Semánticas de procesos para sistemas interactivos y sistemas probabilísticos

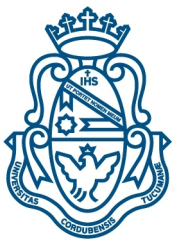
Matias David Lee

Director: Pedro R. D'Argenio

Presentada ante la Facultad de Matemática, Astronomía y Física como parte de los requerimientos para la obtención del grado de Doctor en Ciencias de la Computación de la

Universidad Nacional de Córdoba

Marzo de 2013



Clasificación: D.3.1 Formal Definitions and Theory (Semantics) - F.3.2 Semantics of Programming Languages (Operational semantics)

Palabras claves: semántica de procesos, sistemas de transiciones, acciones de entrada y salida, no interferencia, semántica operacional estructurada, formatos de regla, bisimulación, congruencia.

Resumen

En esta tesis estudiamos semánticas para sistemas interactivos, i.e. sistemas donde la ejecución de una acción está controlada por el mismo sistema o por un agente externo. El enfoque utilizado se basa en tipos de observación y nociones de observabilidad. Utilizando una variante de sistema interactivos se estudia la propiedad de seguridad denominada no-interferencia. Si un sistema satisface esta propiedad, entonces éste realiza una manipulación segura de la información Además estudiamos el problema de definir lenguajes para especificar sistemas de transiciones probabilistas. El enfoque utilizado se basa en semántica operacional estructurada (SOS). En este contexto presentamos el formato $nt\mu f\theta/nt\mu x\theta$ y estudiamos distintas propiedades del mismo.

Índice general

Agradecimientos

Hace unos días empecé a leer un libro: *El Manantial* de Ayn Rand. Hace casi dos años, o tal vez más, que el libro estaba en la oficina. Parece que es un clásico. La edición que encontré tiene la introducción a la edición del 25 aniversario. Ésta está escrita por la autora del libro. En el primera linea ella cuenta que mucha gente le ha preguntado que siente acerca de que hayan pasado 25 años de la primera edición del libro. En la segunda linea ella escribe:

"No puedo decir que sienta nada particular,
excepto una suerte de tranquila satisfacción."

Creo que la frase describe bien como me siento con respecto a mi doctorado. Pero siento la satisfacción no por el contenido dentro de estas páginas, sino por el recorrido en estos 6 años. Fueron años de mucho crecimiento y mucho aprendizaje (principalmente en todo lo que no tiene que ver con ciencia).

Pero sé que este logro no es sólo mérito mío, por esta razón quiero empezar agradeciendo a las instituciones que hicieron esto posible y a la gente que en ellas colaboran (sin un poco de estructura nada de esto hubiese sido posible): FaMAF, SECyT y CONICET, o en otras palabras, al Estado Argentino. También a toda la gente que me ha acompañado de una u otra forma. Son muchas personas... tengo la suerte de cruzarme siempre con mucha buena gente. A todos ellos, muchas gracias, siempre estarán conmigo.

Martes 16 de abril de 2013,
en un día de sol, en la oficina 399.
F@MAF, Córdoba, Argentina.

Capítulo 1

Introducción

"Todo lo que dice este libro
puede ser una gran mentira" [1]

Debido a los avances tecnológicos, los sistemas informáticos se han convertido en parte de nuestro día a día. Estos poseen una gran capacidad de procesamiento y almacenamiento en un espacio físico mínimo. Además, estos sistemas se interrelacionan entre ellos para construir nuevos sistemas mucho más complejos. No sólo esto, las posibilidades de comunicación con el entorno también son muy variadas debido a que los periféricos de entrada y salida también han evolucionado. Estas nuevas formas de comunicación permiten que un sistema informático pueda interactuar de *forma bidireccional* y *continua* con personas, con otros sistemas informáticos y hasta con otros sistemas no informáticos. Bidireccional porque el entorno actúa sobre el sistema y viceversa. Continua porque la interacción se mantiene en el tiempo, lo cual posibilita un *dialogo* entre sistema y entorno.

Ante este gran espectro de posibilidades, la finalidad de los sistemas informáticos es increíblemente variada. Ésta puede ser la recreación, el realizar cálculos complejos, el simular eventos, el facilitar y optimizar la realización de procesos de información, el controlar y garantizar la seguridad de otros sistemas, etc. Es más, muchas veces los sistemas informático manejan informaciones sensibles que no pueden ser reveladas o realizan tareas en las cuales una falla puede significar la pérdida de vidas humanas. Debido a la complejización y a las tareas sensibles que puede realizar un sistema informático, es de suma importancia contar con herramientas formales para describir y estudiar a los mismos.

Los resultados que presentaremos en esta tesis se enmarcan dentro de la rama de la ciencia de la computación que estudia a los *sistemas concurrentes*. En este tipo particular de sistemas los

[1] La "cita" es lo que yo recordoba de la cita "Todo lo que dice este libro puede ser una falacia" de Richad Bach en el libro Ilusiones.

objetos centrales de estudio son los *procesos*. Los procesos suelen ser sistemas (informáticos o no) que interactúan con otros sistemas (informáticos o no) para llevar a cabo una tarea o función. Los ejemplos de estos sistemas son muy variados: servicios webs, sistemas multiprocesadores, sistemas distribuidos, sistemas robóticos, sistemas operativos y hasta sistemas biológicos.

El concepto clave en este contexto es el de *interacción*. La presencia de interacción no permite modelar a los sistemas concurrentes mediante algoritmos o funciones, como en el caso de los programas secuenciales. En otras palabras, el funcionamiento del sistema como un todo no puede representarse como sólo una transformación de estados. Tomemos por ejemplo los siguientes dos fragmentos de programas [70]:

$$X := 2 \qquad \text{y} \qquad X := 1;\ X := X + 1$$

En un contexto secuencial, para cualquier estado inicial, ambos programas producen el mismo resultado, i.e. $X = 2$. Entonces si representamos a los programas como funciones de estados en estados ambos programas podrían considerarse iguales. Pero en un contexto concurrente esta igualdad no se comporta como uno esperaría que se comporte. Supongamos un lenguaje para describir sistemas concurrentes, el cual permite la ejecución en paralelo de dos programas mediante el operador $\|$. Supongamos que se denota con $S \parallel T$ la ejecución en paralelo de los programas S y T, i.e. S y T se ejecutan concurrentemente sobre una memoria común y las instrucciones de cada programa se ejecuta de forma atómica. Estudiemos ahora cómo se comporta la paralelización de los dos fragmentos de programas con el programa $X := 2$. Si se utiliza el primer programa se obtiene

$$X := 2 \parallel X := 2$$

Es este caso, el programa resultante siempre termina en el estado $X = 2$ para todo estado inicial. Por otro lado si se utiliza el segundo programa se obtiene

$$X := 1;\ X := X + 1 \parallel X := 2$$

Notemos que si bien se realizó una substitución con un programa que es "igual" al primero, el programa que se obtiene es distinto al que se obtuvo anteriormente. Este programa no termina siempre en $X = 2$. Si se ejecuta la instrucción $X := 2$ y luego la instrucción $X := X + 1$ se obtiene el estado final $X = 3$.

El ejemplo muestra que la igualdad que se definió no es preservada por la paralelización: el operador de paralelización para argumentos "iguales" puede resultar en sistemas que no son iguales. Formalmente se dice que la semántica no es *composicional*, o que la igualdad de programas no es una *congruencia*.

Una semántica que no es composicional no es deseable porque no permite definir la semántica de un sistema compuesto en base de los sistemas que lo componen. Esto tiene como consecuencia no poder inferir propiedades del sistema total en función de sus componentes o no poder reemplazar una parte de éste por otra igual, dado que el comportamiento del sistema total podría cambiar.

En el ejemplo presentado se pueden apreciar dos problemas que estudiaremos en esta tesis:

1. Cómo definir la igualdad entre los sistemas concurrentes.

2. Cómo se comporta la igualdad con respecto a las operaciones qué se utilizan para describir los sistemas concurrentes.

Por supuesto, estas respuestas dependerán del modelo que se utilice para representar a los sistemas de nuestro interés.

El enfoque para modelar sistemas concurrentes depende muchas veces del componente físico del sistema. Por ejemplo, el modelo PRAM [51] (*parallel random-access machine*) es utilizado para estudiar la complejidad de algoritmos en sistemas con procesadores paralelos, los cuales pueden acceder aleatoriamente en una unidad de tiempo a cualquier celda de una memoria global. En este caso la interacción entre procesos se realiza a través de la memoria compartida. Esta misma forma de interacción se utiliza en el caso de los programas presentados anteriormente. En nuestro caso, estudiaremos sistemas donde la interacción se realiza directamente entre procesos. Esta interacción se realiza a través de "mensajes". Estos mensajes se modelarán utilizando *acciones* con el enfoque denominado *concurrencia uniforme* [9], i.e. no será de nuestro interés la estructura interna de la acción. Por lo tanto, estas acciones pueden representar distintos tipos de eventos: "se presionó el botón X", "se envía un pedido a una base de dato", "se produjo un timeout", etc. Para especificar el orden en que las acciones se realizan utilizamos *sistemas de transiciones*: grafos dirigidos donde las transiciones están etiquetadas con acciones. Este marco se adapta bien para distintos contextos, por ejemplo para describir y verificar protocolos de comunicación [74] o algoritmos de exclusión mutua [77]. Pero en algunas ocasiones es necesario realizar extensiones sobre el modelo para estudiar particularidades del sistema de interés. En esta tesis trabajaremos con dos extensiones distintas: los *sistemas interactivos* y los *sistemas con transiciones probabilistas*.

En los *sistemas interactivos* [78] las acciones (visibles) se dividen en acciones de entrada y de salida. Las acciones de entrada modelan las acciones que el sistema espera que el entorno ejecute para reaccionar de una forma particular. Las acciones de salida son las acciones que ejecuta el sistema y producen un efecto en el entorno. Luego de la ejecución de una acción, no importa cual sea su tipo, siempre transiciona a un estado particular. La diferenciación entre acciones de entrada y acciones de salida se realiza para establecer puntualmente quién controla cada acción. Las acciones de entrada son controladas por el entorno; las acciones de salida son controladas por el sistema. O desde el punto de vista del entorno, las acciones de entrada son controlables mientras que las de salida no. Existen muchos ejemplos de este tipo de sistemas, estos van desde una máquina expendedora de bebidas a un sistema complejo de control que reacciona ante distintos estímulos del ambiente. Por otro lado, en los *sistemas con transiciones probabilistas* luego de ejecutarse una acción se elige el siguiente estado de acuerdo una distribución de probabilidad. Estos sistemas permiten representar sistemas con más detalles ya que permiten modelar acciones del tipo "*se envía un mensaje y éste llega a destino con una probabilidad 0.9 y se pierde con una probabilidad de 0.1*".

1.1. Contribuciones

Esta tesis está dedicada al estudio de los siguientes temas:

- En el contexto de programas interactivos se estudia el problema de definir cuando dos

sistemas interactivos son iguales.

- En este mismo contexto, se estudia una variante de sistema interactivo que manipula información confidencial. Se define cuando uno de estos modelos realiza una manipulación segura de información y se estudian distintos aspectos relacionados con la propiedad de seguridad.
- En el contexto de sistemas de transiciones probabilista se estudia cómo definir operadores para un lenguaje de especificación tal que la bisimulación sea composicional y se estudian distintos aspectos del método desarrollado.

Versiones preliminares de los resultados presentados en esta tesis se publicaron en los siguientes trabajos:

- *"Describing Secure Interfaces with Interface Automata"* [56]. Matias David Lee, Pedro R. D'Argenio. 2010.
- *"A Refinement Based Notion of Non-interference for Interface Automata: Compositionality, Decidability and Synthesis"* [57]. Matias David Lee, Pedro R. D'Argenio. 2010.
- *"Semantics for Interactive Sequential Systems and Non-Interference Properties"* [58]. Matias David Lee, Pedro R. D'Argenio. 2011.
- *"Probabilistic Transition System Specification: Congruence and Full Abstraction of Bisimulation"* [22]. Pedro R. D'Argenio, Matias David Lee. 2012.
- *"Tree rules in probabilistic transition system specifications with negative and quantitative premises"* [59]. Matias David Lee, Daniel Gebler, Pedro R. D'Argenio. 2012.

1.2. Esquema de la tesis

A continuación describimos brevemente los contenido de cada capítulo de esta tesis:

Capítulo 2: Se introducen los sistemas de transiciones con acciones de entrada y salida y se definen los supuestos sobre este modelo particular. Se discute por qué las semánticas definidas para los sistemas de transiciones que no diferencian entre acciones de entrada y de salida no se ajustan bien para este modelo. En función de los supuestos se definen un conjunto de posibles semánticas para sistemas interactivos. Cada semántica es representada por una *noción de observabilidad*.

Capítulo 3: El conjunto de acciones de entrada y de salida es dividido en dos nuevos tipos de acciones: confidenciales y no confidenciales. Sobre esta variación del modelo original se definen dos variantes de la propiedad de seguridad denominada *no interferencia*. Ambas variantes toman como parámetro una noción de observabilidad. Se estudia distintos aspectos de las propiedades

de seguridad. Finalmente se presentan dos algoritmos: el primero chequea si un sistema satisface la propiedad; si la respuesta es negativa, el segundo síntetiza un sistema seguro (en caso de ser posible y teniendo en cuenta algunas resticciones).

Capítulo 4: Se presenta un enfoque algebraico para especificar sistemas de transiciones probabilistas y se presenta una técnica para derivar sistemas de transiciones que se adecuen a una especificación particular. Las acciones de entrada y salida son dejada de lado pues las etiquetas pasan a ser un objeto meramente sintáctico.

Capítulo 5: Se presenta el formato $nt\mu f\theta/nt\mu x\theta$ para especificar sistemas de transiciones probabilistas. Se demuestra que la bisimulación para este tipo de sistemas es una congruencia para todo operador cuya semántica se define mediante reglas *bien fundadas* (Def. 5.3.1) en formato $nt\mu f\theta/nt\mu x\theta$.

Capítulo 6: En este capítulo se demuestra que toda especificación en formato $nt\mu f\theta/nt\mu x\theta$ puede reducirse a una especificación equivalente la cual se encuentra en formato *pntree*, una versión restringida del formato original, la cual sólo posee reglas bien fundadas. Como consecuencia resulta que la bisimulación es una congruencia para toda especificación en formato $nt\mu f\theta/nt\mu x\theta$.

Capítulo 7: Aquí se estudia el poder de los operadores que se pueden definir con el formato $nt\mu f\theta/nt\mu x\theta$. Se demuestra que la congruencia inducida por contextos del formato $nt\mu f\theta/nt\mu x\theta$, con respecto a la equivalencia de trazas, es exactamente la bisimulación.

Capítulo 8: Este capítulo presenta las conclusiones de la tesis y posibles lineas de continuación para el trabajo desarrollado.

Capítulo 2

Semánticas de procesos interactivos

2.1. Sistemas interactivos

El término *proceso interactivo* está asociado a un proceso que coopera con otras entidades para lograr un objetivo. Estas entidades pueden ser de diferentes tipos: otros procesos, el usuario de un sistema, o una celda de memoria. Para estudiar cómo es esta interacción se representan los procesos mediante abstracciones. En estas abstracciones, la forma de interactuar, en su forma más simple, está representada a través de acciones. Por ejemplo, en la Figura 2.1 vemos una representación gráfica de una máquina expendedora de bebidas [70]. En esta representación podemos ver que la máquina puede estar en distintos estados ($\{q_0, q_1, q_2, q_3\}$) y que en cada estado puede realizar distintas acciones ({*moneda?, café?, té?, servirCafé!,servirTé!*}). Notemos que el comportamiento que modela el gráfico es bastante intuitivo: la máquina si se encuentra en el estado q_0 espera que el usuario inserte una moneda, luego que elija entre té o café y en función de eso servir la bebida correspondiente.

Observemos que sólo se describen los aspectos que definen cómo es la interacción entre la máquina y su entorno, todo aspecto no relacionado a esto es dejado de lado. Por ejemplo, no se especifica características físicas de la máquina expendedora, tampoco la forma ni el valor de la moneda, etc. Entre los aspectos que hacen a la interacción podemos ver en este ejemplo que existen dos tipos de acciones: las acciones seguidas por un símbolo "?" y las acciones seguidas por un símbolo "!". Las primeras son denominadas *acciones de entrada* y son las acciones que son ejecutadas por el *ambiente*, i.e. las entidades que interactúan con el sistema, y producen un estímulos sobre el mismo. Este estímulo está representado por el cambio de estado al realizar la transición. Las segundas son denominadas *acciones de salida* y son las acciones que ejecuta el sistema y producen estímulos sobre el ambiente. En este ejemplo no se detalla, pero existe un tercer tipo de acciones, *las ocultas o internas*, éstas se utilizan para representar acciones internas del sistema y se indican agregándoles el símbolo ";". Los tres tipos de acciones a su vez se clasifican en *observables* o *no observables*. El primer grupo se compone por las acciones de entrada y de salida mientras que el segundo por las acciones internas. Esta división se debe a que existe el supuesto de que todo observador del sistema puede observar cualquier acción que no sea

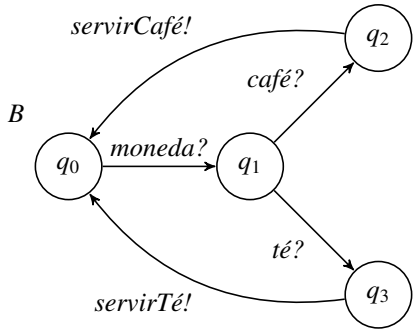


Figura 2.1.: Una máquina expendedora de bebidas

interna. Existe una última diferenciación sobre las acciones: *las acciones controlables* y las *no controlables*. Notemos que un proceso no tiene injerencia sobre las acciones de entrada. Es decir que si un proceso está en un estado en el cual sólo se transiciona mediante acciones de entrada, el sistema no progresará hasta que el ambiente realice un estímulo habilitado. Por ejemplo, la máquina expendedora nunca habilitará las opciones de seleccionar café o té si no se inserta primero una moneda. Por otro lado, el ambiente no tiene injerencia sobre las acciones de salida. En el ejemplo, luego de seleccionar café como bebida el usuario no podrá evitar que la máquina empiece a servirlo. Entonces, con respecto al usuario, las acciones *controlables* son las acciones de entrada, mientras que las *no controlables* son las acciones de salida. Las acciones internas también se clasifican como *no controlables* dado que dependen exclusivamente del proceso.

Este tipo de modelos puede representarse formalmente mediante sistemas interactivos de transiciones etiquetadas las cuales definimos a continuación y denominaremos *Autómatas de Interfaces* [25, 26].

Definición 2.1.1 (IA)**.** *Un* Autómata de Interfaz *S (*IA*, del inglés* Interface Automata*) es una 4-upla* $\langle Q, q_0, A, \rightarrow \rangle$ *tal que* Q *es un conjunto de* estados *(o procesos),* $q_0 \in Q$ *es el* estado inicial*,* $A = A^I \cup A^O \cup A^H$ *es un conjunto de* acciones *tales que* A^I *(resp.* A^O, A^H*) es el conjunto de* acciones de entrada *(resp.* de salida, ocultas o internas*), tal que los conjuntos son disjuntos y* $\rightarrow \subseteq Q \times A \times Q$ *es una* relación de transición (determinística en entradas) *i.e. si* $(q, a, q'), (q, a, q'') \in \rightarrow, a \in A^I$ *entonces* $q' = q''$.

Una parte esencial de la Teoría de Procesos es definir cuando dos procesos deben ser considerados iguales, dado que procesos diferentes podrían exhibir el mismo comportamiento. Una *semántica* define formalmente el comportamiento observable de un proceso. Definido esto, podremos decir que dos procesos son equivalentes si presentan el mismo comportamiento observable.

En [37, 38] se presenta un estudio extenso sobre diferentes semánticas para procesos sin tener y teniendo en cuenta la posibilidad de que los procesos realicen transiciones internas. Sin embargo, en [58] argumentamos que estas semánticas no son acordes para el tipo de sistemas que queremos abordar, i.e. sistemas interactivos, pues estos presentan características distintas. Para sustentar esta afirmación primero estableceremos las suposiciones que rigen a estos modelos.

Luego se presentará una segunda máquina expendedora de bebidas. Esta máquina, teniendo en cuenta las suposiciones establecidas, presentará claramente un comportamiento distinto al de primera máquina presentada. Compararemos ambos modelos con semánticas para sistemas de transiciones donde no se realiza la discriminación entre acciones de entrada y acciones de salida. En esta comparación podremos observar que estas semánticas no encajan bien para el contexto de IA.

