solo si la posición *Index*-ésima del arreglo tiene valor 1. El átomo $p(Index, Value)$ evalúa a verdadero si y solo si la posición *Index*-ésima del arreglo tiene valor *Value*. El átomo $p(NAgents, Round, CAgent, Value)$ se evalúa a verdadero siempre que en un protocolo en el que participan *NAgents* agentes, el agente que escribe en la columna *CAgent* de la memoria, escribe el valor *Value* al ejecutar la iteración *Round* del protocolo.[2]

Consideremos el modelo de Kripke $\mathcal{M}$ de la figura A.2 y supongamos que la variable `Model` está unificada con su representación. Podemos hacer, entre otras, las siguientes preguntas:

Para evaluar $\mathcal{M}, 8 \models K_b(p_{1,b,1})$, preguntamos `valid(k(b,p(2,1,1,1)),8,Model)`. Como es de esperar el intérprete evalúa a *true*.

[2]Realmente se evalúa $p(Index, Value)$ con $Index = NAgents \times Round + CAgent$.

```
?- sample_model(Model),valid(k(b,p(2,1,1,1)),8,Model),!.
Model = ...
```

Para evaluar $\mathcal{M}, 8 \models K_a(p_{1,b,1})$, preguntamos `valid(k(a,p(2,1,1,1)),8,Model)`. Como el mundo 8 y el mundo 11 son indistinguibles por el agente a, y $p_{1,b,1}$ no está en el etiquetamiento del mundo 11, el intérprete evalúa a $false$.

```
?- sample_model(Model),valid(k(a,p(2,1,1,1)),8,Model),!.
false.
```

Para evaluar $\mathcal{M}, 8 \models C_{\{a,b\}}(p_{1,b,1})$, preguntamos `valid(c([a,b],p(2,1,1,1)),8,Model)`. Como $\mathcal{M}, 8 \models \neg K_a(p_{1,b,1})$, el intérprete evalúa a $false$.

```
?- sample_model(Model),valid(c([a,b],p(2,1,1,1)),8,Model),!.
false.
```

Para evaluar $\mathcal{M}, 8 \models \neg C_{\{a,b\}}(p_{1,b,1})$, preguntamos `valid(not c([a,b],p(2,1,1,1)),8,Model)`. Como $\mathcal{M}, 8 \not\models C_{\{a,b\}}(p_{1,b,1})$, el intérprete evalúa a $true$.

```
?- sample_model(Model),valid(not c([a,b],p(2,1,1,1)),8,Model),!.
Model = ...
```

Como último ejemplo, para evaluar $\mathcal{M}, 8 \models D_{\{a,b\}}(p_{1,b,1})$ preguntamos `valid(d([a,b],p(2,1,1,1)),8,Model)`. Recordando que en los modelos de Kripke que tratamos el etiquetamiento de los mundos es el conocimiento distribuido de los agentes correspondientes a los procesos del complejo simplicial, la consulta se evalúa a $true$.

```
?- sample_model(Model),valid(d([a,b],p(2,1,1,1)),8,Model),!.
Model = ...
```

A.7. Predicados generadores de gráficas (Graphviz)

Finalmente mostraremos el uso de los predicados con los que generamos las gráficas que representan complejos simpliciales, modelos de Kripke y modelos de acciones extendidos que implementamos en este trabajo. Los predicados que implementamos generan código en lenguaje DOT, mismos que se transforman usando los diseños *circo*, *dot*, *fdp*, *neato*, *osage*, *patchwork*, *sfdp* o *twopi* del paquete en Graphviz.

En este trabajo encontramos que el diseño sfdp genera una mejor representación de nuestros modelos por lo que usamos dicho diseño para generar todas las figuras del trabajo.

A.7.1. Complejos simpliciales

Para generar el código DOT correspondiente a un complejo simplicial generamos la representación gráfica para las facetas del complejo. Es importante señalar que Graphviz representa gráficas y por tal motivo los simplejos de dimensión mayor a 1 no se muestran como superficies. De esta forma, las imágenes producidas para complejos simpliciales de dimensiones superiores pueden no ser lo más adecuadas.

El predicado para representar complejos simpliciales gráficamente es `generate_complex(Complex,Columns,Colors,Program,DFile,PFile)`. La variable `Complex` es la representación del complejo simplicial. La variable `NColumns` es el número de columnas en las que se pintará la vista. La variable `Colors` es una lista de elementos de la forma (*proceso* => *color*) que relaciona un color distinto[3] a cada proceso del complejo simplicial. Por ejemplo, para generar la imagen de la figura A.1 teniendo la variable `Complex` unificada con la representación mostrada en la subsección A.1.1, llamaríamos al predicado como sigue:

```
generate_complex(
    Complex,
    3,
    [a=>gray60,b=>gray80,c=>gray100],
```

[3]Los colores válidos se pueden consultar aquí: `https://www.graphviz.org/doc/info/colors.html`.

```
    sfdp,
    'complex.dot',
    'complex.png'
).
```

A.7.2. Modelos de Kripke

A diferencia de los complejos simpliciales, los modelos de Kripke son gráficas, de tal forma que Graphviz es adecuado para representar modelos de Kripke sobre cualquier número de agentes.