Suposición 1. *Las siguientes son las suposiciones sobre la interfaz:*

(sup1) Acciones generativas. *El proceso bajo estudio es el que genera las acciones de salida y las internas. Es el único que las puede controlar y realizar de manera autónoma. El entorno sólo puede observarlas, pero nunca evitar su ejecución.*

(sup2) Entradas reactivas. *El proceso bajo estudio no tiene control de las acciones de entrada y no puede ejecutarlas de manera autónoma. Sólo podrá avanzar con una acción de entrada si dicha entrada es provista por el entorno. Es decir que el proceso sólo puede ejecutar una entrada como reacción al entorno que la provee.*

(sup3) Acciones instantáneas. *Consideraremos que la ejecución de cada acción es instantánea.*

(sup4) Reposo. *Entre la ejecución de una acción y la siguiente puede transcurrir un cierto tiempo (no especificado por el modelo). Es decir, el proceso bajo estudio tiene la capacidad de reposar en cualquier estado por una cantidad de tiempo no especifico.*

(sup5) Weak fairness. *Si un proceso puede ejecutar una acción de salida o interna en un estado, entonces no reposará indefinidamente y ejecutará una de ellas al cabo de un cierto tiempo.*

(sup6) Ausencia de (la paradoja de) Zenon. *En un lapso acotado de tiempo sólo se puede ejecutar una cantidad finita de acciones.*

Las condiciones (sup1) y (sup2) establecen quién controla cada tipo de acción. La condición (sup3) garantiza que no se podrán ejecutar dos acciones al mismo tiempo; además, que todo sistema se encontrará en un estado particular al recibir una acción de entrada y no en medio de la ejecución de otra acción. Esto último es necesario pues las acciones de entrada no son controladas por el sistema. Esta suposición es estándar en este tipo de modelos [37]. La condición (sup4) establece que el usuario siempre tiene tiempo suficiente para enviar uno o más estímulo en cada estado. La condición (sup5) garantiza que el sistema siempre progresa. Finalmente la condición (sup6) indica que ese progreso consume una cantidad de tiempo. Las últimas tres condiciones serán necesarias para registros que necesitan modelar nociones relacionadas con el tiempo de una ejecución.

En la Figura 2.2 se modela una IA B', la cual representa una segunda máquina expendedora de bebidas. La misma tiene un comportamiento errático, pues luego de que un usuario inserte una moneda, por momentos ofrecerá sólo café y por momentos ofrecerá sólo té. Teniendo en cuenta las suposiciones recién realizadas se puede inferir que la interfaz descripta por B (Figura 2.1) es distinta a la interfaz descripta por B'. En B, luego de insertar una moneda, un usuario

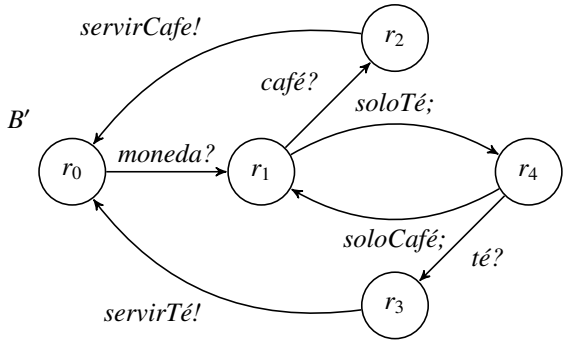


Figura 2.2.: Una expendedora de bebida indecisa

podrá siempre elegir entre café o té; una vez realizada la elección, la máquina servirá la bebida elegida. Lo mismo no se puede asegurar en B': un usuario luego de insertar una moneda podrá elegir café o té, pues éstas son acciones que él controla, pero no necesariamente la interfaz responderá a ese estímulo. Esto dependerá si la interfaz se encuentra en el estado r_1 o r_4.

A continuación definimos los sistemas de transiciones que no diferencian entre acciones de entrada y de salida y algunas semánticas para este tipo de modelo.

Definición 2.1.2 (LTS)**.** *Un* sistema de transiciones etiquetadas *(*LTS*, del inglés* Labeled Transition System*) es una 4-upla* $\langle Q, q_0, A, \rightarrow \rangle$ *tal que* Q *es un conjunto de* estados (o procesos)*,* $q_0 \in Q$ *es el* estados inicial*,* $A = A^V \cup A^H$ *es un conjunto de* acciones *tales que* A^V *(resp.* A^H*) es el conjunto de* acciones visibles u observables *(resp.* ocultas o internas*), los conjuntos son disjuntos y* $\rightarrow \subseteq Q \times A \times Q$ *es una* relación de transición*.*

Las semánticas con las que trabajaremos son: *trace semantic*, *ready trace semantic* y *bisimulación débil*. Las definiciones que presentamos están basadas en el trabajo desarrollado en [37].

- Trace semantic modela los posible comportamiento lineales observables del sistema (trazas).
- Ready trace semantic es una semántica de trazas decoradas: "semántica de trazas" porque se modelan comportamientos lineales; "decoradas" porque además es posible saber un poco más sobre la estructura del sistema. En este caso, en cada estado estable (estados que no pueden realizar transiciones internas) será posible conocer que acciones pueden ejecutarse en ese estado.
- Weak bisimulation es una relación más fuerte pues ésta considera que dos sistemas son equivalentes cuando toda transición que puede realizar un sistema puede ser imitada por el otro (quizás ejecutando cierta actividad interna) y viceversa. Esta propiedad debe ser mantenida a medida que se van ejecutando las transiciones en ambos procesos.

Para representar las primeras dos semánticas utilizaremos *trazas*. Una traza será el registro de una secuencia de hechos. En el caso de trace sematic, los hechos que se podrán registrar serán las ejecuciones de acciones visibles. En el caso de ready trace semantic los hechos serán las

ejecuciones de acciones visibles o conjuntos de acciones visibles, los cuales representarán las posibles acciones que puede ejecutar un estado (estable).

En cambio, para representar bisimulación débil utilizaremos una caracterización relacional. En este tipo de caracterización se define una relación entre dos procesos, tal que las restricciones impuesta por la misma garantiza que las observaciones que cada proceso puede realizar son las mismas.

En la definición de las semánticas utilizaremos la siguiente notación. Denotamos con $q \xrightarrow{a} q'$ a la tupla $(q, a, q') \in \rightarrow$, mientras que con $q \xrightarrow{a}$ se denota la existencia de q' tal que $q \xrightarrow{a} q'$, y con $q \not\xrightarrow{a}$ el caso contrario. Con $q \xrightarrow{\tau} q'$ se denota la existencia de $a \in A^H$ tal que $q \xrightarrow{a} q'$. Con $q \not\xrightarrow{\tau}$ denotamos $q \not\xrightarrow{a}$ para todo $a \in A^H$, y en este caso diremos que q es un *estado estable*. Una *ejecución* desde q_0 es una secuencia finita $q_0\, a_0\, q_1\, a_1 \ldots q_n$ tal que $q_i \in Q$, $a_i \in A$ y $q_i \xrightarrow{a_i} q_{i+1}$ para $0 \leq i < n$. Si existe una ejecución con todas las acciones internas tal que desde q se alcanza q', denotamos esto con $q \Rightarrow q'$. Esto incluye el caso $q = q'$. Con $q \overset{a}{\Rightarrow} q'$ denotamos que existen $q_1, q_2 \in Q$ tales que $q \Rightarrow q_1 \xrightarrow{a} q_2 \Rightarrow q'$. Además, $q \overset{\hat{a}}{\Rightarrow} q'$ denota $q \overset{a}{\Rightarrow} q'$ o $q \Rightarrow q'$ si $a \in A^H$. Una traza de q es una secuencia de acciones visibles $a_0, \ldots, a_n \in A^V$ tal que $q \overset{a_0}{\Rightarrow} q_0 \overset{a_1}{\Rightarrow} q_1 \ldots \overset{a_n}{\Rightarrow} q_n$ para algunos estados $q_0, \ldots, q_n$. Con $A(q)$ denotaremos el conjunto de acciones observables que puede ejecutar un estado q, i.e. $A(q) = \{a \mid q \xrightarrow{a}, a \notin A^H\}$.

Definición 2.1.3 (Trace Semantic)**.** *Dado un LTS $S = \langle Q, q_0, A, \rightarrow \rangle$* el conjunto de trazas *de un estado $q \in Q$, notación traces(q), está definido como*

$$traces(q) = \{\top\} \cup \{a_1 a_2 \ldots a_n \top \mid a_1 a_2 \ldots a_n \text{ es una traza de } q\}$$

El conjunto de trazas de S, notación traces(S), es el conjunto de trazas de su estado inicial, i.e. traces$(S) =$ traces(q_0). Dos procesos S y T son equivalentes por trazas *si traces$(S) =$ traces(T).*

El uso de $\top$ al final de la traza en la Def. 2.1.3 para denotar el fin de una observación por parte del usuario. Este símbolo también se utiliza en ready trace semantic.

Definición 2.1.4 (Ready Trace Semantic)**.** *Dado un LTS $S = \langle Q, q_0, A, \rightarrow \rangle$* el conjunto de trazas ready *de un estado $q \in Q$, notación rTraces(q), está definido como*

$$\begin{aligned}
rTraces(q) &= \{\top\} \cup rTraces_1(q) \cup rTraces_2(q) \cup rTraces_3(q) \\
rTraces_1(q) &= \{a\phi \mid \exists q' : q \xrightarrow{a} q', a \in A^V, \phi \in rTraces(q')\} \\
rTraces_2(q) &= \{A(q)\phi \mid q \not\xrightarrow{\tau}, \phi \in rTraces(q)\} \\
rTraces_3(q) &= \{\phi \mid \exists q' : q \Rightarrow q', \phi \in rTraces(q')\}
\end{aligned}$$

El conjunto de trazas ready de S, notación rTraces(S), es el conjunto de trazas de su estado inicial, i.e. rTraces$(S) =$ rTraces(q_0). Dos procesos S y T son equivalentes por ready trace *si rTraces$(S) =$ rTraces(T).*

El conjunto $rTraces_1(q)$ es usado para modelar las acciones visibles que se ejecutan, el conjunto $rTraces_2(q)$ para modelar las acciones que pueden ejecutarse en un estado estable q y $rTraces_3(q)$ para considerar las posibles ejecuciones de transiciones internas.

Definición 2.1.5 (Weak Bisimulation)**.** *Sean* $S = \langle Q_S, s_0, A_S, \rightarrow_S \rangle$ *y* $T = \langle Q_T, t_0, A_T, \rightarrow_T \rangle$ *dos LTS. Una relación* $R \subseteq Q_S \times Q_T$ *es una* bisimulación débil *si para todo* $s \; R \; t$ *vale la* propiedad de transferencia *definida por*

1. *si* $s \xrightarrow{a} s'$ *entonces* $t \stackrel{\hat{a}}{\Rightarrow} t'$ *y* $s' \; R \; t'$ *y*

2. *si* $t \xrightarrow{a} t'$ *entonces* $s \stackrel{\hat{a}}{\Rightarrow} s'$ *y* $s' \; R \; t'$.

Los LTS S *y* T *son* weak bisimilares*, notación* $S \sim T$*, si existe una bisimulación débil* R *tal que* $s_0 \; R \; t_0$.

Es directo interpretar una IA S como un LTS. Simplemente debemos tomar como conjunto de acciones visibles la unión de las acciones de entrada y salida, i.e. $A^V = A^I \cup A^O$. Luego extender estas definiciones a IA es directo.

Las semánticas elegidas son interesantes para demostrar nuestra afirmación pues cubren distintos espectros de semánticas para LTS. Trace semantic es la más simple de las semánticas. Ready trace semantic podría considerarse intermedia, pues si bien refleja comportamientos lineales del sistema, tiene suficiente poder expresivo para expresar características particulares de cada estado estable, en este caso puntual, las acciones que el estado puede ejecutar. Observemos la dinámica de estas semánticas: dado dos LTS, primero se generan las trazas de cada sistema, luego éstas se comparan. Esta dinámica deja de lado la estructura de los sistemas y se centra en las trazas que cada uno puede generar. Esto no ocurre con la bisimulación débil, pues la propiedad de transferencia garantiza que los estados relacionados poseen una estructura similar, i.e. dos estados relacionados pueden imitar las transiciones que puede realizar el otro (quizás utilizando transiciones internas).

Analicemos cómo se comportan las semánticas definidas al momento de comparar las IA B y B'. Recordemos que estos sistemas, bajo las suposiciones realizadas para las interfaces, presentan comportamientos distintos. Si se compara B y B' utilizando trace semantic se obtiene que ambas interfaces son iguales pues $traces(B) = traces(B')$. Notemos que las acciones *soloTé;* y *soloCafé;* son acciones internas y por lo tanto no aparecen en las trazas de los procesos. Por otro lado si usamos ready trace semantic la diferencia se hace presente debido a que

$$moneda?\{café?, té?\}\top \in rTraces(q_0) \quad \text{y} \quad moneda?\{café?, té?\}\top \notin rTraces(r_0)$$

luego $rTraces(q_0) \neq rTraces(r_0)$. Notemos que si bien a primera vista pareciese que la bisimulación débil tiene mayor poder de diferenciación, en realidad esto no es así. De hecho, $B \sim B'$ mediante la relación

$$\{(q_0, r_0), (q_1, r_1), (q_1, r_4), (q_2, r_2), (q_3, r_3)\}$$

Esto se debe a que las acciones que se pueden realizar en el estado q_1 también pueden realizarse desde r_1 y r_4 quizás realizando previamente algunas transiciones internas.

La equivalencia por ready trace logra diferenciar ambos modelos pues sus características permiten saber que acciones puede ejecutar un estado estable. En [37, 38] se justifica esta característica dotando al usuario con un menú. Este menú muestra que acciones está esperando el sistema. Extender esta suposición al contexto de IA es razonable, pues una interfaz podría

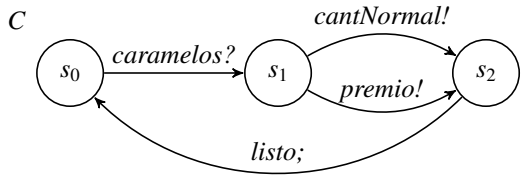


Figura 2.3.: Una expendedora de caramelos generosa

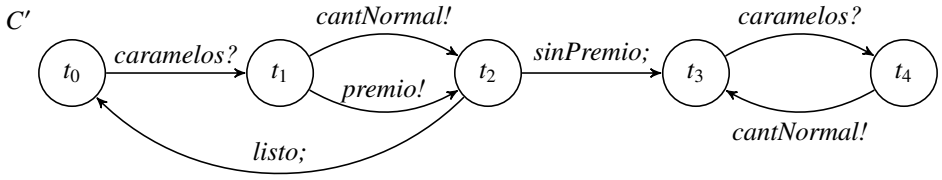


Figura 2.4.: Una expendedora de caramelos generosa pero con límite

brindar un mecanismo para informar que acciones de entrada está esperando. Sin la suposición de este mecanismo, no se podría justificar el utilizar la semántica ready trace.

Ya se menciono que en [37, 38] no existe la diferencia entre acciones de entrada y acciones de salida, por lo tanto si utilizamos directamente la semántica ready trace, el menú también informará que acciones de salida puede ejecutar un estado y esto podría resultar no razonable. Para ver esto comparemos las interfaces C y C' que se modelan en las Figuras 2.3 y 2.4. La primera es una máquina expendedora de caramelos generosa, pues no necesita monedas para funcionar. No sólo eso, luego del pedido de caramelos, a veces entrega los mismos, pero otras veces entrega un premio especial. Luego de ambos casos, retorna al estado inicial. El segundo caso es una máquina expendedora de caramelos generosa, pero con límites. Esto quiere decir que en algún momento, luego de cumplir con un pedido, podrá dejar de brindar premios especiales y comportarse como una maquina que no cobra por caramelos pero tampoco da premios. Podemos suponer que el fabricante de la máquina no brinda métodos para saber cuando la máquina deja de brindar premios. Luego, un usuario normal no debería poder notar diferencias entre la máquina C y la máquina C'.

Las expendedoras de bebidas y las expendedoras de caramelos no presentan diferencias con respecto al tipo de sistema que modelan, por lo tanto la semántica que fue útil para diferenciar las máquinas expendedoras de bebidas B y B', si realmente fuera útil para este tipo de procesos, no debería diferenciar entre C y C', i.e. se debería satisfacer $rTraces(C) = rTraces(C')$. Esto no es así, dado que $caramelos?premio!caramelos?\{cantNormal!\} \in rTraces(C') - rTraces(C)$.

Esto se debe a que las naturalezas de las acciones de entrada y las acciones de salida son distintas. Luego, las suposiciones para ambos tipos de acciones no tienen por que ser compartidas. En este ejemplo, la semántica de ready traces extiende la suposición de contar con un mecanismo para ver las acciones de entrada que el estado espera a las acciones de salida que el estado

puede ejecutar.

La diferencia entre acciones de entrada y salida es más evidente al estudiar cómo es la composición para los sistemas que estamos estudiando. En [25, 26] se define a las IA como una especificación de la interfaz de un sistema, la cual describe el comportamiento correcto de la interacción entre éste y su entorno. Por ejemplo la IA B garantiza que en el estado inicial, luego de introducir una moneda, se podrá elegir entre café o té sin problemas, pero si se vuelve a introducir otra moneda entonces el comportamiento del sistema no está definido y podría pasar cualquier cosa. Por esta razón al componer dos IA, para que la interacción se realice sin problemas, una acción común de salida no puede ser ejecutada por una de las interfaces si la otra no está lista para recibirla.

La primera condición que se impone sobre la composición es que si una acción es común, entonces esta acción es una entrada en uno de los procesos y una salida en el otro. Estas acciones son utilizadas para sincronizar los procesos. Dado un conjunto de acciones A_i, A_i^I (resp. A_i^O, A_i^H) denotará el subconjunto de acciones de entrada (resp. de salida, internas) del conjunto A_i.

Definición 2.1.6. *Sean S y T dos IA, y sea $\mathsf{shared}(S,T) = A_S \cap A_T$ el conjunto de* acciones compartidas*. Diremos que S y T son* componibles *si $\mathsf{shared}(S,T) = (A_S^I \cap A_T^O) \cup (A_S^O \cap A_T^I)$.*

Previo a la definición de la composición se define un sistema intermedio, al cual denominaremos *producto*. El producto de dos IA P y Q componibles está definido de forma similar que la composición paralela binaria de CSP [47] *(i)* el espacio de estados del producto es el producto de los estados de las componentes, *(ii)* sólo las acciones comunes pueden sincronizar, i.e. ambas componentes deben realizar la transición con la misma acción (notar que una es una acción de entrada, la otra de salida), y *(iii)* las transiciones con acciones no compartidas pueden ejecutarse independientemente. Además, las acciones compartidas se transforman en acciones ocultas en la composición, pues forman parte del funcionamiento interno del nuevo componente.

Definición 2.1.7. *Sean S y T dos IA componibles. El* producto *de S y T es el IA $S \otimes T = \langle P \times Q, (p_0, q_0), A^I_{S\otimes T} \cup A^O_{S\otimes T} \cup A^H_{S\otimes T}, \rightarrow_{S\otimes T}\rangle$ donde:*

- $A^I_{S\otimes T} = A^I_S \cup A^I_T - \mathsf{shared}(S,T)$,
- $A^O_{S\otimes T} = A^O_S \cup A^O_T - \mathsf{shared}(S,T)$, *y*
- $A^H_{S\otimes T} = A^H_S \cup A^H_T \cup \mathsf{shared}(S,T)$*; finalmente*
- $(s,t) \xrightarrow{a}_{S\otimes T} (s',t')$ *si alguna de las siguiente condiciones es válida:*
  - $a \in A_S - \mathsf{shared}(S,T)$, $s \xrightarrow{a}_S s'$ *y* $t = t'$*;*
  - $a \in A_T - \mathsf{shared}(S,T)$, $t \xrightarrow{a}_T t'$ *y* $s = s'$*;*
  - $a \in \mathsf{shared}(S,T)$, $s \xrightarrow{a}_S s'$ *y* $t \xrightarrow{a}_T t'$.

Probablemente habrá muchos estados en el producto $S \otimes T$ para las cuales una de las componentes, por ejemplo S, produce una acción de salida común que la otra componente, T en este caso, no está lista para aceptar (i.e., la acción de entrada correspondiente no está habilitada en el estado en el que se encuentra). Entonces S está violando las suposiciones de T. Esto no es aceptable pues la interacción no es correcta: S envía un mensaje a T y T no está listo para recibirlo. Este tipo de estados son denominados *estados de error*.

Definición 2.1.8. *Sean S y T dos IA componibles. Un estado $(p,q) \in S \otimes T$ es un* estado de error *si existe una acción $a \in \mathsf{shared}(S,T)$ tal que alguna de las siguientes condiciones se satisface:*

- $a \in A^O_S$, $s \xrightarrow{a}_S$ y $t \not\xrightarrow{a}_T$.
- $a \in A^O_T$, $t \xrightarrow{a}_T$ y $s \not\xrightarrow{a}_S$.

Si el estado $(s_0, t_0) \in S \otimes T$ no alcanza estados de error entonces s_0 y t_0 interactúan de forma correcta y por lo tanto son compatibles. Pero la presencia de estados de error en $S \otimes T$ no es necesariamente una evidencia de que una ejecución desde el estado (s_0, t_0) producirá una violación a las suposiciones de alguna de las componentes. Esto se debe a que alcanzar un estado de error desde (s_0, t_0) dependerá de los estímulos que se le brinden al mismo. Luego, si se restringen los estímulos que permiten alcanzar un estado de error, éste nunca se alcanzará. Por otro lado, podría ser el caso que el estado (s_0, t_0) no necesite ningún estímulo del ambiente para alcanzar un estado de error debido a que esto puede hacerse de forma autónoma (i.e., vía acciones de salida o internas). En este caso, (s_0, t_0) será un estado incompatible.

Definición 2.1.9. *Sean S y T dos IA componibles. Un estado $(s, t) \in S \otimes T$ es un* estado incompatible *si existe un estado de error alcanzable desde (s, t) a través de una ejecución autónoma. Si un estado no es incompatible entonces es* compatible*. Diremos que S y T son compatibles sii sus estados iniciales lo son.*

La composición entre S y T será posible sólo si son compatibles. El nuevo proceso será el resultado de eliminar del producto toda transición con una acción de entrada que alcance un estado incompatible. De esta forma se evitará alcanzar estados de error.

Definición 2.1.10. *Sean S y T dos IA componibles. La* composición *$S \parallel T$ es el IA que resulta de tomar el producto $S \otimes T$ y eliminar las transiciones $r \xrightarrow{a}_{S\otimes T} r'$ tales que (*I*) r es un estado compatible de $S \otimes T$, (*II*) $a \in A^I_{S\otimes T}$, y (*III*) r' es un estado incompatible de $S \otimes T$.*

Ejemplo 2.1.1. *Sean S y T los primeros dos IA graficados en la Fig. 2.5. Entonces* shared$(S, T) = \{a, b, c\}$*. Sea el producto $S \otimes T$ el tercer IA que se grafica, mientras que la composición $S \parallel T$ es el cuarto. Notemos que este último se obtiene del producto luego de eliminar la transición $s_0 \otimes t_0 \xrightarrow{m_2} s_5 \otimes t_0$. El estado $s_5 \otimes t_0$ es un estado incompatible debido a que alcanza de forma autónoma un estado de error: $s_6 \otimes t_1$. $s_6 \otimes t_1$ es un estado de error pues s_6 puede ejecutar la acción común $c!$ pero t_1 no puede recibirlo, i.e. $s_6 \xrightarrow{c!}$ y $t_1 \not\xrightarrow{c?}$. Al mismo tiempo $t_1 \xrightarrow{b!}$ pero $s_6 \not\xrightarrow{b?}$.*

Notemos que en la composición de este tipo de sistema la comunicación no se realiza por *handshaking*, como en el caso de CSP. En la Figura 2.6 tenemos los LTS S_0 y T_0, los cuales se componen con el proceso U_0 siguiendo este estilo, i.e. existe una cooperación entre los procesos que se componen. Esto es claro al momento de componer S_0 con U_0: el estado u_0 no avanza hasta que en S_0 se transiciona desde s_0 a s_1, luego de esto, ambos sistemas transicionan juntos.

Por otro lado, en las Figuras 2.7 y 2.8 tenemos los mismo procesos transportados al contexto de IA. Para realizar esto, la acción a tiene que ser considerada como una acción de entrada en una de las interfaces y como acción de salida en la otra. En la Figura 2.7 la acción a se define en U_1 como una acción de entrada, por lo tanto en S_1 y T_1 se define como una acción de salida. En 2.8 se define el tipo de a de forma opuesta. Consideremos primero los procesos en la Figura 2.7. En

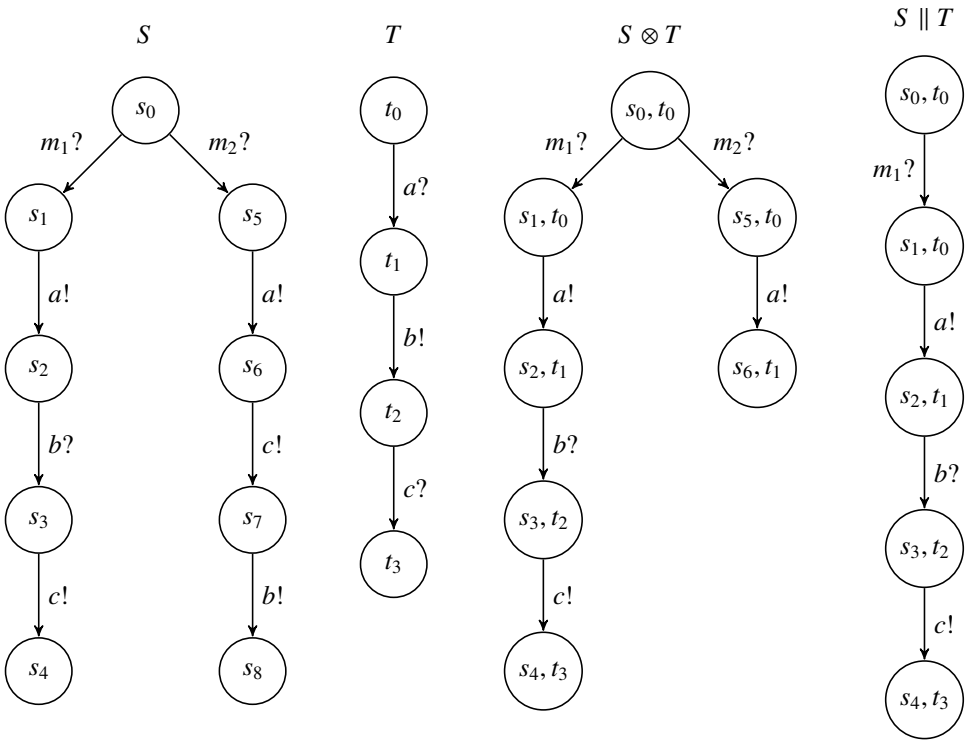


Figura 2.5.: Ejemplo de composición.

la composición de S_1 y U_1, la interfaz U_1 debe esperar que la interfaz S_1 ejecute la acción $a!$ para progresar. Esto ocurrirá sólo después que se realiza la transición s_1. Notemos que esto no pasa en los procesos de la Figura 2.8, donde los tipos de acciones están invertidos. Al realizar la composición de S_2 con T_2 encontramos una incompatibilidad. El estado u_0 puede transicionar al estado u_1 mediante la acción $a!$ cuando lo desee. Si decide esperar a que s_0 transicione a s_1 entonces la sincronización se produce sin problemas, por otro lado, si esto no es así, el mensaje $a!$ se pierde, lo cual indica la incompatibilidad entre los procesos. Este problema no ocurre al componer T_2 y U_2 pues T_2 no debe realizar una transición interna para estar listo para recibir la acción de entrada $a?$.

Una semántica adecuada para estos tipos de modelos debería considerar a los modelos S_1 y T_1 equivalentes; mientras que a S_2 y T_2 los debería considerar diferentes, dado que uno sincroniza bien con U_2 y el otro no. Utilizando las semánticas ya definidas obtendríamos para todos los casos que S_2 y T_2 son equivalentes:

$$traces(S_2) = traces(T_2) = \{\top, a?T\}$$
$$rTraces(S_2) = rTraces(T_2) = \{T, a?T, \{a?\}T, \{a?\}a?T\}$$
$$S_2 \sim T_2 \text{ con } \sim = \{(s_0, t_0), (s_1, t_0), (s_2, t_1)\}$$

No es raro que la semántica basada en trazas y la bisimulación no diferencien entre ambos

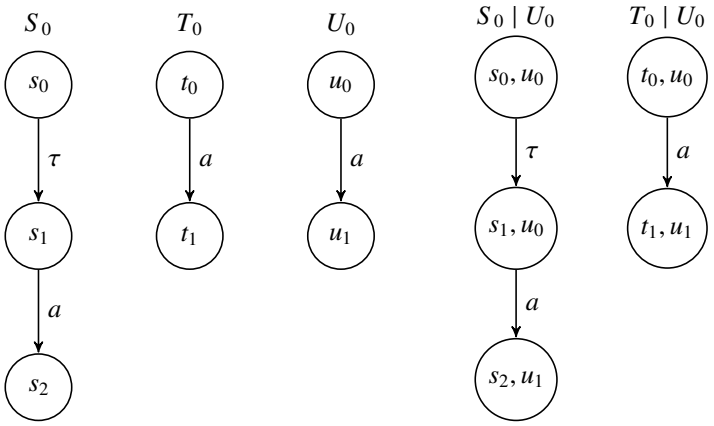


Figura 2.6.: Composición al estilo CSP. Los procesos sincronizan usando *handshaking*

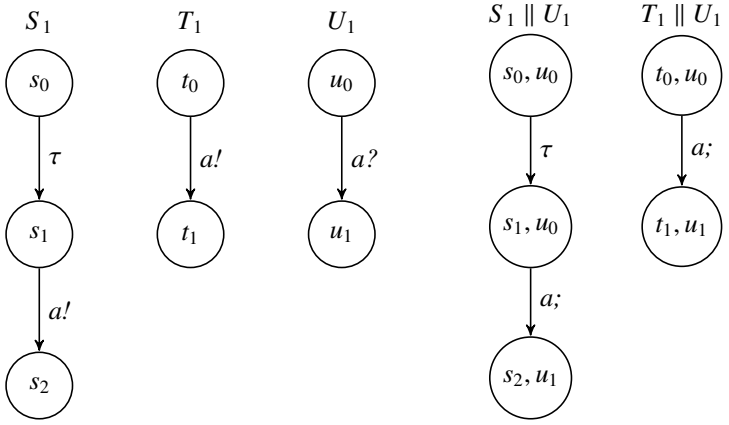


Figura 2.7.: Primera adaptación de los LTS de la Fig. 2.6 al contexto de IA. Las IA S_1 y T_1 son compatibles con la IA U_1.

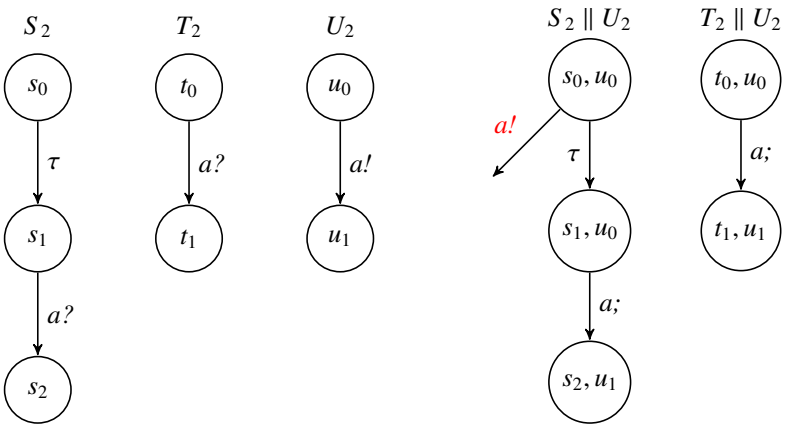


Figura 2.8.: Segunda adaptación de los LTS de la Fig. 2.6 al contexto de IA. La IA S_2 no es compatible con la IA U_2 pero si es compatible con T_2.

procesos por lo ya expuesto. Por otro lado, dado que ready trace funcionó bien para distinguir las expendedoras de bebidas de las Figuras 2.1 y 2.2, donde el problema se origino con las acciones de entradas, uno esperaría que para estos ejemplos también funcione. Como se puede puede apreciar, este no es el caso.

La particularidad de S_2 con respecto a T_2 es que no está listo para recibir la acción común a en su estado inicial. Por otro lado, T_2 sí lo está y por esta razón se compone bien con U_2. Si observamos la definición de ready trace (Def. 2.1.4), sólo en los estados estables es posible observar que acciones espera un estado, por esta razón el estado s_0 de S_2 no genera una traza $\emptyset\top$, la cual diferenciaría a S_2 de T_2.

Para solucionar este inconveniente, se podría definir una nueva relación $rTraces'$, la cual es una variante de $rTraces$, la cual permite observar que acciones puede ejecutar un estado sin la restricción de que sea estable:

$$rTraces'(p) = \{T\} \cup rTraces_1(p) \cup rTraces'_2(p) \cup rTraces_3(p)$$
$$rTraces'_2(p) = \{A(p)\phi \mid \phi \in rTraces(p), \phi \text{ es finita}\}$$

Con esta variante, S_2 y T_2 son diferenciables: $\emptyset T \in rTraces'(S_2)$ pero $\emptyset T \notin rTraces'(T_2)$. No sólo esto, la diferencias entre las expendedoras de bebidas de las Figuras 2.1 y 2.2 aún se mantiene, i.e. $rTraces'(B) \neq rTraces'(B')$. Por otro lado, las expendedoras de caramelos de las Figuras 2.3 y 2.4 siguen siendo consideradas diferentes.

De los ejemplos presentados se desprende que las acciones de entrada y de salida no se pueden manipular de la misma manera. Esto no es una novedad, en [25] ya se presenta una relación que distingue entre las acciones de entrada y las acciones de salida. La misma se denomina *alternating simulation* y se utiliza para representar una noción de refinamiento entre autómatas de interface. Esta noción de refinamiento garantiza que el refinamiento de una interfaz puede aceptar al menos las mismas acciones de entrada y producir a lo sumo las mismas acciones de salida que la interfaz original. Además en [57] se presenta una relación que fortalece las condiciones de

la relación alternating simulation. Esta relación se denomina *refinamiento estricto en entradas* (*SIR*, del inglés, *strict input refinement*) y requiere que ambos procesos acepten exactamente las mismas acciones de entrada.

Definición 2.1.11. *Sean S y T dos IA. Una relación $R \subseteq Q_S \times Q_T$ es un* refinamiento estricto en entradas *(*SIR*, del inglés, strict input refinement) si $s_0\ R\ t_0$ y para todo $s\ R\ t$:*

(a) $\forall a \in A_S^I, s' \in S$, *si* $s \xrightarrow{a}_S s'$ *entonces* $\exists t' \in T : t \xrightarrow{a} t'$ *y* $s\ R\ t'$.

(b) $\forall a \in A_T^I, t' \in T$, *si* $t \xrightarrow{a} t'$ *entonces* $\exists s' \in S : s \xrightarrow{a} s'$ *y* $s'\ R\ t'$.

(c) $\forall a \in A^O, t' \in T$, *si* $t \xrightarrow{a} t'$ *entonces* $\exists s' \in S : s \overset{\varepsilon}{\Rightarrow} \xrightarrow{a} s'$ *y* $s'\ R\ t'$;

(d) $\forall a \in A^H, t' \in T$, *si* $t \xrightarrow{a} t'$ *entonces* $\exists s' \in S : s \overset{\varepsilon}{\Rightarrow} s'$ *y* $s'\ R\ t'$.

Decimos que S es refinado (estrictamente en entradas) por T, *o, T* refina (estrictamente en entradas) a S, *notación $S \preccurlyeq T$, si existe una relación SIR $\preccurlyeq$ entre los estados iniciales de S y T. Dos procesos S y T son SIR equivalentes, notación $S =_{SIR} T$, si T refina a S y S refina a T.*

La relación alternating simulation [25] se define de forma similar pero sólo se requiere que se satisfagan las condiciones (a), (c) y (d).

La Def. 2.1.11 impone unas condiciones para las acciones de entrada y otras para las acciones de salida e internas. Las condiciones (a) y (b) garantizan que dos estados relacionados pueden recibir el mismo conjunto de entradas. Por otro lado, las condiciones (c) y (d) garantizan que T no realiza ninguna transición autónoma que S no realice.

La nueva relación de equivalencia basada en la relación SIR se amolda bien a todos los ejemplo anteriores.

$$B \neq_{\text{SIR}} B' \qquad C =_{\text{SIR}} C' \qquad S_1 =_{\text{SIR}} T_1 \qquad S_2 \neq_{\text{SIR}} T_2$$

La relación SIR es sólo una de las tantas semánticas que se pueden definir para IA. En el siguiente sección se estudia de forma general la definición de este tipo de semánticas.

2.2. Nociones de observabilidad

Un proceso interactúa con su entorno a través de su interfaz. Quien interactúa con ese proceso podrá reconocer su comportamiento sólo a través de tal interacción. De esta manera, ese usuario (sea hombre o máquina) podrá distinguir un proceso de otro sólo con la precisión que dicha interacción permite. Así, un usuario que se limita sólo a observar las acciones que un proceso realiza distinguirá mucho menos que otro usuario que tenga permitido inyectar eventos y observar las posibles respuestas.

Al modo en que un usuario tiene permitido interactuar con un proceso lo llamamos *noción de observabilidad*. Una noción de observabilidad está definida por un conjunto de *tipos de observaciones*. Cada tipo de observación establece una manera en la que el usuario puede interactuar con el proceso.

Para poder definir los tipos de observaciones se necesita establecer claramente las suposiciones del modelo. Esto se estableció en la sección anterior, en la Suposición 1.

A continuación definimos los distintos tipos de observación.

- $\top$ El usuario finaliza la sesión de observación. Esta acción se puede realizar en todo momento. Ninguna observación posterior es posible.
- a^O El usuario puede observar la ejecución de una acción de salida.
- $\rightleftarrows$ El usuario ejecuta acciones de entrada que se encuentran habilitadas en la interfaz y la interfaz reacciona a este estímulo realizando la transición correspondiente.
- $\rightsquigarrow$ El usuario ejecuta acciones de entrada que no se encuentran habilitadas en la interfaz. La ejecución de estas acciones es ignorada por la interfaz y no produce un comportamiento visible.
- F El usuario ejecuta acciones de entrada que no se encuentran habilitadas en la interfaz. Cuando se ejecuta una de estas acciones, la misma es rechazada y la ejecución termina.
- FT Similar al tipo F, pero luego de rechazar la acción, la ejecución continúa.
- R Similar al tipo F, pero en vez de rechazar la acción se provee la información sobre que acciones de entradas se encontraban habilitadas en el estado donde se ejecuto la entrada incorrecta.
- RT Similar al caso R, pero luego de mostrar qué acciones están habilitadas, la ejecución continúa.
- 0 El usuario puede detectar cuando un proceso es incapaz de realizar cualquier tipo de actividad, i.e. no puede realizar transiciones autónoma ni tampoco responder a alguna acción de entrada.
- δ Es posible concluir que el sistema no realiza ninguna actividad de forma autónoma. Es decir, no puede ejecutar una acción interna o de salida. Por lo tanto está esperando que el usuario ejecute alguna acción de entrada.
- $\bigwedge$ El usuario tiene la capacidad de hacer copias del sistema. Las cantidad de las mismas es arbitraria, pero finita, y se realizan con respecto a un estado particular del sistema.
- η El usuario tiene los mecanismos para garantizar que luego de realizar una copia, el sistema no ejecuta ninguna transición interna antes de ejecutar una acción de entrada.

b El usuario tiene los mecanismos para hacer copias del sistema de forma continua a través del tiempo. Es decir que el usuario puede usar el mecanismo provisto por el tipo $\bigwedge$ para realizar una copia, esperar un lapso de tiempo y volver a utilizarlo. Con esto será posible observar que ciertas ejecuciones son solamente posibles después de ciertas otras ejecuciones.

Δ El usuario puede detectar que el sistema se encuentra realizando una cantidad infinita de actividad interna donde no es posible la interacción, i.e., el sistema no habilita acciones de entrada. En este caso, decimos que el sistema *diverge*.

λ El usuario puede detectar que el sistema deja de interactuar, ya sea porque diverge o porque ha llegado a un estado final. En este caso decimos que estamos en presencia de un *deadlock*.

$\neg$ Es posible testear el sistema bajo todas las posibles condiciones internas y externas del mismo. Esto permite asegurar que una observación particular no es posible.

$\odot$ El usuario cuanta con un mecanismo para detectar actividad interna. Las transiciones internas serán detectables, pero no diferenciables. Es decir que dos transiciones internas con distintas acciones producirán la misma observación.

ε El usuario siempre supone que luego de una acción visible pueden ocurrir cero o más transiciones internas.

!! El usuario cuenta con los mecanismo para interactuar con una interfaz, luego de que se ejecute una acción visible, con la suficiente rapidez para garantizar que no se ejecutará ninguna acción interna previo a la interacción.

$\bigvee$ El usuario observa $\bigvee_{i\in I}\phi_i$, si puede observar al menos una de las observaciones ϕ_i.

τ Este tipo define una condición global sobre las posibles observaciones. La condición establece que no puede existir actividad interna previo a una observación generada con los tipo $\neg, \bigvee, \bigwedge$. El conjunto $\mathbb{O}(\tau)$ es el conjunto de observaciones que no empiezan con estos símbolos.

Como ya se mencionó, una noción de observabilidad es un conjunto de tipos de observaciones que definen los mecanismos con los que cuenta un usuario para realizar observaciones. Pero no todo conjunto compuesto por tipos de observaciones definirá una noción de observabilidad, dado que este conjunto podría resultar incoherente. La definición de noción de observabilidad establecerá restricciones que garantizan la coherencia.