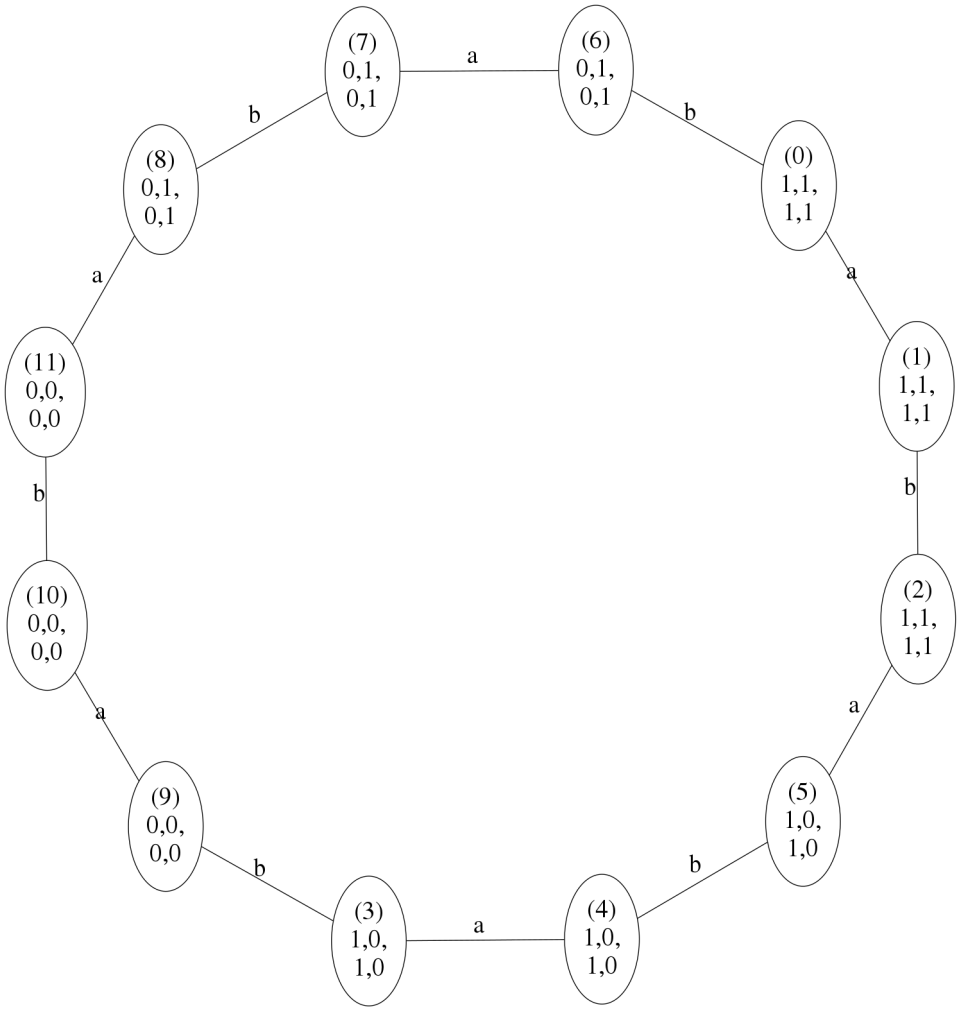

Figura A.8: Modelo epistémico correspondiente al de la figura A.2.

El predicado para generar las imágenes de modelos de Kripke es `generate_model(Model,Columns,Program,DotFile,PngFile)`. La variable `Model` debe estar unificada con una representación del modelo de Kripke de la forma descrita en la subsección A.1.2. La variable `Columns` será el número de columnas en las que se pintará el estado de memoria correspondiente al etiquetamiento. La variable `Program` es el diseño de Graphviz con el que se generará la gráfica. La variable `DotFile` es el archivo DOT a generar. Por último, la variable `PngFile` es el nombre del archivo png a generar. El predicado genera una gráfica no dirigida ya que en los modelos que estudiamos las

relaciones de accesibilidad son siempre simétricas. Por ejemplo, para generar la imagen de la figura A.2 teniendo la variable `Model` unificada con la representación mostrada en la subsección A.1.2, llamaríamos al predicado como sigue:

```
generate_model(Model,3,sfdp,'model.dot','model.png').
```