Definición 2.2.1. *Definimos como una* noción de observabilidad*, como un subconjunto N tal que*

$$N \subseteq \{a^O, \top, \rightleftarrows, \rightsquigarrow, F, R, FT, RT, 0, \delta, \bigwedge, \neg, \odot, \varepsilon, \tau, \bigvee, \eta, b, \lambda, \Delta\}$$

el cual satisface las siguientes condiciones:

1. $\{a^O, \top, \varepsilon, \tau, \bigvee\} \subseteq N$
2. $|\{\rightsquigarrow, F, FT, R, RT\} \cap N| = 1 \Rightarrow \rightleftarrows \in N$ *y* $|\{\rightsquigarrow, F, FT, R, RT\} \cap N| \leq 1$
3. $b \in N \Rightarrow \eta \in N$ *y* $\eta \in N \Rightarrow \{\bigwedge, \rightleftarrows\} \subseteq N$
4. $\Delta \in N \Rightarrow 0 \in N$
5. $\odot \in N \Rightarrow \{0, \delta\} \subseteq N$

La condición 1 establece cuales son los tipos básicos que constituyen una noción de observabilidad: las acciones de salida son siempre visibles (a^O); el usuario puede abandonar en cualquier momento la observación ($\top$); el usuario siempre puede suponer que luego de la ejecución de una acción visible se produce actividad interna (ε); la condición global τ siempre es válida; finalmente, siempre es posible utilizar la disyunción ($\bigvee$). La primera parte de la condición 2 ($|\{\rightsquigarrow, F, R, FT, RT\}| = 1 \Rightarrow \rightleftarrows \in N$) modela que si un usuario puede ejecutar una acción de entrada que no se encuentra habilitada entonces esto significa que el usuario está interactuado con la interfaz; por lo tanto la interfaz deberá responder a la ejecución de acciones de entradas habilitadas. La segunda parte establece que el sistema podrá comportarse, a lo sumo, de una forma particular cuando se ejecuta una acción que no se encontraba habilitada. La condición 3 establece la jerarquía entre las distintas formas de realizar copias del sistema. Realizar copias del tipo η no sería razonable sin antes poder realizar copias "comunes", i.e. $\eta \in N \Rightarrow \bigwedge \in N$; además, el tipo η está definido con respecto a las acciones de entrada, por esta razón algún tipo de interacción debe estar incluida , i.e. $\rightleftarrows \in N$. Por otro lado el tipo de copia b habilita el realizar copias del tipo η. La condición 4 establece que no es posible definir divergencia si no se cuenta con un mecanismo para detectar finalización. Si esto fuese así, no se podría diferenciar entre un sistema que realiza actividad interna infinita y un estado que no puede realizar ninguna actividad. La condición 5 establece que el observar transiciones internas habilita detectar estados estables y estados finales. Por la suposición de weak fairness, todo estado que no sea estable o final, luego de una cantidad de tiempo, debería producir al menos la observación de un transición interna; cuando esto no ocurra estamos en presencia de estado estable o final.

Dado que los mecanismos con los que cuenta un usuario para observar una interfaz ya están definidos, ahora nos enfocaremos en definir las observaciones que se realizan. Una observación del sistema será representada por una *fórmula de observación*. Estas fórmulas se construyen para cada estado del sistema a partir de una noción de observabilidad. El comportamiento observable de una IA dada una noción de observabilidad es el conjunto de fórmulas que genera el estado inicial de la misma.

Definición 2.2.2. *Sea* $S = \langle Q, s_0, A^I \cup A^O \cup A^H, \rightarrow \rangle$ *una IA y N una noción de observabilidad. La función* $\hat{O}_N : Q \rightarrow \mathcal{P}(\mathbb{O}_N)$ *está definida inductivamente en función de los tipos en N; para cada tipo de observación en N se debe satisfacer su correspondiente cláusula en la Tabla 2.1 más la siguiente condición: si* $\{\eta, \rightsquigarrow\} \subseteq N$*, entonces se agrega también la cláusulas* ($\eta_{\rightsquigarrow}$).

El conjunto de observaciones de un estado $s \in Q$ con una noción de observabilidad *N, notación* $O_N(q)$*, está definido por:*

- *Si* $!! \in N$ *o* $\odot \in N$ *entonces* $O_N(q) = \hat{O}_N(q)$.

$(\top)$	$\top \in$	$O_N(q)$	$\forall q \in Q$
(a^O)	$a\phi \in$	$O_N(q)$	si $a \in A^O$ y $\exists q' \in Q$: $q \xrightarrow{a} q'$ y $\phi \in O_N(q')$
$(\rightleftarrows)$	$a\phi \in$	$O_N(q)$	si $a \in A^I$ y $\exists q' \in Q$: $q \xrightarrow{a} q'$ y $\phi \in O_N(q')$
$(\rightsquigarrow)$	$a\phi \in$	$O_N(q)$	si $a \in A^I$, $q \xrightarrow{a}$ y $\phi \in O_N(q)$
(F)	$\not{a} \in$	$O_N(q)$	si $a \in A^I$ y $q \xrightarrow{a}$
(FT)	$\not{a}\phi \in$	$O_N(q)$	si $a \in A^I$, $q \xrightarrow{a}$ y $\phi \in O_N(q)$
(R)	$X \in$	$O_N(q)$	si $X = I(q)$
(RT)	$X\phi \in$	$O_N(q)$	si $X = I(q)$ y $\phi \in O_N(q)$
(ε)	$\varepsilon\phi \in$	$O_N(q)$	si $\phi \in O_N(q)$
(τ)	$\phi \in$	$O_N(q)$	si $\exists q' \in Q, a \in A^H : q \xrightarrow{a} q'$ y $\phi \in O_N(q')$ y $\phi \in \mathbb{O}(O)$
(0)	$0 \in$	$O_N(q)$	si $q \xrightarrow{a}$ para todo $a \in A$
(δ)	$\delta\phi \in$	$O_N(q)$	si $q \xrightarrow{a}$ para todo $a \in A^O \cup A^H$ y $\phi \in O_N(q)$
$(\bigwedge)$	$\bigwedge_{i\in I} \phi_i \in$	$O_N(q)$	si $\phi_i \in O_N(q)$ para todo $i \in I$
(η)	$\phi a \psi \in$	$O_N(q)$	si $a \in A^I$ y $\exists q' \in Q : q \xrightarrow{a} q'$, $\phi \in O_N(q)$ y $\psi \in O_N(q')$
$(\eta_{\rightsquigarrow})$	$\phi a \psi \in$	$O_N(q)$	$a \in A^I$ y $q \xrightarrow{a}$ y $\phi, \psi \in O_N(q)$
(b)	$\phi \tau \psi \in$	$O_N(q)$	si $\exists q' \in Q, a \in A^H : (q \xrightarrow{a} q' \bigvee q = q')$ y se da que $\phi \in O_N(q)$ y $\psi \in O_N(q')$
(Δ)	$\Delta \in$	$O_N(q)$	si $\exists q_1, q_2, \ldots \in Q : q \xrightarrow{\tau} q_1 \xrightarrow{\tau} q_2 \ldots$ y $q_i \xrightarrow{a}$ con $i \in \mathbb{N}, a \in A^I$
(λ)	$\lambda \in$	$O_N(q)$	si $q \xrightarrow{a}$ para todo $a \in A$ o
			$\exists q_1, q_2, \ldots \in Q : q \xrightarrow{\tau} q_1 \xrightarrow{\tau} q_2 \ldots$ y $q_i \xrightarrow{a}$ con $i \in \mathbb{N}, a \in A^I$
$(\neg)$	$\neg\phi \in$	$O_N(q)$	si $\phi \notin O_N(q)$
$(\odot)$	$\odot\phi \in$	$O_N(q)$	si $\exists q' \in Q, a \in A^H : q \xrightarrow{a} q'$ y $\phi \in O_N(q')$

Tabla 2.1.: Semánticas de observaciones

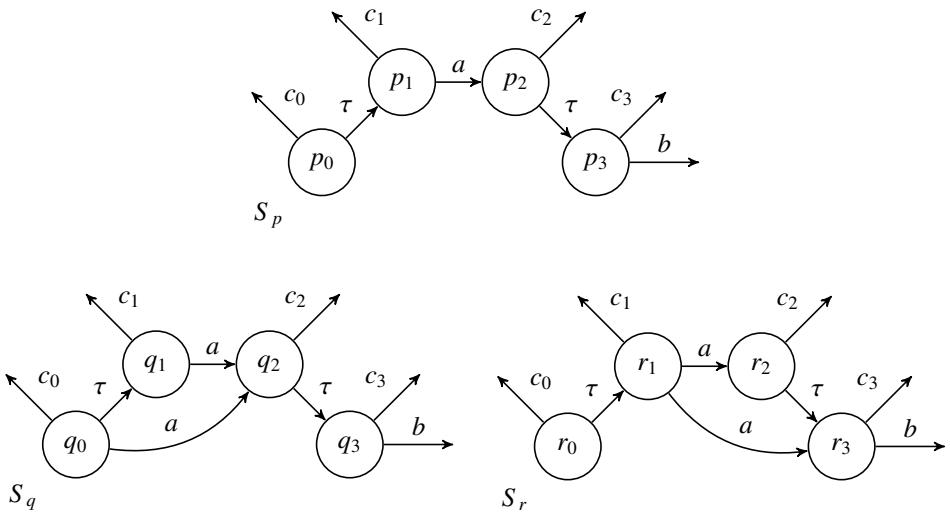


Figura 2.9.: ILTS para ejemplificar la manipulación de transiciones internas

- *Si* $\{!!, \odot\} \cap N = \emptyset$ *entonces* $O_N(q) = \{\phi \in \hat{O}_N(q) \mid$ *todo símbolo* $a \in A^I \cup A^O$ *que aparece en* ϕ *es seguido por el símbolo* $\varepsilon\}$.

El conjunto de observaciones de S con una noción de observabilidad N *se define como* $O_N(S) = O_N(s_0)$. *Un estado* s *genera una observación* ϕ, *notación* $s \models \phi$, *si* $\phi \in O_N(q)$.

Si bien las reglas de la Tabla 2.1 son directas no es claro porqué la definición de O_N cambia en función de $!! \in N$ o $\odot \in N$. Esta división por casos junto a las cláusulas ε, τ, η y b proveen la maquinaria necesaria para modelar las observaciones que se generan a partir de las transiciones internas. Cabe destacar que esta forma de manipulación es la misma que se presenta en [37].

Para explicar cómo funciona esta manipulación utilizaremos como ejemplo una bisimulación débil (Def. 2.1.5). Recordemos que una bisimulación débil R es una relación tal que si $s\ \mathsf{R}\ t$ y $s \xrightarrow{a} s'$ entonces existe t' tal que $t \overset{a}{\Rightarrow} t'$ y $t' \sim t'$ y viceversa. Observemos que el proceso t debe ejecutar a y alcanzar un estado t' tal que $s' \sim t'$; la ejecución de a puede ser precedida y seguida de transiciones internas. Luego, esta relación no pone restricciones sobre los estados intermedios necesarios para alcanzar t'. Esta relación puede ser representada por la siguiente noción de observabilidad $\mathrm{WB} = \{a^O, \top, \tau, \varepsilon, \bigvee, \rightleftarrows, \bigwedge, \neg\}$.

Lema 2.2.1. *Sean* S *y* T *dos IA y sea* $WB = \{a^O, T, \tau, \varepsilon, \bigvee, \rightleftarrows, \bigwedge, \neg\}$ *una noción de observabilidad. Luego* $S \sim T$ *sii* $O_{WB}(S) = O_{WB}(T)$.

Omitimos la demostración del lema, pues esta será evidente al introducir la caracterización relacional de las nociones de observabilidad en la Sección 2.3.

Comparemos las IA de la Figura 2.9 de acuerdo a la noción de observabilidad WB. Comencemos por comparar los estados p_0 y q_0. Notemos que p_2 y q_2 son isomorfos, por consiguiente $\phi \in O_N(p_2)$ si y sólo si $\phi \in O_N(q_2)$. Por otro lado, q_0 ejecuta una transición con una acción a

que p_0 no puede ejecutar. A pesar de esto, p_0 generará las mismas observaciones que q_0 gracias al tipo τ: supongamos $q_0 \models a\varepsilon\phi$ (a es seguida de ε por Def. 2.2.2 y $\{!!, \odot\} \cap N = \emptyset$). Luego $q_2 \models \varepsilon\phi$ ya sea por el tipo ($\rightleftarrows$) o a^O según a sea de entrada o de salida (alternativamente, $q_0 \models a\varepsilon\phi$ si $q_1 \models a\varepsilon\phi$ por (τ), luego $q_2 \models \varepsilon\phi$ por ($\rightleftarrows$) o (a^O)). Pero luego $p_2 \models \varepsilon\phi$ y en consecuencia $p_1 \models a\varepsilon\phi$ ya sea por ($\rightleftarrows$) o (a^O). Finalmente $p_0 \models a\varepsilon\phi$ por (τ) dado que $a\varepsilon\phi \in \mathbb{O}_N(\tau)$. Inversamente es claro que si $p_0 \models a\varepsilon\phi$ entonces $q_0 \models a\varepsilon\phi$.

Consideremos ahora p_0 y r_0 y notemos que p_2 y r_2 (p_3 y r_3) satisfacen las mismas fórmulas por ser isomorfos. Nuevamente es claro que si $p_0 \models a\varepsilon\phi$ también $r_0 \models a\varepsilon\phi$. Por otro lado supongamos que $r_0 \models a\varepsilon\phi$ y que se satisface por $r_1 \models a\varepsilon\phi$ y (τ). Luego, sea por ($\rightleftarrows$) o (a^O), $r_3 \models \varepsilon\phi$. Por lo tanto $p_3 \models \varepsilon\phi$ y por (τ), $p_2 \models \varepsilon\phi$ (aquí ε garantiza $\varepsilon\phi \in \mathbb{O}_N(\tau)$). Por ($\rightleftarrows$) o ($a^O$), $p_1 \models a\varepsilon\phi$ y, por (τ), $p_0 \models a\varepsilon\phi$.

Si en cambio consideramos la noción de observabilidad $\mathrm{WB}_\eta = \mathrm{WB} \cup \{\eta\}$, los estados p_0 y q_0 no son equivalentes. En este caso, si $\phi \in O_{\mathrm{WB}_\eta}(q_2)$ entonces $c_0\varepsilon\top(a\varepsilon\phi) \in O_{\mathrm{WB}_\eta}(q_0)$ pero $c_0\varepsilon T(a\varepsilon\phi) \notin O_{\mathrm{WB}_\eta}(p_0)$. El tipo η permite detectar que uno de los sistemas, para imitar una transición visible, debe transicionar por estados no bisimilares al estado que ejecuto la acción, previo a la ejecución de la acción visible. En este caso, para imitar la transición $q_0 \to q_2$, p_0 debe transicionar primero al estado p_1 el cual no es bisimilar a q_0. Por otro lado, si se compara p_0 con r_0 utilizando WB_η se obtiene $O_{\mathrm{WB}_\eta}(p_0) = O_{\mathrm{WB}_\eta}(r_0)$.

En cambio, p_0 y r_0 no son equivalentes si se los compara con la noción de observabilidad $\mathrm{WB}_{!!} = \mathrm{WB} \cup \{!!\}$. Sea $(c_3 \wedge b) \in O(r_3)$ por el tipo ($\bigwedge$), $a(c_3 \wedge b) \in O(r_1)$ por (a^O) o ($\rightleftarrows$) (notar que luego de a no aparece ε porque $!! \in N$) y por (τ), $a(c_3 \wedge b) \in O(r_0)$; pero $a(c_3 \wedge b) \notin O(p_1)$ y por lo tanto $a(c_3 \wedge b) \notin O(p_0)$. Si bien $(c_3 \wedge b) \in O(p_3)$, no se puede aplicar la regla (τ) con la fórmula $(c_3 \wedge b)$ dado que ésta no pertenece a $\mathbb{O}(\tau)$. En este caso, el tipo ($!!$) es similar al tipo (η) pero con respecto a los estados que se visitan utilizando transiciones internas luego de la acción visible. Por esta razón, esta noción no diferencia entre p_0 y q_0, i.e. $O_{N_{!!}}(p_0) = O_{N_{!!}}(q_0)$.

2.3. Caracterización relacional

El conjunto de observaciones que genera una interfaz bajo cualquier noción de observabilidad es usualmente infinito. Luego el conjunto de fórmulas de observación que un proceso genera no ofrece un marco adecuado para comparar distintos procesos. Por esta razón introducimos una caracterización relacional para algunas nociones de observabilidad. En otros palabras, dadas dos interfaces S y T y una noción de observabilidad N, se satisface $O_N(S) \subseteq O_N(T)$ si y sólo si existe una relación R_N tal que $S\ \mathsf{R}_N\ T$. Esta relación se llamará N-simulación.

Definición 2.3.1. *Sean $S = \langle Q, s_0, A, \to\rangle$ y $T = \langle Q, t_0, A, \to\rangle$ dos IA y sea N una noción de observabilidad tal que $\bigwedge \in N$. Sea $\hat{N} = N$ si $\rightsquigarrow \notin N$, caso contrario sea $\hat{N} = N - \{\rightleftarrows\}$. Una N-simulación es una relación binaria $\mathsf{R}_N \subseteq Q \times Q$ tal que para cada tipo $\mathsf{t} \in \hat{N}$ la cláusula (t) de la Tabla 2.2 se satisface. Además si $\mathsf{t}, \mathsf{t}', \mathsf{t}'' \in \hat{N}$ son tipos de observaciones tales que ($\mathsf{t}_{\mathsf{t}'}$) o ($\mathsf{t}_{\mathsf{t}',\mathsf{t}''}$) es una cláusula, entonces ésta también debe satisfacerse.*

La necesidad de utilizar $\hat{N}$ en la definición Def. 2.3.1 se debe a que la cláusula $\rightleftarrows$ es más fuerte que la cláusula $\rightsquigarrow$. Dado que $\rightsquigarrow \in N$ implica que $\rightleftarrows \in N$, entonces $\hat{N}$ es necesario para dejar de lado al tipo $\rightleftarrows$. No pasa lo mismo con las cláusulas t y $\mathsf{t}_{\mathsf{t}'}$ dado que $\mathsf{t}_{\mathsf{t}'}$ es tenida en cuenta

(a^O) $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^O \Rightarrow \exists q' : q \Rightarrow \xrightarrow{a} \Rightarrow q', p' \mathrel{\mathsf{R}} q'$

$(a^O_{!!})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^O \Rightarrow \exists q' : q \Rightarrow \xrightarrow{a} q', p' \mathrel{\mathsf{R}} q'$

$(a^O_{\odot})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^O \Rightarrow \exists q' : q \Rightarrow \xrightarrow{a} q', p' \mathrel{\mathsf{R}} q'$

$(\rightleftarrows)$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^I \Rightarrow \exists q' : q \Rightarrow \xrightarrow{a} \Rightarrow q', p' \mathrel{\mathsf{R}} q'$

$(\rightleftarrows_{!!})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^I \Rightarrow \exists q' : q \Rightarrow \xrightarrow{a} q', p' \mathrel{\mathsf{R}} q'$ (igual condición para $(\rightleftarrows_{\odot})$)

$(\rightsquigarrow)$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p' \vee (p \overset{a}{\not\rightarrow} \wedge p' = p), a \in A^I$

$\qquad \Rightarrow \exists q', q'' : ((q \Rightarrow q'' \Rightarrow q' \wedge q'' \overset{a}{\not\rightarrow}) \vee q \Rightarrow \xrightarrow{a} \Rightarrow q'), p' \mathrel{\mathsf{R}} q'$

$(\rightsquigarrow_{!!})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p' \vee (p \overset{a}{\not\rightarrow} \wedge p' = p), a \in A^I$

$\qquad \Rightarrow \exists q' : ((q \Rightarrow q' \overset{a}{\not\rightarrow}) \vee q \Rightarrow \xrightarrow{a} q)', p' \mathrel{\mathsf{R}} q'$ (igual condición para $(\rightsquigarrow_{\odot})$)

(F) $\quad p \mathrel{\mathsf{R}} q, p \overset{a}{\not\rightarrow}, a \in A^I \Rightarrow \exists q' : q \Rightarrow q' \overset{a}{\not\rightarrow}$

(FT) $\quad p \mathrel{\mathsf{R}} q, p \overset{a}{\not\rightarrow}, a \in A^I \Rightarrow \exists q'' : q \Rightarrow q' \overset{a}{\not\rightarrow}, p \mathrel{\mathsf{R}} q'$

(R) $\quad p \mathrel{\mathsf{R}} q, X = I(p) \Rightarrow \exists q' : q \Rightarrow q', I(q') = X$

(RT) $\quad p \mathrel{\mathsf{R}} q, X = I(p) \Rightarrow \exists q' : q \Rightarrow q', I(q') = X, p \mathrel{\mathsf{R}} q'$

(ε) $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^H \Rightarrow \exists q' : q \Rightarrow q', p' \mathrel{\mathsf{R}} q'$

(0) $\quad p \mathrel{\mathsf{R}} q, (\forall a \in A : p \overset{a}{\not\rightarrow}) \Rightarrow \exists q' : q \Rightarrow q', (\forall a \in A : q' \overset{a}{\not\rightarrow})$

(δ) $\quad p \mathrel{\mathsf{R}} q, (\forall a \in A^O \cup A^H : p \overset{a}{\not\rightarrow}) \Rightarrow \exists q' : q \Rightarrow q', (\forall a \in A^O \cup A^H : q' \overset{a}{\not\rightarrow})$

$(\eta_{\rightleftarrows})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^I \Rightarrow \exists q', q'' : q \Rightarrow q'' \xrightarrow{a} \Rightarrow q', p \mathrel{\mathsf{R}} q'', p' \mathrel{\mathsf{R}} q'$

$(\eta_{\rightleftarrows,!!})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^I \Rightarrow \exists q', q'' : q \Rightarrow q'' \xrightarrow{a} q', p \mathrel{\mathsf{R}} q'', p' \mathrel{\mathsf{R}} q'$

$(\eta_{\rightleftarrows,\odot})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^I \Rightarrow \exists q', q'' : q \Rightarrow q'' \xrightarrow{a} q', p \mathrel{\mathsf{R}} q'', p' \mathrel{\mathsf{R}} q'$

$(\eta_{\rightsquigarrow})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p' \vee (p \overset{a}{\not\rightarrow} \wedge p' = p), a \in A^I$

$\qquad \Rightarrow \exists q', q'' : (q \Rightarrow q'' \xrightarrow{a} \Rightarrow q') \vee (q \Rightarrow q'' \Rightarrow q' \wedge q'' \overset{a}{\not\rightarrow}), p \mathrel{\mathsf{R}} q'', p' \mathrel{\mathsf{R}} q'$

$(\eta_{\rightsquigarrow,!!})$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p' \vee (p \overset{a}{\not\rightarrow} \wedge p' = p), a \in A^I$ (igual condición para $(\eta_{\rightsquigarrow,\odot})$)

$\qquad \Rightarrow \exists q', q'' : (q \Rightarrow q'' \xrightarrow{a} q', p \mathrel{\mathsf{R}} q'', p' \mathrel{\mathsf{R}} q') \vee (q \Rightarrow q' \overset{a}{\not\rightarrow}, p \mathrel{\mathsf{R}} q')$

(b) $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{a} p', a \in A^H \Rightarrow \exists q', q'' : q \Rightarrow q'' \wedge (q'' = q' \vee q'' \to q'), p \mathrel{\mathsf{R}} q'', p' \mathrel{\mathsf{R}} q'$

(Δ) $\quad p \mathrel{\mathsf{R}} q, p = p_0 \xrightarrow{\tau} p_1 \xrightarrow{\tau} p_2 \ldots, (\forall i \in \mathbb{N}_0 : p_i \overset{a}{\not\rightarrow}, a \in A^I)$

$\qquad \Rightarrow q = q_0 \xrightarrow{\tau} q_1 \xrightarrow{\tau} q_2 \ldots, (\forall i \in \mathbb{N}_0 : q_i \overset{a}{\not\rightarrow}, a \in A^I)$ (igual condición para (λ))

$(\neg)$ $\quad p \mathrel{\mathsf{R}} q \Rightarrow q \mathrel{\mathsf{R}} p$

$(\odot)$ $\quad p \mathrel{\mathsf{R}} q, p \xrightarrow{\tau} p' \Rightarrow q \xrightarrow{\tau} q', p' \mathrel{\mathsf{R}} q'$

Tabla 2.2.: Caracterización Relacional.

sólo si $\mathsf{t} \in N$. La cláusula ($\mathsf{t}_{\mathsf{t}'}$) es siempre más fuerte que (t), entonces considerar a ambas no produce inconsistencias.

En la Def. 2.3.1, Las cláusulas $a^O, a^O_{!!}$ y $a^O_\odot$ establecen como puede ser imitada una transición de salida para los distintos posibilidades de N. En el caso más simple, $a^O \in N$ entonces la transición puede ser imitada utilizando transiciones internas antes o después de la acción visible. Si $\{a^O, !!\} \subseteq N$, entonces se tiene en cuenta la cláusula ($a^O_{!!}$), la cual establece que no se pueden utilizar transiciones internas después de la acción visible. Si $\{a^O, \odot\} \subseteq N$, entonces se tiene en cuenta la cláusula ($a^O_\odot$), la cual establece que no se pueden utilizar transiciones internas debido a que estas son detectables. Las cláusulas ($\rightleftarrows$), ($\rightleftarrows_{!!}$) y ($\rightleftarrows_\odot$) son análogas pero para las acciones de entrada. Las cláusulas ($\rightsquigarrow$), ($\rightsquigarrow_\varepsilon$) y ($\rightsquigarrow_\odot$) son similares a las últimas pero se le suma la posibilidad de que la acción se imitada por un estado que no realiza la acción. Esto es posible dado que si se ejecuta la acción en ese estado, la acción no es tenida en cuenta por la interfaz. Las cláusulas $(F), (FT), (R)$ y (RT) son directas. El tipo η nos permite diferenciar los estados previos a la ejecución de la acción visible. Por esta razón las variantes de la cláusula (η) piden $p \mathsf{R} q$ y $p \mathsf{R} q''$. Esto garantiza que para todo $\hat{q}$ tal que $q \Rightarrow \hat{q} \Rightarrow q''$ vale $p \mathsf{R} \hat{q}$. El resto de las cláusulas son directas y siguen los lineamientos de [37].

El Teorema 2.3.1 establece que las cláusulas de la Definición 2.3.1 se comportan apropiadamente con las respectivas nociones de observabilidad.

Teorema 2.3.1. *Dado dos IA P y Q y una noción de observabilidad N,*

$$O_N(P) \subseteq O_N(Q) \Leftrightarrow \text{ existe una } N\text{-simulación tal que } P \mathsf{R}_N Q$$

Demostración. Primero demostremos que $O_N(p) \subseteq O_N(q)$ implica que se satisfacen las cláusulas para la relación R_N. Definamos a R_N como $p \mathsf{R}_N q \Leftrightarrow O_N(p) \subseteq O_N(q)$. Supongamos $p \mathsf{R}_N q$ (i.e. $O_N(p) \subseteq O_N(q)$).

$a^O \in N$ Si $p \xrightarrow{a} p'$ y $a \in A^O$ entonces se debe demostrar que existe q' tal que $q \Rightarrow\!\xrightarrow{a}\!\Rightarrow q'$ y $O_N(p') \subseteq O_N(q')$. Supongamos que este no es el caso. Sea $S = \{q' \mid q \Rightarrow\!\xrightarrow{a}\!\Rightarrow q'\}$. Entonces para cada $q' \in S$ existe una formula $\phi_{q'} \in O_N(p') - O_N(q')$. Luego la fórmula $\bigwedge_{q' \in S} \phi_{q'} \in O_N(p') - O_N(q')$ no puede ser generada por ningún estado en S. Por lo tanto $a\varepsilon \bigwedge_{q' \in S} \phi_{q'} \in O_N(p) - O_N(q)$ lo que contradice la hipótesis $O_N(p) \subseteq O_N(q)$.

Las variantes $a^O_{!!}, a^O_\odot$ se demuestran de forma análoga, así como los tipos $\rightleftarrows$, $\rightsquigarrow$, ε, δ, 0, $\odot$ y sus respectivas variantes.

$RT \in N$ Supongamos $I(p) = X$, entonces debemos demostrar que existe q' tal que $q \Rightarrow q'$, $I(q') = X$ y $O_N(p') \subseteq O_N(q')$. Supongamos por el absurdo que este no es el caso. Luego para todo q' tal que $q \Rightarrow q'$, $I(q') \neq X$ o existe $\phi_{q''} \in O_N(p') - O_N(q'')$. Sea $S = \{q' \mid q \Rightarrow q', O_N(p) - O_N(q') \neq \emptyset\}$ Luego $p \models \bigwedge_{q' \in S} \phi_{q'}$ y por RT, $p \models X \bigwedge_{q' \in S} \phi_{q'}$. Por suposición, $q \models X \bigwedge_{q' \in S} \phi_{q'}$, luego por (τ) existe q'' tal que $I(q'') = X$ y $q'' \models \bigwedge_{q' \in S} \phi_{q'}$, pero si $I(q'') = X$ necesariamente $q'' \in S$ y por consiguiente $q'' \not\models \phi_{q''}$ lo cual contradice $q'' \models \bigwedge_{q' \in S} \phi_{q'}$.

Los tipos F, T y FT se demuestran de forma análoga.

$\eta \in N$ Supongamos $p \xrightarrow{a} p', a \in A^I$ debemos demostrar que existen q', q'' tal que $q \Rightarrow q'' \xrightarrow{a} q', O(p)_N \subseteq O_N(q'')$ y $O_N(p') \subseteq O_N(q')$. Supongamos por el absurdo que este no es el caso. Sean $Q'' = \{q'' \mid q \Rightarrow q'' \xrightarrow{a}, \exists \phi_{q''} \in O_N(p) - O_N(q'')\}$ y $Q' = \{q' \mid q \Rightarrow q'' \xrightarrow{a} q', \exists \psi_{q'} \in O_N(p') - O_N(q')\}$. Luego $p \models \bigwedge_{q'' \in Q''} \phi_{q''}(a \bigwedge_{q' \in Q'} \psi_{q'})$ y como $O_N(p) \subseteq O_N(q), q \models \bigwedge_{q'' \in Q''} \phi_{q''}(a \bigwedge_{q' \in Q'} \psi_{q'})$ lo cual llevaría a una contradicción a través de τ, $\bigwedge$ y η.

Las variantes de η y el (b) se demuestran de forma similar.

$\neg \in N$ Supongamos $\phi \notin O_N(p)$, entonces $\neg\phi \in O_N(p)$. $\phi \notin O_N(p)$ y $O_N(p) \subseteq O_N(q)$ implican $\phi \notin O_N(q)$, por lo tanto $\neg\phi \in O_N(q)$.

Finalmente los tipos Δ y λ son directos.

La demostración de que si existe una simulación R_N tal que $p \; \mathsf{R}_N \; q$ luego $O_N(p) \subseteq O_N(q)$ no presenta mayores dificultades. La demostración procede por inducción en la estructura de la fórmula. A continuación la prueba para los casos casos a^O y $\neg$. La prueba para todos los otros tipos de observación sigue el lineamiento del primer caso.

$a^O \in N$ Sea $a\phi \in O(p)$. Luego existe p' tal que $p \Rightarrow \xrightarrow{a} p'$ y $\phi \in O(p')$. Por la cláusula $a^O \in N$ de la relación R_N debe existir q' tal que $q \Rightarrow \xrightarrow{a} \Rightarrow q'$ y $p' \; \mathsf{R}_N \; q'$. Por hipótesis inductiva $\phi \in O(q')$ y por lo tanto $a\phi \in O(q)$.

$\neg \in N$ $\neg\phi \in O(p)$ por lo tanto $\phi \notin O(p)$. Si $\phi \in O(q)$ luego por la cláusula $(\neg)$ y la hipótesis inductiva, $\phi \in O(p)$, absurdo. Por lo tanto $\phi \notin O(q)$ y $\neg\phi \in O(q)$.

□

2.4. Observaciones finales

2.4.1. Trabajos relacionados

En [38] se estudian semánticas para sistemas de transiciones concretos sin transiciones internas. En [37] el trabajo se centra en estudiar semánticas para procesos con acciones internas, divergencia y sub-especificados. En este caso, por estar los procesos sub-especificados, no se define el conjunto de observaciones que genera un proceso sino que cada proceso posee un conjunto de observaciones que se han realizado y un conjunto de observaciones que se podrían realizar. A medida que más observaciones se realizan del sistema estos conjuntos se aproximan al comportamiento concreto del proceso. Además, el enfoque utilizado para generar los comportamientos de cada conjunto está basado en tipos de observación y nociones de observabilidad. En este capítulo se han estudiados semánticas para sistemas de transiciones concretos con acciones de entrada, de salida e internas. Para esto se utilizó el enfoque basado en nociones de observabilidad de [37] el cual permite definir las semánticas (nociones de observabilidad) a partir de suposiciones individuales del modelo (tipos de observaciones).

Por otro lado, parte del trabajo presentado en esta sección está relacionado con el marco desarrollado por la teoría de testing basada en la relación **ioco** (*Input Output COnformace*) [75]. En este contexto, las especificaciones y las implementaciones son modeladas por sistemas de transiciones con acciones de entrada y de salida. Como una implementación no tiene control sobre las acciones de entrada que recibe desde su ambiente estos sistemas deben ser input enabled, i.e. para toda acción de entrada $a?$ y estado s existe un estado s' tal que $s \xrightarrow{a?} s'$. Es natural que una implementación no reaccione en todo momento a todos los posibles estímulos; la falta de reacción ante un estímulo $a?$ se modela definiendo $s' = s$. Esta restricción no está presente en las especificaciones dado que son descripciones del comportamiento deseado. Esto es equivalente a la intuición detrás de los autómatas de interface. Además, en ambos tipos de sistemas se modela la necesidad del estímulo del ambiente para que el sistema progrese. Esto se realiza mediante la inclusión de una transición con una acción especial δ que no es ni de entrada, ni de salida y que indica que se está en presencia de un estado inactivo (*"quiescent state"*). Entonces todo estado s, tal que $s \overset{a}{\not\rightarrow}$ con a una acción de salida o interna, ejecuta una transición visible $s \xrightarrow{\delta} s$. En este contexto, el comportamiento de cada modelo está representado por sus trazas.

Las interfaces de autómatas y las nociones de observabilidad presentadas subsumen los modelos introducidos por esta teoría de testing. El comportamiento que describe la especificación puede describirse con una interface P_s a la cual se le asocia una noción de observabilidad $sp = \{a^O, T, \bigvee, \rightleftarrows, \delta, \tau, \varepsilon\}$. Notemos que la forma de interacción está definida por el tipo de observación $\rightleftarrows$, el cual especifica que las acciones de entrada se ejecutan sólo si están habilitadas en el estado actual. Por otro lado, el comportamiento de la implementación puede modelarse por una interfaz P_i y una noción de observabilidad $im = \{a^O, T, \bigvee, \rightleftarrows, \delta, \tau, \varepsilon, \rightsquigarrow\}$. En este caso la forma de interacción está definida por los tipos de observación $\rightleftarrows$ y $\rightsquigarrow$. Luego todas las acciones de entrada pueden ejecutarse en todo momento pero si una de estas acciones es ejecutada en un estado donde no se encontrase habilitada, ésta será descartada por el sistema. Este enfoque permite separar claramente la parte funcional de los modelos, expresada por los sistemas de transición, de las suposiciones sobre el modelo, expresada por los tipos que componen las nociones de observabilidad.

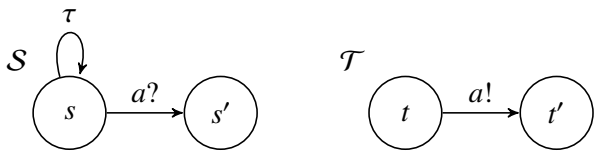
La relación **ioco** establece una relación de *conformidad* entre la la implementación y la especificación, i.e. la implementación cumple con la especificación. La misma establece que la implementación cumple con la especificación si cada comportamiento posible en la implementación, el cual haya sido descripto por la especificación, no produce una acción de salida o δ que no haya sido especificado. Esta condición puede definirse utilizando las observaciones de cada sistema: P_i **ioco** P_s sii[1]

$$\forall \phi \in O_{sp}(P_s) : \phi\phi' \in O_{im}(P_i), \phi' \in \{a\varepsilon : a \in A^O\} \cup \{\delta\} \Rightarrow \phi\phi' \in O_{sp}(P_s)$$

En [75] también se presentan otras variantes conformidad que también pueden ser definidas utilizando los comportamientos observables de P_s y P_i. Pero esto no siempre es así. En [49] se define la noción de conformidad **TGV – ioco**. Esta relación es la relación **ioco** extendida para tener en cuenta divergencia: cuando el sistema diverge es posible ver una acción δ. Si bien en el marco que se ha definido no es posible ver δ sino Δ, esta no es la diferencia sustancial. La

[1] Se omiten los $\top$ al final de las observaciones ϕ y $\phi\phi'$.

diferencia sustancial está en la semántica *unfair* utilizada para definir la divergencia. Formalmente, en [49], un estado s también puede ejecutar $s \xrightarrow{\delta} s$ si existen estados $s_0, s_1, s_2, \ldots$ tales que $s = s_0 \xrightarrow{\tau} s_1 \xrightarrow{\tau} s_2 \xrightarrow{\tau} \ldots$, mientras que para el tipo (Δ) se requiere además que $s_i \xrightarrow{a}$ para todo $i = 0, 1, \ldots$ y toda acción $a \in A^I$. En [49] no se encuentra justificada el porque de la semántica unfair. Por otro lado nuestra decisión está sustentada por la manera en que fue definida la composición de interfaces. Expliquemos esto utilizando las siguientes interfaces



Un proceso diverge si éste no interactúa de ninguna forma con su entorno. Si el estado s divergiese debido a la transición $s \xrightarrow{\tau} s$ entonces la interacción mediante la acción de entrada $a?$ debería ser anulada. Esto no es así, porque en el estado (s, t) del producto $\mathcal{S} \times \mathcal{T}$ los estados s y t sincronizan sin problemas con la acción a. Luego, desde el estado s no es posible divergir.

Una particularidad de las semánticas utilizadas para testing es que nunca se utilizan tipos de observaciones que permitan distinguir particularidades del branching de una ejecución. Esto se debe a que para garantizar conformidad alcanza con verificar que cada ejecución de la implementación está conforme a una ejecución de la especificación. Una relación que si permite distinguir particulares del branching y diferencia entre acciones de entrada y salida es la relación alternating simulation de [25]. Esta relación no puede representarse utilizando lo desarrollado en este capítulo. Recordemos que una alternating simulation R satisface las condiciones de la Def. 2.1.11 excepto la condición (b). Luego si $s \mathsf{R} t$, el estado t debe recibir al menos las mismas entradas que t pero no puede ejecutar ejecutar acciones de salida que no pueda ejecutar s (tal vez utilizando transiciones internas). El vocablo *alternating* refleja la particularidad de esta relación: s debe simular las entradas de t pero t debe simular las acciones de s. Que t puede ejecutar una acción de entrada que s no ejecuta, permite que t puede generar, a partir de ésta, observaciones que no pueden ser generadas desde s. Por lo tanto, no existe noción de observabilidad N tal que si $s \mathsf{R} t$ entonces $O_N(s) \supseteq O_N(T)$. Pero si se agrega la restricción (b) si es posible utilizar las nociones de observabilidad para modelar la relación. El Lema 3.2.1, en el próximo capítulo, establece que una relación SIR puede expresarse utilizando una noción de observabilidad.

2.4.2. Conclusiones

En este capítulo hemos estudiado distintas semánticas para procesos con acciones de entrada y de salida. Hemos aplicado un enfoque basado en tipos de observaciones y nociones de observabilidad. Un aspecto sumamente positivo de este enfoque es que una vez definidos los tipos de observaciones y las nociones de observabilidad el espectro de semánticas queda totalmente definido. La parte fundamental para realizar esto es definir primero las suposiciones que rigen el modelo. Notar que esto fue esencial para justificar la diferencia con los sistemas de transiciones que no diferencian entre los dos tipos de acciones.

Cabe destacar que el conjunto de tipos de observaciones aquí presentados difiere con el conjunto presentado en [37]. Por ejemplo, no hemos presentado distintas versiones de los operadores

para estados estables; tampoco se ha estudiado la posibilidad de observar branching infinitos. Es más, también podrían agregarse variantes de los tipos aquí presentados, entre ellos podemos destacar los tipos F, R, FT y RT pero para acciones de salida. Además es necesario realizar un análisis comparativo de las semánticas el cual establecerá el poder de diferenciación de cada una. La inclusión de los tipos de observaciones más el análisis comparativo estará presente en un futuro trabajo [28].

Finalmente es necesario realizar un estudio más profundo de cómo las semánticas que se han presentado pueden aplicarse y adaptarse a distintos contextos. Con respecto a este punto, en el próximo capítulo se abordará el problema de definir qué es una interfaz segura en un contexto donde éstas manejan información confidencial.

Capítulo 3

No Interferencia en sistemas interactivos

En el Capítulo anterior se estudiaron los comportamiento observables de una interfaz con respecto a un usuario. En esta abstracción se consideró que sólo un usuario interactuaba con la interfaz. En muchos casos esto no es así: un mismo sistema puede tener distintos usuarios y presentar distintas interfaces para cada uno de ellos. Es más, la interacción de un usuario puede generar efectos observables en la interfaz que el sistema le presenta a otro usuario. Así como cada usuario del sistema puede poseer su propia interfaz, el tipo de información que cada uno manipula también puede ser diferente. Una de las posibles diferencias es con respecto al grado de confidencialidad que posee la información. Este capítulo se aboca al estudio de sistemas donde dos tipos distintos de usuarios coexisten, uno de los cuales manipula información confidencial. El objetivo será definir cuando una interfaz es segura, i.e. el sistema no posee filtraciones de información confidencial, y estudiar propiedades sobre este tipo de sistemas.