Recordemos que las relaciones de accesibilidad de los modelos de Kripke correspondientes a conjuntos de simplejos son siempre simétricas y transitivas. Además, cuando los modelos son epistémicos, sus relaciones de accesibilidad cumplen también con ser reflexivas. Usualmente al tratar con modelos epistémicos, la representación elimina la reflexividad y la transitividad de las relaciones para que el modelo no se sature visualmente. Para generar las imágenes correspondientes a modelos de Kripke se implementó el predicado `generate_simplified_model(Model,Columns,Program,DotFile,PngFile)` con los mismos parámetros de `generate_model`. Por ejemplo, para generar la imagen de la figura A.8, teniendo la variable `Model` unificada con la representación mostrada en la subsección A.1.2 llamaríamos al predicado como sigue:

```
generate_simplified_model(
    Model,
    3,
    sfdp,
    'model.dot',
    'model.png'
).
```

A.7.3. Modelos de acciones extendidos

Recordemos que los modelos de acciones extendidos son estructuras del tipo $\mathbf{M} = \langle S, \sim, pre, L\rangle$, donde S es el conjunto de puntos de acciones, $\sim$ es la relación de indistinguibilidad de agentes sobre puntos de acción, pre es la función de precondición y L es la función de etiquetamiento que relaciona a cada punto de acción la nueva información que se tendrá después de aplicar la acción correspondiente.

Para la representación gráfica de los modelos de acciones extendidos dejamos de lado la función de precondición pre, pues la fórmula epistémica asociada a cada punto de acción no tiene límite en tamaño, lo que podría saturar la imagen generada. Además, consideramos de mayor importancia

la nueva información que los modelos de acciones son capaces de aportar al nuevo modelo de Kripke.

Para generar los modelos de acciones extendidos, tenemos el predicado `generate_extended_action_model(AModel,Cols,Prog,DotFile,PngFile)`. La variable `AModel` es una representación del modelo de acciones extendido como se describe en la subsección A.1.3. La variable `Cols` representa las columnas en las que se pintan las nuevas proposiciones del etiquetamiento. La variable `Prog` es el diseño de Graphviz a utilizar. La variable `DotFile` es el archivo auxiliar en lenguaje DOT. Por último, la variable `PngFile` es el archivo en el que se guardará la imagen generada. Por ejemplo, para generar la imagen de la figura A.3 con la variable `ActionModel` unificada con la representación del modelo de acciones extendido mostrada en la subsección A.1.3 llamamos al predicado como sigue:

```
generate_extended_action_model(
    ActionModel,
    2,
    sfdp,
    'mae.dot',
    'mae.png'
).
```

Bibliografía

[1] Alexandru Baltag, Lawrence S. Moss, and Slawomir Solecki. The logic of public announcements, common knowledge, and private suspicions. In *Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge*, TARK '98, page 43–56, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[2] Elizabeth Borowsky and Eli Gafni. A simple algorithmically reasoned characterization of wait-free computation (extended abstract). In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, PODC '97, page 189–198, New York, NY, USA, 1997. ACM.

[3] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*. Springer Publishing Company, Incorporated, 1st edition, 2007.

[4] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning About Knowledge*. MIT Press, Cambridge, MA, USA, 2003.

[5] E. Goubault and S. Rajsbaum. A simplicial complex model of dynamic epistemic logic for fault-tolerant distributed computing. *ArXiv e-prints*, March 2017.

[6] Éric Goubault, Marijana Lazić, Jérémy Ledent, and Sergio Rajsbaum. A dynamic epistemic logic analysis of the equality negation task. Proc. of the 2nd Workshop on Dynamic Logic: New Trends and Applications (DaLí), October 2019.

[7] Éric Goubault, Marijana Lazić, Jérémy Ledent, and Sergio Rajsbaum. Wait-free solvability of equality negation tasks. Proc. 33rd International Symposium on Distributed Computing (DISC), October 2019.

[8] Éric Goubault, Jérémy Ledent, and Sergio Rajsbaum. A simplicial complex model for dynamic epistemic logic to study distributed task computability. In Andrea Orlandini and Martin Zimmermann, editors, Proceedings Ninth International Symposium on *Games, Automata, Logics, and Formal Verification,* Saarbrücken, Germany, 26-28th September 2018, volume 277 of *Electronic Proceedings in Theoretical Computer Science*, page 73–87. Open Publishing Association, 2018.

[9] Maurice Herlihy, Dmitry Kozlov, and Sergio Rajsbaum. *Distributed Computing Through Combinatorial Topology.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2013.

[10] Michael Huth and Mark Ryan. *Logic in Computer Science: Modelling and Reasoning About Systems.* Cambridge University Press, New York, NY, USA, 2004.

[11] Wai-kau Lo and Vassos Hadzilacos. All of us are smarter than any of us: Nondeterministic wait-free hierarchies are not robust. *SIAM Journal on Computing*, 30, 07 2001.

[12] Davide Sangiorgi. *Introduction to Bisimulation and Coinduction.* Cambridge University Press, 2012.

[13] Davide Sangiorgi and Jan Rutten. *Advanced Topics in Bisimulation and Coinduction.* Cambridge University Press, New York, NY, USA, 1st edition, 2011.