Una *interfaz para seguridad* (ISS, del inglés *Interface Structure for Security*) [56] es una variante de Autómatas de Interfaces donde las acciones visibles son divididas es dos conjuntos disjuntos. Un tipo se utiliza para modelar las acciones *publicas* o *de baja confidencialidad*, el otro modela las acciones *privadas* o *de alta confidencialidad*. Por simplicidad, estas acciones se denominan *acciones bajas* y *acciones altas*. Las acciones bajas pueden ser accedidas por todo tipo de usuarios, mientras que las acciones altas pueden ser accedidas sólo por los usuarios con los privilegios adecuados. Luego existen dos tipos de usuarios: los *usuarios bajos*, que sólo tienen permiso para acceder a las acciones bajas, los *usuarios altos* que pueden acceder a ambos tipos de acciones.

En este contexto un sistema no posee filtraciones de información confidencial si satisface la propiedad denominada *no interferencia* [41]. Informalmente esta propiedad establece que un usuario bajo no podrá deducir ningún tipo de información o actividad confidencial en base a los siguientes items:

1. El conocimiento de cómo funciona el sistema.

2. Los mecanismo con los que cuenta para interactuar con éste.

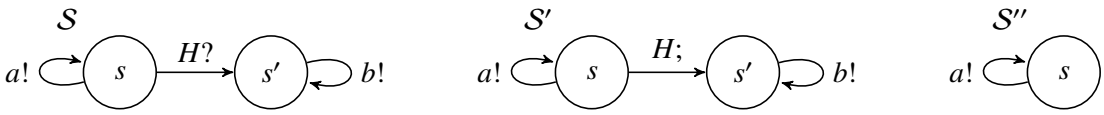


Figura 3.1.: $\mathcal{S}$ es una ISS donde $H?$ es la única acción confidencial. $\mathcal{S}'$ es la interfaz $\mathcal{S}$ con respecto al punto de vista de un usuario bajo. $\mathcal{S}''$ es la interfaz $\mathcal{S}$ sin la interacción del usuario bajo.

El primer ítem supone que no es posible mantener en secreto el funcionamiento del sistema. Esta suposición es análoga a la que se realiza en sistemas criptográfico donde se establece que el algoritmo de encriptación no es secreto, pues esto es algo que no se puede garantizar. El segundo punto establece puntualmente cuales son los mecanismo con que el usuario, o más bien el *atacante*, cuenta para estudiar el sistema. Estos mecanismo definen la semántica a utilizar al momento de definir si una interfaz es segura o no. Otra forma de expresar esta propiedad es que, desde la perspectiva del usuario bajo, la interfaz sin interacción del usuario alto *se comporte igual* que la interfaz con la interacción de éste. En este caso, las dos interfaces *"se comportan igual"* en función de la semántica utilizada.

La propiedad de no interferencia para ISS se presenta en [56] y se corresponde con la introducida en [32] para el contexto de LTS. En este trabajo la semántica utilizada es weak bisimulation (Def. 2.1.5). En [57] se extienden los resultados de [56] para una variante de no interferencia que utiliza la relación SIR (Def. 2.1.11). Finalmente en [58] se presenta una definición de no interferencia que toma como parámetro una noción de observabilidad. Con este nuevo enfoque se generalizan los resultados obtenidos en los trabajos anteriores.

En la Figura 3.1 se grafica una ISS $\mathcal{S}$. La única diferencia con respecto a los diagramas utilizados para representar una IA es la división de las acciones visibles en acciones bajas y altas. En este ejemplo, la interfaz posee dos acciones bajas de salida $a!$ y $b!$ y una acción alta de entrada $H?$. A la derecha de $\mathcal{S}$ se grafica $\mathcal{S}'$, que representa la interfaz $\mathcal{S}$ con respecto a un usuario bajo. Dado que $H?$ es una acción confidencial no puede ser observada por este tipo de usuario, por esta razón el tipo de la acción H cambia de entrada ($H?$) a interna/oculta ($H;$). Por último se grafica $\mathcal{S}''$ que es la interfaz $\mathcal{S}$ sin la interacción del usuario alto. Como no existe este tipo de interacción se puede garantizar que la acción $H?$ nunca va a ser ejecutada, por esta razón la transición $s \xrightarrow{H?} s'$ es eliminada.

La interfaz $\mathcal{S}$ no satisface la propiedad de no interferencia, pues si comparamos el sistema con interacción del usuario alto y sin esta interacción, desde el punto de vista del usuario bajo, i.e. comparamos $\mathcal{S}'$ con $\mathcal{S}''$, se obtiene que los sistemas no presentan los mismos comportamientos. En este ejemplo particular, para toda noción de observabilidad N, $O_N(\mathcal{S}') \neq O_N(\mathcal{S}'')$. Esto se debe que a que si un usuario bajo ve la ejecución de la acción $b!$ entonces puede inferir, bajo la suposición de que conoce el sistema, que un usuario alto ejecutó una acción $H?$. Por lo tanto se produce una filtración de información confidencial.

La primera parte de este Capítulo se centra en la variante de no interferencia que toma como parámetro una noción de observabilidad. Este enfoque provee un marco de trabajo que permite demostrar ciertos resultados de la propiedad de forma general. Utilizando los resultados del

Capítulo 2, se extienden los resultados de [58]. Luego se centra en la variante de no interferencia basada en la relación SIR [57]. Se presentan condiciones suficientes para garantizar que la composición de interfaces seguras es segura. Además se presentan dos algoritmos. El primero determina si una interfaz satisface la propiedad de no interferencia basada en la relación SIR. El segundo determina si una interfaz que no satisface la propiedad puede transformase en una que sí la satisface mediante la restricción de ciertas acciones de entrada, si esto es así, entonces el algoritmo sintetiza una ISS segura. Ambos algoritmos son polinomiales en el número de estados de la interfaz bajo estudio.

3.1. No interferencia basada en nociones de observabilidad

El modelo con el que se trabajará es una IA donde las acciones visibles están divididas en dos conjuntos disjuntos: las *acciones altas* y las *acciones bajas*.

Definición 3.1.1. *Una* interfaz para seguridad *(ISS, del inglés* Interface Structure for Security*), es una tupla* $\mathcal{S} = \langle P, A^h, A^l \rangle$ *donde* $P = \langle S, p_0, A, \rightarrow \rangle$ *es un IA y* A^h *y* A^l *son conjuntos disjuntos tales que* $A^h \cup A^l = A^O \cup A^I$*. Denotaremos con* $A^{X,m}$ *al conjunto* $A^X \cap A^m$ *donde* $X \in \{I, O\}$ *y* $m \in \{h, l\}$*.*

Extender la definición de composición de autómatas de interfaces a este contexto es directo.

Definición 3.1.2. *Sean* $\mathcal{S} = \langle S, A_S^h, A_S^l \rangle$ *y* $\mathcal{T} = \langle T, A_T^h, A_T^l \rangle$ *dos ISS. Los ISS* $\mathcal{S}$ *y* $\mathcal{T}$ *son* componibles *si* S *y* T *lo son. Dado dos ISS componibles* $\mathcal{S}$ *y* $\mathcal{T}$*, su* composición $\mathcal{S} \parallel \mathcal{T}$ *está definida por* $\langle S \parallel T, (A_S^h \cup A_T^h) - \mathit{shared}(S, T), (A_S^l \cup A_T^l) - \mathit{shared}(S, T) \rangle$*.*

Definido el modelo, podemos abocarnos a la formalización de no interferencia. Informalmente, esta propiedad establece que un usuario bajo no debe poder deducir ningún tipo de información o actividad sólo interactuando a través de las acciones de su nivel. Por lo tanto se esperaría que, desde el punto de vista del usuario bajo, el sistema con la interacción alta oculta se comporte de igual manera que el sistema sin este tipo de interacción. De esto se desprende que para formalizar la propiedad de seguridad necesitaremos un operador de ocultamiento y un operador de restricción.

Definición 3.1.3. *Dado un IA* S *y un conjunto de acciones* $X \subseteq A^I \cup A^O$*, definimos:*

- *la* restricción *de* X *en* S *como*

  $$S \backslash X = \langle S, A^I - X, A^O - X, A^H, \rightarrow_{P \backslash X} \rangle \text{ donde } q \xrightarrow{a}_{S \backslash X} q' \text{ sii } q \xrightarrow{a} q' \text{ y } a \notin X.$$

- *el* ocultamiento *de* X *en* S *como*

  $$S / X = \langle S, A^I - X, A^O - X, A^H \cup X, \rightarrow \rangle.$$

Dado un ISS $\mathcal{S} = \langle S, A^h, A^l \rangle$ *definimos la* restricción *de* X *en* $\mathcal{S}$ *cómo* $\mathcal{S} \backslash X = \langle S \backslash X, A^h - X, A^l - X \rangle$ *y el* ocultamiento *de* X *en* $\mathcal{S}$ *cómo* $\mathcal{S} / X = \langle S / X, A^h - X, A^l - X \rangle$*.*

Ahora estamos en condiciones de definir no interferencia basados en una noción de observabilidad. Esta definición se basa en la presentada en [32].

Definición 3.1.4. *Sea $\mathcal{S} = \langle S, s_0, A^h, A^l \rangle$ una ISS y V una noción de observabilidad, entonces:*

- *$\mathcal{S}$ es V* strong non-deterministic non-interference *(V-SNNI) si $O_V(\mathcal{S}/A^h) = O_V(\mathcal{S}\backslash A^h)$.*
- *$\mathcal{S}$ es V* non-deterministic non-interference *(V-NNI) si $O_V(\mathcal{S}/A^h) = O_V((\mathcal{S}\backslash A^{h,I})/A^{h,O})$.*

Notemos la diferencia entre ambas definiciones. V-SNNI formaliza la propiedad de seguridad que describimos con anterioridad: un sistema satisface V-SNNI si un usuario bajo no distingue utilizando las acciones de su nivel (las únicas visibles en los procesos resultantes) si el proceso realiza transiciones con acciones altas (las mismas están ocultas) o no (las mismas están restringidas). Por otro lado, en la definición de V-NNI sólo las acciones altas de entradas son restringidas debido a que estas son las únicas sobre las cuales el usuario alto tiene control; las acciones altas de salida pueden ocurrir autónomamente, luego no pueden ser restringidas y por lo tanto se ocultan. Esta versión es más adecuada para el contexto de IA debido a que sólo las acciones altas son controlables por el entorno. Para diferenciar entre las dos variantes de no interferencia, se denominará a la primera *variante strong* y a la segunda *variante simple*.

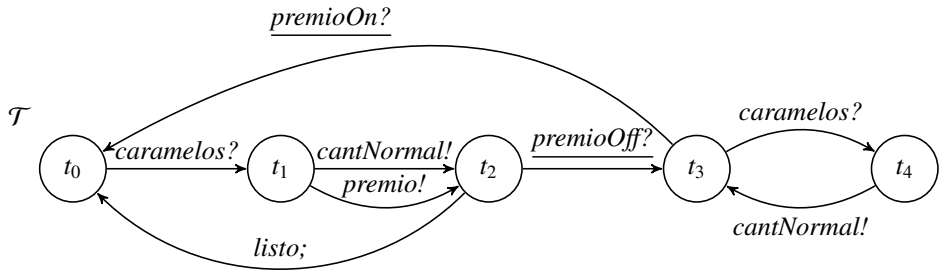


Figura 3.2.: $\mathcal{T}$ representa una expendedora de caramelos configurable

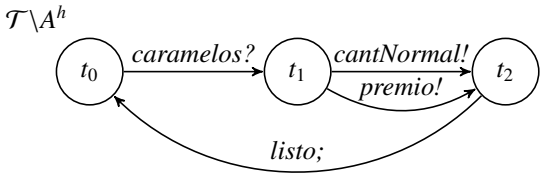


Figura 3.3.: El ISS $\mathcal{T}$ de la Figura 3.2 con las acciones altas restringidas

Ejemplo 3.1.1. *El la Figura 3.2 se grafica un ISS $\mathcal{T}$ que representa una máquina expendedora de caramelos. La misma se puede configurar para que otorgue o no premios. Las acciones altas se encuentran subrayadas, mientras que las bajas no. Es este sistema el usuario alto es un administrador el cual puede activar o desactivar la posibilidad de que la máquina otorgue*

premios. El usuario bajo es el usuario de la máquina. En este ejemplo, se desea verificar que el usuario no puede detectar si la máquina se encuentra con la opción de premios desactivada.

El sistema $\mathcal{T}/A^h$ es exactamente el sistema $\mathcal{T}$ graficado en la Figura 3.2 donde las acciones altas son tratadas como acciones internas. El sistema $\mathcal{T}\backslash A^h$ es el sistema $\mathcal{T}$ sin la interacción alta; este sistema se grafica en la Figura 3.3. Notar que no hay diferencia entre $(\mathcal{T}\backslash A^{h,I})/A^{h,O}$ y $\mathcal{T}\backslash A^h$ dado que $\mathcal{T}$ no posee acciones confidenciales de salida.

El ISS $\mathcal{T}$ será V-SNNI (o V-NNI) dependiendo de la definición de la noción de observabilidad V. Por ejemplo:

1. *Si $V_1 = \{a^O, T, \bigvee, \rightleftarrows, FT\}$ entonces $O_{V_1}(\mathcal{T}/A^h) = O_{V_1}(\mathcal{T}\backslash A^h)$.*
2. *Si $V_2 = \{a^O, T, \bigvee, \rightleftarrows, \odot\}$ entonces $O_{V_2}(\mathcal{T}/A^h) \neq O_{V_2}(\mathcal{T}\backslash A^h)$ dado que*

   $$caramelos?cantNormal!\odot caramelos?cantNormal!caramelos? \in O_{V_2}(\mathcal{T}/A^h) - O_{V_2}(\mathcal{T}\backslash A^h)$$

3. *Si $V_3 = \{a^O, T, \bigvee, \rightleftarrows, FT, \bigwedge\}$ entonces $O_{V_3}(\mathcal{T}/A^h) = O_{V_3}(\mathcal{T}\backslash A^h)$.*
4. *Si $V_4 = \{a^O, T, \bigvee, \rightleftarrows, FT, \bigwedge, \neg\}$ entonces $O_{V_4}(\mathcal{T}/A^h) \neq O_{V_4}(\mathcal{T}\backslash A^h)$. Utilizando la caracterización relacional de la noción de observabilidad de V_4, el estado t_4 de $\mathcal{T}/A^h$ no puede ser relacionado con un estado de $\mathcal{T}\backslash A^h$.*

Cuando la máquina $\mathcal{T}$ está configurada para no entregar premios, no entrega premios y no realiza una transición interna entre la entrega de caramelos y el estar lista para un nuevo pedido. La noción de observabilidad V_4 permite detectar lo primero utilizando el tipo de observabilidad $\neg$, pues no existe un comportamiento simétrico: un estado entrega premios, el otro no. La noción de observabilidad V_2 permite detectar lo segundo mediante el tipo de observabilidad $\odot$, el cual permite detectar la ejecución de acciones internas. Por estas razones el sistema es inseguro para V_2 y V_4. En cambio, el sistema es seguro para V_1 y V_3 debido a que los tipos de observaciones $\neg$ y $\odot$ no están presentes y por lo tanto no es posible detectar las particularidades del sistema cuando está configurado para no entregar premios.

El enfoque basado en nociones de observabilidad permite estudiar de forma general la relación entre la variante strong y la variante simple de no interferencia. La primera relación establece que para toda noción de observabilidad V la propiedad V-NNI no es más fuerte que V-SNNI.

Teorema 3.1.1. *Para toda noción de observabilidad V existe un ISS $\mathcal{S}$ que es V-NNI pero no es V-SNNI.*

Demostración. Sea $\mathcal{S}$ el ISS $s_0 \xrightarrow{H!} s' \xrightarrow{a!} s''$ con a una acción baja y H una acción confidencial. Como $S/A^h = (S\backslash A^{h,I})/A^{h,O}$ entonces $\mathcal{S}$ es V-NNI para todo V. Por otro lado $\mathcal{S}$ no es V-SNNI para todo V: $a!\varepsilon\top \in O_V(\mathcal{S}/A^h)$ y $a!\varepsilon\top \notin O_V(\mathcal{S}\backslash A^h)$. □

Este resultado no es una novedad pues en [32] se demuestra que la variante strong es más fuerte que la variante simple para el caso en el que se comparan los sistemas utilizando equivalencia de trazas. Dado que la semántica de trazas es la semántica que menos distingue entonces es natural que la propiedad valga para toda otra noción de observabilidad más discriminatoria. El Teorema 3.1.1 sólo formaliza esto para toda posible semántica que se puede definir para una ISS utilizando lo desarrollado en el Capítulo 2.

Por otro lado, si una ISS $\mathcal{S}$ es V-SNNI sólo para algunos casos se satisface $\mathcal{S}$ es V-NNI. Esto dependerá de cómo V está constituido. Los siguientes dos teoremas formalizan estas relaciones.

Para realizar la demostración del primer Teorema se necesita un lema auxiliar. Este lema garantiza que si a una IA se le eliminan algunas transiciones internas, la nueva IA no presentará nuevos comportamientos con respecto a una noción de observabilidad V que no puede distinguir si un estado es estable (no puede ejecutar acciones autónomas) o final (no puede ejecutar acciones), es decir $V \cap \{0, \neg, \delta, \lambda\} = \emptyset$. Este resultado se basa en que eliminar transiciones internas sólo puede crear este tipo de estados y sin los tipos $0, \neg, \delta$ y λ la aparición de estos nuevos estados no puede ser observada.

Lema 3.1.1. *Sea S un ILTS y sea V una noción de observabilidad tal que $\{0, \neg, \delta, \lambda\} \cap V = \emptyset$. Sea S' el ILTS que se obtiene de S luego de eliminar de éste algunas transiciones internas. Entonces, $O_V(S') \subseteq O_V(S)$.*

Demostración. La prueba es directa por inducción en la estructura de la fórmula ϕ. □

Teorema 3.1.2. *Sea S una ISS y sea V una noción de observabilidad tal que $\{0, \neg, \delta, \lambda\} \cap V = \emptyset$. Si $\mathcal{S}$ es V-SNNI entonces $\mathcal{S}$ es V-NNI.*

Demostración. Si $\mathcal{S}$ es V-SNNI entonces $O_V(S/A^h) = O_V(S \setminus A^h)$. Notemos $S \setminus A^h$ se obtiene eliminando algunas transiciones internas de $(S \setminus A^{h,I})/A^{h,O}$, entonces por el Lema 3.1.1:

$$O_V((S \setminus A^{h,I})/A^{h,O}) \supseteq O_V(S \setminus A^h)$$

Por otro lado, $(S \setminus A^{h,I})/A^{h,O}$ se obtiene eliminando algunas transiciones internas de S/A^h, por el Lema 3.1.1:

$$O_V(S/A^h) \supseteq O_V((S \setminus A^{h,I})/A^{h,O})$$

Ambas inclusiones implican $O_V(S/A^h) = O_V((S \setminus A^{h,I})/A^{h,O})$, i.e. $\mathcal{S}$ es V-NNI. □

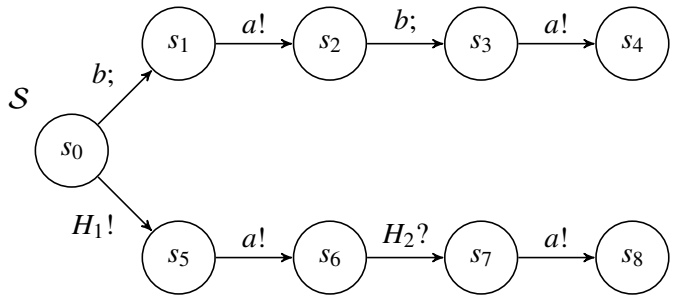


Figura 3.4.: $\mathcal{S}$ es V-SNNI pero no es V-NNI si $V \cap \{0, \neg, \delta, \lambda\} \neq \emptyset$

Teorema 3.1.3. *Para toda noción de observabilidad V tal que $V \cap \{0, \neg, \delta, \lambda\} \neq \emptyset$ existe una ISS $\mathcal{S}$ tal que $\mathcal{S}$ es V-SNNI pero no es V-NNI.*

Demostración. Sea $\mathcal{S}$ el ISS de la Figura 3.4 donde H_1 y H_2 son las únicas acciones altas. $\mathcal{S}$ es V-SNNI para todo V tal que $V \cap \{0, \neg, \delta, \lambda\} \neq \emptyset$. Demostremos que no es V-NNI con $\odot \notin V$:

- si $\neg \in V$ entonces $a\varepsilon\neg(a\varepsilon\top) \in O_V((S \setminus A^{h,I})/A^{h,O})$ mientras que $a\varepsilon\neg(a\varepsilon\top) \notin O_V(S \setminus A^h)$;
- si $0 \in V$ entonces $a\varepsilon 0 \in O_V((S \setminus A^{h,I})/A^{h,O})$ mientras que $a\varepsilon 0 \notin O_V(S \setminus A^h)$;
- si $\delta \in V$ entonces $a\varepsilon\delta\top \in O_V((S \setminus A^{h,I})/A^{h,O})$ mientras que $a\varepsilon\delta\top \notin O_V(S \setminus A^h)$.
- si $\lambda \in V$ entonces $a\varepsilon\lambda \in O_V((S \setminus A^{h,I})/A^{h,O})$ mientras que $a\varepsilon\lambda \notin O_V(S \setminus A^h)$.

Entonces $\mathcal{S}$ no es V-NNI para todo V tal que $V \cap \{0, \neg, \delta, \lambda\} \neq \emptyset$. Los casos para $\odot \in V$ son directos. □

Por último, el enfoque basado en nociones de observabilidad permite demostrar que las propiedades de seguridad no son preservadas por la composición.

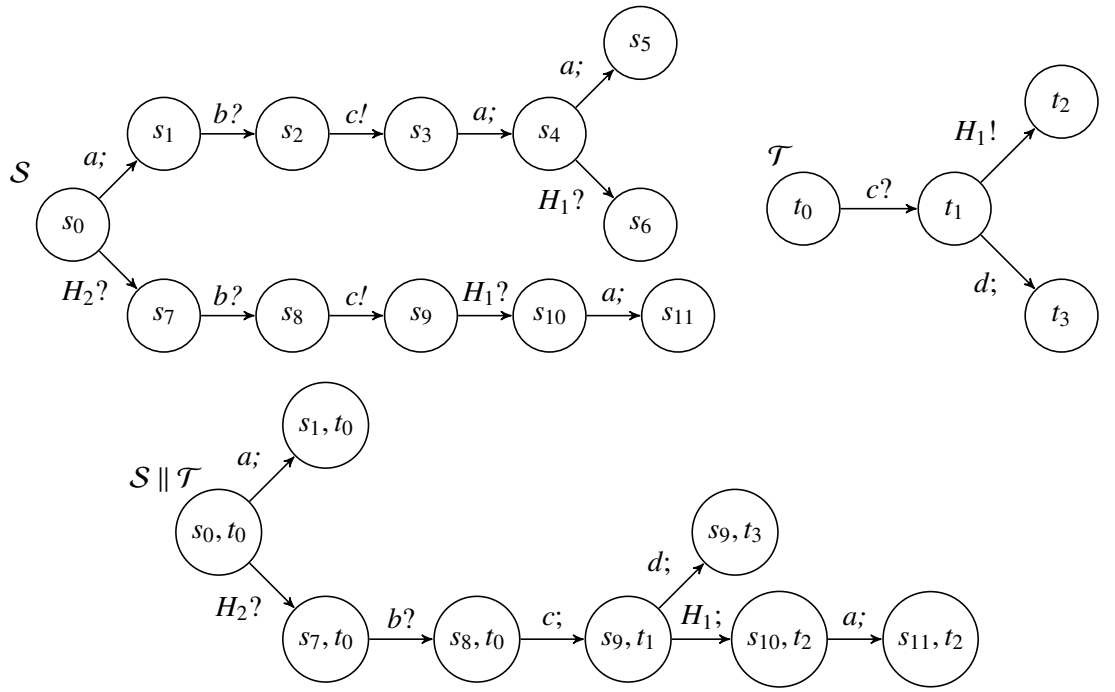


Figura 3.5.: $\mathcal{S}$ y $\mathcal{T}$ son dos ISS V-NNI/V-SNNI.

Teorema 3.1.4. *Para toda noción de observabilidad V existen dos ISS $\mathcal{S}$ y $\mathcal{T}$ tal que $\mathcal{S}$ y $\mathcal{T}$ son V-SNNI (resp. V-NNI) y componibles, pero la composición $\mathcal{S} \parallel \mathcal{T}$ no es V-SNNI (resp. V-NNI).*

Demostración. En la Figura 3.5 se grafican dos ISS $\mathcal{S}$ y $\mathcal{T}$ componibles y su composición $\mathcal{S} \parallel \mathcal{T}$. Las únicas acciones confidenciales son H_1 y H_2.

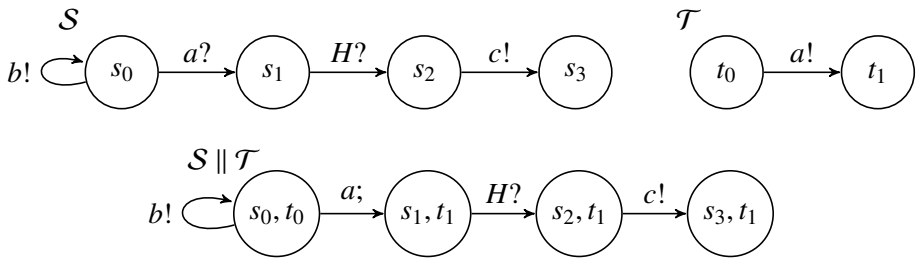


Figura 3.6.: $\mathcal{S}$ y $\mathcal{T}$ son dos ISS V-NNI/V-SNNI si $\rightleftarrows \notin V$. Su composición no lo es.

Para toda noción de observabilidad V tal que $\rightleftarrows \in V$, $\mathcal{S}$ y $\mathcal{T}$ son V-SNNI pero su composición $\mathcal{S} \parallel \mathcal{T}$ no lo es:

$$b?\varepsilon\top \in O_V((\mathcal{S} \parallel \mathcal{T})/A^h) - O_V((\mathcal{S} \parallel \mathcal{T})\backslash A^h)$$

Notar que la suposición $\rightleftarrows \in V$ es necesaria para garantizar que la acción $b?$ es observable en $\mathcal{S} \parallel \mathcal{T}$. La observación de esta acción produce la filtración de información confidencial.

Supongamos $\rightleftarrows \notin V$. En la Figura 3.6 se grafican dos ISS $\mathcal{S}$ y $\mathcal{T}$ que son V-SNNI pero su composición $\mathcal{S} \parallel \mathcal{T}$ no lo es. Dado que $\rightleftarrows \notin V$ la acción $a?$ no es tenida en cuenta, luego $O_V(\mathcal{S}/A^h) = O_V(\mathcal{S}\backslash A^h)$. Por otro lado $\mathcal{T}$ no posee acciones confidenciales entonces $O_V(\mathcal{T}/A^h) = O_V(\mathcal{T}\backslash A^h)$. A pesar de esto la composición no es segura pues

$$b?\varepsilon\top \in O_V((\mathcal{S} \parallel \mathcal{T})/A^h) - O_V((\mathcal{S} \parallel \mathcal{T})\backslash A^h)$$

Notar que en ambos casos $(\mathcal{S} \parallel \mathcal{T})\backslash A^h = ((\mathcal{S} \parallel \mathcal{T})\backslash A^{h,I})/A^{h,O}$ con lo que el contraejemplo se aplica directamente a V-NNI.

□

3.2. Comparación con otras definiciones de no interferencia

Si bien existen muchas variantes de no-interferencia sobre muchos modelos, en el contexto de interfaces de autómatas se han presentado sólo las siguientes variantes: en [56] se presentan *Bisimulation NNI* (BNNI) y *Bisimulation SNNI* (BSNNI); mientras que en [57] se presentan SIR-NNI y SIR-SNNI. Las primeras dos nociones están basadas en bisimulación débil, las segundas dos están basadas en la relación SIR (Def. 2.1.11).

Definición 3.2.1. *Sea $\mathcal{S}$ una ISS, entonces:*

- *$\mathcal{S}$ es BSNNI (*bisimulation strong non-deterministic non-interference*) si $\mathcal{S}/A^h \sim \mathcal{S}\backslash A^h$.*
- *$\mathcal{S}$ es BNNI (*bisimulation non-deterministic non-interference*) si $\mathcal{S}/A^h \sim (\mathcal{S}\backslash A^{h,I})/A^{h,O}$.*
- *$\mathcal{S}$ es SIR-SNNI(*SIR strong non-deterministic non-interference*) si $\mathcal{S}\backslash A^h \succcurlyeq \mathcal{S}/A^h$.*
- *$\mathcal{S}$ es SIR-NNI(*SIR non-deterministic non-interference*) si $(\mathcal{S}\backslash A^{h,I})/A^{h,O} \succcurlyeq \mathcal{S}/A^h$.*

El Lema 2.2.1 garantiza que la bisimulación débil puede ser representada por la noción de observabilidad $\{a^O, \top, \tau, \varepsilon, \vee, \rightleftarrows, \varepsilon, \wedge, \neg\}$, luego:

- $\mathcal{S}$ es BNNI (resp. BSNNI) si y sólo si $\mathcal{S}$ es *WB*-NNI (resp. *WB*-SNNI) con

$$WB = \{a^O, \top, \tau, \varepsilon, \vee, \rightleftarrows, \varepsilon, \wedge, \neg\}$$

Las nociones de seguridad SIR-SNNI y SIR-NNI también pueden representarse utilizando el enfoque basado en nociones de observabilidad. Esto se demuestra en el Lema 3.2.2, el cual utiliza a su vez el siguiente Lema (su demostración en el Apéndice A.1).

Lema 3.2.1. *Sean S y T dos IA y sea $V = \{a^O, \top, \tau, \varepsilon, \vee, \rightleftarrows, RT, \bigwedge, \eta, b\}$ una noción de observabilidad. Entonces $S \succcurlyeq T$ sii $O_V(S) \supseteq O_V(T)$.*

Lema 3.2.2. *Sea $V = \{a^O, \top, \tau, \varepsilon, \vee, \rightleftarrows, RT, \bigwedge, \eta, b\}$ una noción de observabilidad. Una ISS $\mathcal{S}$ es SIR-SNNI (resp. SIR-NNI) si y sólo si $\mathcal{S}$ es V-SNNI (resp. V-NNI)*

Demostración. Supongamos que $\mathcal{S}$ es SIR-SNNI. Por Lema 3.2.1

$$\mathcal{S}\backslash A^h \succcurlyeq \mathcal{S}/A^h \quad \text{sii} \quad O_V(\mathcal{S}\backslash A^h) \supseteq O_V(\mathcal{S}/A^h)$$

Por Lema 3.1.1, $O_V(\mathcal{S}\backslash A^h) \subseteq O_V(\mathcal{S}/A^h)$, luego $\mathcal{S}$ es V-SNNI. La vuelta es similar. □

A partir de ahora se denominará SIR a la noción de observabilidad $\{a^O, \top, \tau, \varepsilon, \vee, \rightleftarrows, RT, \bigwedge, \eta, b\}$

3.3. Composición de sistemas seguros

El Teorema 3.1.4 demuestra que para toda noción de observabilidad V las propiedades de no interferencia no son preservadas por la composición. Esto es problemático porque a medida que un sistema suma más componentes resulta más difícil verificar si el sistema satisface o no la propiedad. Para atacar este problema, en esta sección presentamos condiciones suficientes para garantizar que ciertas variantes de no interferencia son preservadas por la composición. Primero nos enfocaremos en las propiedades BSNNI y BNNI [56]. Luego en las propiedades SIR-NNI y SIR-SNNI. Cabe destacar que el resultado que se presentará para SIR-NNI y SIR-SNNI difiere del presentado en [57] pues requiere menos restricciones.

Las condiciones que se requieren para garantizar que las propiedad BSNNI/BNNI es preservada por la composición son:

1. las componentes del proceso resultante son *totalmente compatibles*, i.e. ningún estado de error es alcanzable en la composición (aquí no nos referimos solamente a las ejecuciones autónomas), y

2. los procesos no utilizan acciones altas para sincronizar.

Desde el punto de vista de la construcción del software, el punto 1 resulta razonable, pero el punto 2 resulta muy restrictivo. Por el momento nos enfocaremos en demostrar este resultado, pero luego se estudiará cómo adaptar la condición 2 para que resulte más razonable.

Las condiciones tienen por objetivo el poder definir la relación entre los estados del sistema compuesto, con y sin interacción alta, en base a las relaciones definidas en cada componente. Para entender el porqué de estas condiciones, observemos primero que las transiciones internas son de suma importancia para definir la bisimulación débil que requiere la propiedad BSNNI o BNNI. Segundo, como consecuencia de las abstracciones de las acciones confidenciales las semánticas no diferencian entre:

(a) una transición con una acción común y confidencial, la cual es oculta debido a la abstracción de las acciones altas y

(b) una transición con una acción interna o; una acción no común y confidencial que fue ocultada debido a la abstracción de las acciones altas.

Que los procesos sean totalmente compatibles garantiza que la única forma de perder transiciones en la composición es que una componente espere una acción de entrada común y que la otra componente no la ejecute. Por lo tanto, una transición que satisface (*a*) podría no estar presente en la composición, mientras que toda transición que satisface (*b*) siempre lo estará. El segundo requerimiento implica que no existen transiciones que satisfacen (*a*), entonces todas las transiciones internas que se utilizan para definir la relación de bisimulación débil en cada componente estarán presentes en la composición.

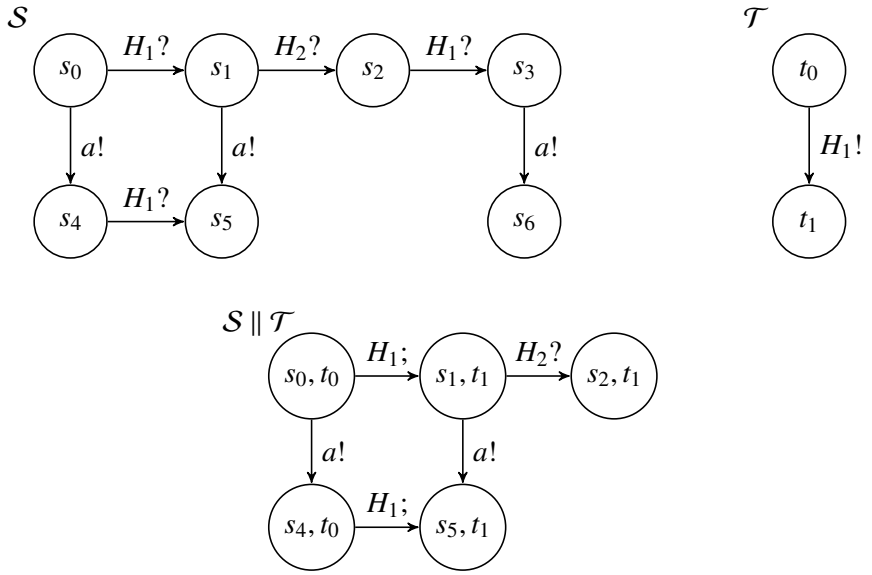


Figura 3.7.: $\mathcal{S}$ y $\mathcal{T}$ son dos ISS BSNNI/BNNI componibles que sincronizan usando acciones confidenciales. A pesar de que el producto $\mathcal{S} \times \mathcal{T}$ no alcanza estados de error, la composición no es BSNNI/BNNI.

En la Figura 3.7 se grafican dos interfaces $\mathcal{S}$ y $\mathcal{T}$. La única acción baja en los sistemas es la acción $a!$, por lo tanto son BSNNI/BNNI. Las interfaces no satisfacen la condición 2. La

interfaz $\mathcal{S} \parallel \mathcal{T}$ no es BSNNI/BNNI; si lo fuese debería satisfacerse $(s_0, t_0) \sim (s_2, t_1)$, donde (s_0, t_0) pertenece a la ISS $(\mathcal{S} \parallel \mathcal{T})\backslash\{H_2\}$ y (s_2, t_1) pertenece a la ISS $(\mathcal{S} \parallel \mathcal{T})/\{H_2\}$. Esto no es así debido a que el estado (s_0, t_0) puede ejecutar una acción $a!$ y (s_2, t_1) no puede imitar esta acción (notar que $s_2 \xrightarrow{H_1?} s_3 \xrightarrow{a!} s_6$ pero $(s_2, t_1) \overset{a!}{\not\Rightarrow}$ debido a que $t_1 \overset{H_1!}{\not\Rightarrow}$). Este tipo de situaciones impide definir la relación en el sistema compuesto, utilizando las relaciones en cada componente.

Teorema 3.3.1. *Sean $\mathcal{S} = \langle S, A_S^h, A_S^l \rangle$ y $\mathcal{T} = \langle T, A_T^h, A_T^l \rangle$ dos ISS componibles tales que no existe una acción común confidencial, i.e. shared$(\mathcal{S}, \mathcal{T}) \cap (A_S^h \cup A_T^h) = \emptyset$. Sea $S \otimes T$ tal que no existe un estado de error alcanzable desde (s_0, t_0). Si $\mathcal{S}$ y $\mathcal{T}$ son BSNNI (resp. BNNI) entonces $\mathcal{S} \parallel \mathcal{T}$ es BSNNI (resp. BNNI).*

Demostración. Sea $\mathsf{R}_\mathcal{S}$ y $\mathsf{R}_\mathcal{T}$ la bisimulación entre $\mathcal{S}\backslash A_S^h$ y $\mathcal{S}/A_S^h$ y $\mathcal{T}\backslash A_T^h$ y $\mathcal{T}/A_T^h$ respectivamente. Sólo se demostrará la prueba para el caso BSNNI, el caso BNNI es análogo. Definamos la relación R como $(s_r, t_r)\ \mathsf{R}\ (s_a, t_a)$ sii $s_r\ \mathsf{R}_\mathcal{S}\ s_a$ y $t_r\ \mathsf{R}_\mathcal{T}\ t_a$. Demostremos que R es una bisimulación entre $(\mathcal{S} \parallel \mathcal{T})\backslash A^h$ y $(\mathcal{S} \parallel \mathcal{T})/A^h$ donde $A^h = (A_S^h \cup A_T^h) - \text{shared}(S, T) = A_S^h \cup A_T^h$. Supongamos $(s_r, t_r)\ \mathsf{R}\ (s_a, t_a)$ y demostremos que la propiedad de transferencia se satisface para los distintos casos.

Sea $(s_r, t_r) \xrightarrow{a} (s_r', t_r)$. Como $(s_r, t_r)\ \mathsf{R}\ (s_a, t_a)$ entonces $s_r\ \mathsf{R}_\mathcal{S}\ s_a$ y por lo tanto existe s_a' tal que $s_a \overset{\hat{a}}{\Rightarrow} s_a'$ con $s_r'\ \mathsf{R}_\mathcal{S}\ s_a'$. Si a es una acción visible, entonces existen estados s_a'' y s_a''' tales que $s_a \Rightarrow s_a'' \xrightarrow{a} s_a''' \Rightarrow s_a'$. Que no se utilicen acciones confidenciales para sincronizar y que no existan estados de error garantiza que $(s_a, t_a) \Rightarrow (s_a'', t_a)$ y $(s_a''', t_a) \Rightarrow (s_a', t_a)$. Que no existan estados de error garantiza que $(s_a'', t_a) \xrightarrow{a} (s_a''', t_a)$. Por lo tanto $(s_a, t_a) \overset{\hat{a}}{\Rightarrow} (s_a', t_a)$ y $(s_r', t_r)\ \mathsf{R}\ (s_a', t_a)$. Si a es una acción interna entonces $s_a \Rightarrow s_a'$; repitiendo el razonamiento anterior se concluye que $(s_a, t_a) \Rightarrow (s_a', t_a)$. Una vez más $(s_a, t_a) \overset{\hat{a}}{\Rightarrow} (s_a', t_a)$ y $(s_r', t_r)\ \mathsf{R}\ (s_a', t_a)$. Los casos $(s_r, t_r) \xrightarrow{a} (s_r, t_r')$, $(s_a, t_a) \xrightarrow{a} (s_a', t_a)$ y $(s_a, t_a) \xrightarrow{a} (s_a, t_a')$ son análogos.

Sea $(s_r, t_r) \xrightarrow{a;} (s_r', t_r')$. Sin pérdida de generalidad supongamos $s_r \xrightarrow{a?} s_r'$ y $t_r \xrightarrow{a!} t_r'$. Como $(s_r, t_r)\ \mathsf{R}\ (s_a, t_a)$ entonces $s_r\ \mathsf{R}_\mathcal{S}\ s_a$, $t_r\ \mathsf{R}_\mathcal{T}\ t_a$ y por lo tanto existen $s_a', s_a'', s_a''', t_a', t_a''$ y t_a''', tales que $s_a \Rightarrow s_a'' \xrightarrow{a?} s_a''' \Rightarrow s_a'$ y $t_a \Rightarrow t_a'' \xrightarrow{a!} t_a''' \Rightarrow t_a'$. Que no se utilicen acciones confidenciales para sincronizar y que no existan estados de error garantiza que en la composición se cumple que

$$(s_a, t_a) \Rightarrow (s_a'', t_a) \Rightarrow (s_a'', t_a'') \xrightarrow{a;} (s_a''', t_a''') \Rightarrow (s_a', t_a''') \Rightarrow (s_a', t_a')$$

Luego $(s_a, t_a) \Rightarrow (s_a', t_a')$ y $(s_r', t_r')\ \mathsf{R}\ (s_a', t_a')$. La demostración del caso $(s_a, t_a) \xrightarrow{a;} (s_a', t_a')$ es análoga. □

Este resultado es útil cuando uno desarrolla todos los componentes de un sistema. Como es posible diseñar cada componente, es posible garantizar una compatibilidad total entre los mismos. De esta forma, para garantizar que el sistema total es seguro basta con desarrollar subsistemas seguros que no se comuniquen con acciones confidenciales.

En caso de que la condición 2 no se satisfaga, el resultado se puede adaptar de la siguiente manera: dado dos procesos que no satisfacen la segunda condición, definimos dos nuevos ISS donde las acciones altas que se utilizan para sincronizar son definidas como bajas. Luego sólo se

debe verificar si los nuevos procesos satisfacen las hipótesis del Teorema 3.3.1 para garantizar que el sistema compuesto es seguro.

Corolario 3.3.1. *Sean* $\mathcal{S} = \langle S, A_S^h, A_S^l \rangle$ *y* $\mathcal{T} = \langle T, A_T^h, A_T^l \rangle$ *dos ISS componibles. Sean* $\mathcal{S}' = \langle S, A_S^h - \mathit{shared}(S,T), A_S^l \cup \mathit{shared}(S,T) \rangle$ *y* $\mathcal{T}' = \langle T, A_T^h - \mathit{shared}(S,T), A_T^l \cup \mathit{shared}(S,T) \rangle$. *Sea* $\mathcal{S}' \otimes \mathcal{T}'$ *tal que no existe un estado de error alcanzable desde* (s_0, t_0). *Si* $\mathcal{S}'$ *y* $\mathcal{T}'$ *son BSNNI (resp. BNNI) entonces* $S \parallel T$ *es BSNNI (resp. BNNI).*

El corolario establece como condición suficiente para que el sistema compuesto satisfaga no interferencia que $\mathcal{S}$ y $\mathcal{T}$ satisfagan la propiedad con los niveles de seguridad cambiados. De todas formas, uno espera de todas formas que $\mathcal{S}$ y $\mathcal{T}$ sean seguros cuando son analizados como procesos independientes.

En lo que queda de la sección analizaremos el problema de composicionalidad sobre las nociones de no interferencias basadas en la relación SIR. En este caso, para que las propiedades sean preservadas por la composición, sólo basta requerir que el producto no posea estados de error alcanzables. Recordemos que la condición 2 garantizaba que las transiciones internas que realizaba cada componente estaban presente en la composición. Esto permitía definir la relación (bisimulación en el caso de BSNNI/BNNI) del sistema compuesto en función de la relación en cada componente. En el caso de propiedades SIR-NNI y SIR-SNNI, esto está garantizado por la definición de no interferencia y la definición de la relación SIR. La definición de la relación SIR (Def. 2.1.11), en los items (a) y (b) no utiliza transiciones internas. por lo tanto no hay transiciones que preservar al momento de realizar la composición. En cambio, los items (c) y (d) sí utilizan este tipo de transiciones. En estos casos, la definición de SIR-NNI/SIR-SNNI y el hecho de que no existan estados de error, garantiza que no se perderán transiciones internas al realizar la composición. Si $\mathcal{S}$ es SIR-SNNI entonces $\mathcal{S}\backslash A^h \succcurlyeq \mathcal{S}/A^h$. Sean s_r y s_a tales que $s_r \succcurlyeq s_a$. Luego si $s_a \xrightarrow{a!} s_a'$ entonces $s_r \Rightarrow s_r'' \xrightarrow{a!} s_r'$. Dado que los estados s_r y s_r'' pertenecen a la interfaz $\mathcal{S}\backslash A^h$ las transiciones internas utilizadas en $s_r \Rightarrow s_r''$ no pueden ser producto de una acción confidencial que ha sido ocultada, por lo tanto las transiciones internas utilizadas permanecerán al componer $\mathcal{S}$ con otra interfaz. Si $\mathcal{S}$ es SIR-NNI entonces $(\mathcal{S}\backslash A^{h,I})/A^{h,O} \succcurlyeq \mathcal{S}/A^h$. Una vez más, si $s_r \succcurlyeq s_a$ y $s_a \xrightarrow{a!} s_a'$ entonces $s_r \Rightarrow s_r'' \xrightarrow{a!} s_r'$. En este caso, las transiciones internas utilizadas en $s_r \Rightarrow s_r''$ sí pueden ser producto de una acción confidencial (de salida) que ha sido ocultada, pero este tipo de transiciones no puede eliminarse.

Teorema 3.3.2. *Sean* $\mathcal{S} = \langle S, A_S^h, A_S^l \rangle$ *y* $\mathcal{T} = \langle T, A_T^h, A_T^l \rangle$ *dos ISS componibles tales que en el producto* $S \otimes T$ *no existe un estado de error alcanzable desde* (s_0, t_0). *Si* $\mathcal{S}$ *y* $\mathcal{T}$ *son SIR-SNNI (resp. SIR-NNI) entonces* $\mathcal{S} \parallel \mathcal{T}$ *es SIR-SNNI (resp. SIR-NNI).*

Demostración. Sea $\mathsf{R}_\mathcal{S}$ (resp. $\mathsf{R}_\mathcal{T}$) una relación SIR entre $S\backslash A_S^h$ y S/A_S^h (resp. $T\backslash A_T^h$ y T/A_T^h). Sólo se demostrará la prueba para el caso SIR-SNNI, el caso SIR-NNI es análogo. Definamos R como $(s_r, t_r)\ \mathsf{R}\ (s_a, t_a)$ sii $s_r\ \mathsf{R}_\mathcal{S}\ s_a$ y $t_r\ \mathsf{R}_\mathcal{T}\ t_a$. Demostremos que R es una relación SIR entre $(\mathcal{S} \parallel \mathcal{T})\backslash A^h$ y $(\mathcal{S} \parallel \mathcal{T})/A^h$ donde $A^h = (A_S^h \cup A_T^h) - \mathit{shared}(S,T) = A_S^h \cup A_T^h$.

Supongamos $(s_r, t_r)\ \mathsf{R}\ (s_a, t_a)$. Procedemos haciendo análisis por caso para los diferentes items de la Def 2.1.11:

(a) Supongamos $(s_r, t_r) \xrightarrow{a?} (s_r', t_r)$. Dado que $s_r\ \mathsf{R}_\mathcal{S}\ s_a$ existe un estado s_a' tal que $s_a \xrightarrow{a?} s_a'$ y

s'_r R s'_a. Como no existen estado de error en el producto, podemos asegurar que $(s_a, t_a) \xrightarrow{a?}$ (s'_a, t_a) y (s'_r, t_r) R (s'_a, t_a). El caso $(s_r, t_r) \xrightarrow{a?} (s_r, t'_r)$ es análogo.

(b) Análogo al caso anterior.

(c) Sea $(s_a, t_a) \xrightarrow{a!} (s'_a, t_a)$ y s_r $\mathsf{R}_{\mathcal{S}}$ s_a. Entonces existe s'_r tal que $s_r \Rightarrow\xrightarrow{a!} s'_r$ y s'_r $\mathsf{R}_{\mathcal{S}}$ s'_a. Sea $\hat{s}$ un estado tal que $s_r \Rightarrow \hat{s} \xrightarrow{a!} s'_r$. Todas las transiciones internas usadas para alcanzar $\hat{s}$ en $S\backslash A^h$ pueden ser ejecutadas en $(\mathcal{S} \parallel \mathcal{T})\backslash A^h$. Entonces $(s_r, t_r) \Rightarrow (\hat{s}, t_r) \xrightarrow{a!} (s'_r, t_r)$ y (s'_r, t_r) R (s'_a, t_a). El caso $(s_a, t_a) \xrightarrow{a!} (s_a, t'_a)$ es análogo.

(d) Los casos $(s_a, t_a) \xrightarrow{\tau} (s'_a, t_a)$ y $(s_a, t_a) \xrightarrow{\tau} (s_a, t'_a)$ son similares al caso anterior. Supongamos entonces que $(s_a, t_a) \xrightarrow{\tau_c} (s'_a, t'_a)$ donde τ_c es la acción interna resultante de la sincronización entre $\mathcal{S}$ y $\mathcal{T}$ en la acción común c. Sin perdida de generalidad, supongamos $s_a \xrightarrow{c?} s'_a$ y $t_a \xrightarrow{c!} t'_a$. Dado que s_r $\mathsf{R}_{\mathcal{S}}$ s_a existe un estado s'_r tal que $s_r \xrightarrow{c?} s'_r$. Repitiendo el razonamiento del caso anterior, se tiene que existe un estado $\hat{t}$ tal que $t_r \Rightarrow \hat{t} \xrightarrow{c!} t'_r$, t'_r $\mathsf{R}_{\mathcal{T}}$ t'_a y las transiciones para alcanzar $\hat{t}$ pueden realizarse en la composición. Luego $(s_r, t_r) \Rightarrow (s_r, \hat{t}) \xrightarrow{c;} (s'_r, t'_r)$ y (s'_r, t'_r) R (s'_a, t'_a).

□

3.4. Chequeo y síntesis de sistemas seguros

Esta sección se divide en dos partes. La primera presenta el algoritmo para verificar si una ISS es SIR-SNNI. La segunda parte presenta el algoritmo de síntesis.

3.4.1. Verificación de sistemas seguros

Dado que una ISS $\mathcal{S}$ es SIR-SNNI si y sólo si $\mathcal{S}\backslash A^h \succcurlyeq \mathcal{S}/A^h$, para decidir si $\mathcal{S}$ es SIR-SNNI se debe verificar la existencia de la relación $\succcurlyeq$ entre $\mathcal{S}\backslash A^h$ y $\mathcal{S}/A^h$. En este proceso, al mismo tiempo, se incluye la información necesaria para que en el caso de que la relación no exista, decidir si es posible sintetizar una interfaz segura o no. Los pasos que realiza este algoritmo son los siguientes:

(i) Las acciones internas se reetiquetan como τ y τ'. En particular, utilizaremos τ' para reetiquetar las acciones altas de entradas que fueron ocultadas en S/A^h para saber cuales son las acciones internas eliminables en el algoritmo de síntesis. Todas las otras se reetiquetan con τ.

(ii) La interfaz $\mathcal{S}\backslash A^h$ es *semi-saturada*. En este proceso se agregan transiciones $q \xrightarrow{a} q'$ con $a \in A^O$ si $q \Rightarrow\xrightarrow{a} q'$. Esto se realiza para verificar las condición (c) de la Def. 2.1.11.

(iii) Se construye el *producto semi-sincronizado P* entre $\mathcal{S}\backslash A^h$ y $\mathcal{S}/A^h$. En este producto, las transiciones con igual etiqueta se sincronizan bajo ciertas condiciones que dependen de la Def. 2.1.11.

(iv) Se analizan las propiedades de transferencia de SIR a través de P y se lo clasifica en tres categorías: pasa el test de relación, puede pasarlo y no lo pasa. Sólo en el primer caso se verifica SIR-SNNI. Si el algoritmo determina que P puede pasar el test entonces se puede sintetizar de S una interfaz segura.

Las etiquetas diferentes para las transiciones internas no tiene ningún rol en la definición de la relación SIR. Entonces, para simplificar el análisis, se reemplazan todas las transiciones internas por dos nuevas: τ y τ'. La etiqueta τ' es usada para representar una transición interna que puede ser eliminada al momento de la síntesis; en nuestro contexto una transición interna de este tipo puede ser eliminada porque es una acción de entrada confidencial que fue oculta para verificar la propiedad de seguridad. La etiqueta τ se utiliza para identificar las acciones internas que no pueden ser eliminadas. Esta transformación se formaliza en la siguiente definición, la cual también agrega *transiciones self-loops* (transiciones de un estado a sí mismo) con las acciones τ y τ' para obtener futuras simplificaciones.

Definición 3.4.1. *Sea* $S = \langle Q, s_0, A^I \cup A^O \cup A^H, \rightarrow \rangle$ *un IA y* $B \subseteq A^H$. *Definimos* el marcado de B en S *como la IA* $S_B = \langle Q, s_0, A^I \cup A^O, \{\tau, \tau'\}, \rightarrow_{S_B} \rangle$ *donde* $\rightarrow_{S_B}$ *es la menor relación que satisface las siguientes reglas:*

$$q \xrightarrow{\tau}_{S_B} q \qquad q \xrightarrow{\tau'}_{S_B} q \qquad \frac{q \xrightarrow{a} q' \quad a \in A^I \cup A^O}{q \xrightarrow{a}_{S_B} q'}$$

$$\frac{q \xrightarrow{a} q' \quad a \in B}{q \xrightarrow{\tau'}_{S_B} q'} \qquad \frac{q \xrightarrow{a} q' \quad a \in A^H - B}{q \xrightarrow{\tau}_{S_B} q'}$$

Dado un ISS $\mathcal{S}$*, el ISS* $\mathcal{S}_B$*, es el ISS obtenido luego de marcar B en el IA subyacente.*

En algunos casos el posible transformar un sistema de transiciones y de esa forma reducir una equivalencia R en una equivalencia R' [70]. Por ejemplo, es posible *saturar* un sistema de transiciones para reducir el problema de verificar una bisimulación débil a un problema de verificar una bisimulación fuerte. La saturación se realiza agregando una nueva transición $q \xrightarrow{a} q'$ al modelo por cada transición $q \stackrel{a}{\Rightarrow} q'$. Luego si estamos comparando p y q y $p \xrightarrow{a} p'$, la saturación asegura que alcanza con relacionar p' con algún q' tal que $q \xrightarrow{a} q'$. Utilizando una idea similar, podemos verificar si existe una relación SIR entre dos sistemas. Se agregará al sistema una transición $q \xrightarrow{a} q'$ siempre que $q \Rightarrow \xrightarrow{a} q'$ con a una acción de salida. Llamamos a este proceso *semi-saturación*.

Definición 3.4.2. *Sea* S *un IA tal que* $A^H_S = \{\tau\}$. *La* semi-saturación de S *es un IA* $\overline{S} = \langle Q, s_0, A^I \cup A^O \cup \{\tau\}, \rightarrow_{\overline{S}} \rangle$ *donde* $\rightarrow_{\overline{S}}$ *es la menor relación que satisface las siguientes reglas:*

$$\frac{q \xrightarrow{a} q'}{q \xrightarrow{a}_{\overline{S}} q'} \qquad \frac{q \xrightarrow{\tau}_{\overline{S}} q' \quad q' \xrightarrow{a}_{\overline{S}} q'' \quad a \in A^O}{q \xrightarrow{a}_{\overline{S}} q''}$$

Dado un ISS $\mathcal{S}$*, su* semi-saturación $\overline{\mathcal{S}}$*, es obtenida semi-saturando la IA subyacente.*

La última definición nos asegura que si $a \in A^O$ entonces $q \Rightarrow\!\!\xrightarrow{a}_S q'$ si y sólo si $q \xrightarrow{a}_{\overline{S}} q'$. Esto se utilizará para verificar la condición (c) de la relación SIR. La restricción $A_S^H = \{\tau\}$ se debe a que el proceso se aplicará solo a la interfaz $S\backslash A^h$ o $(S\backslash A^{h,I})/A^{h,O}$ y por lo tanto ésta no posee acciones internas que pueden ser eliminadas. Este detalle será de suma importancia para demostrar luego la corrección del procedimiento para sintetizar interfaces seguras.

Siguiendo las ideas de [56] y [30], la definición del producto semi-sincronizado depende de las condiciones impuestas por la relación SIR. Primero recapitulemos estas condiciones y luego presentemos la definición del producto. Si dos estados s y t son tales que $s \succcurlyeq t$, toda acción de salida/oculta que t pueda ejecutar tiene que ser una acción que s también pueda ejecutar (probablemente usando transiciones internas). Por otro lado, t no está forzado a realizar las acciones de salida/oculta que ejecuta s. Además, ambos estados tienen que ejecutar las mismas acciones de entrada pero sin realizar transiciones internas para lograrlo. Cuando una de las condiciones no se satisface entonces la relación no existe; esto se modela en el producto mediante una transición a un estado especial: el estado *fail*. Teniendo en cuenta esto, definimos el *producto semi-sincronizado* como sigue.

Definición 3.4.3. *Sean* $S = \langle Q_S, s_0, A^I \cup A^O \cup \{\tau\}, \rightarrow_S \rangle$ *y* $T = \langle Q_T, t_0, A^I \cup A^O \cup \{\tau, \tau'\}, \rightarrow_T \rangle$ *dos IA tales que* S *está semi-saturada. El* producto semi-sincronizado de S *y* T *es un IA* $S \times T = \langle (S \times T) \cup \{fail\}, (s_0, t_0), A^I \cup A^O \cup \{\tau, \tau'\}, \rightarrow_{S\times T} \rangle$ *donde* $\rightarrow_{S\times T}$ *es la menor relación que satisface las siguientes reglas:*

$$\frac{s \xrightarrow{a}_S s' \quad t \xrightarrow{a}_t t'}{(s,t) \xrightarrow{a}_{S\times T} (s',t')} \qquad \frac{s \xrightarrow{\tau}_S s' \quad t \xrightarrow{\tau'}_T t'}{(s,t) \xrightarrow{\tau}_{S\times T} (s',t')} \qquad \frac{s \xrightarrow{\tau}_S s' \quad t \xrightarrow{\tau'}_T t'}{(s,t) \xrightarrow{\tau'}_{S\times T} (s',t')}$$

$$\frac{s \xrightarrow{a}_S \quad t \not\xrightarrow{a}_T \quad a \in A^I}{(s,t) \xrightarrow{a}_{S\times T} fail} \qquad \frac{s \not\xrightarrow{a}_S \quad t \xrightarrow{a}_T \quad a \in A^I}{(s,t) \xrightarrow{a}_{S\times T} fail} \qquad \frac{s \not\xrightarrow{a}_S \quad t \xrightarrow{a}_T \quad a \in A^O}{(s,t) \xrightarrow{a}_{S\times T} fail}$$

Dado dos ISS $\mathcal{S} = \langle S, A^h, A^l \rangle$ *y* $\mathcal{T} = \langle T, A^h, A^l \rangle$ *con* S *y* T *satisfaciendo las condiciones anteriores, el* producto semi-sincronizado *de* $\mathcal{S}$ *y* $\mathcal{T}$ *está definido por el ISS* $\mathcal{S} \times \mathcal{T} = \langle S \times T, A^h, A^l \rangle$.

Expliquemos ahora cómo es posible usar el producto sincronizado para verificar y derivar, si es posible, dos procesos que se encuentren relacionados por una relación SIR.

Supongamos que (s,t) es un estado del producto sincronizado tal que sólo ejecuta una transición $(s,t) \xrightarrow{a}_{S\times T} fail$. Esta transición es una evidencia de que $s \not\succcurlyeq t$: las primeras dos reglas de la última fila garantizan que existe una entrada no común entre los estados. La última regla garantiza que t ejecuta una acción de salida que s no puede ejecutar; notar que s es un estado semi-saturado, por esta razón se sabe que s no puede ejecutar la acción a utilizando transiciones internas. Si $a \in A^O$ entonces t ejecuta la acción de forma autónoma, por lo tanto este par es insalvable. En este caso se dice que (s,t) *falla el test de la relación SIR*. Por otro lado, si $a \in A^I$ entonces un estado ofrece un servicio que el otro no. Dado que a es una acción controlable, sería posible lograr que estos estados sean seguros si se prohíbe la realización de esta acción. Notar que eliminando la transición de entrada con etiqueta a se evita la transición $(s,t) \xrightarrow{a}_{S\times T} fail$ en el producto sincronizado y dado que esta es la única transición que realiza el estado (s,t) se

$$Fail^0 = \{(s,t) \mid (s,t) \xrightarrow{a}_{S\times T} fail, a \in A^O\} \cup \{fail\}$$

$$Fail^{k+1} = Fail^k \cup \{(s,t) \mid a \in A^O \cup \{\tau\}, t \xrightarrow{a} t', (\forall s' : (s,t) \xrightarrow{a} (s',t') : (s',t') \neq (s,t), (s',t') \in Fail^k)\}$$

Tabla 3.1.: Definición del conjunto de estados *Fail*.

$$May^0 = \bigcup\{May^0_{q \xrightarrow{a} q'} \mid q \xrightarrow{a} q' \in \rightarrow_S \cup \rightarrow_T\}$$

$$May^0_{q \xrightarrow{a} q'} = \{(s,t) : (q = s \vee q = t), a \in A^I, (s,t) \xrightarrow{a}_{S\times T} fail\}$$

$$May^{k+1} = May^k \cup \left(\bigcup\{May^{k+1}_{q \xrightarrow{a} q'} \mid q \xrightarrow{a} q' \in \rightarrow_S \cup \rightarrow_T\}\right)$$

$$May^{k+1}_{s \xrightarrow{a}_S s'} = \{(s,t) \notin Fail : a \in A, (\forall t' : (s,t) \xrightarrow{a}_{S\times T} (s',t') : (s',t') \neq (s,t), (s',t') \in Fail \cup May^k)\}$$

$$May^{k+1}_{t \xrightarrow{a}_T t'} = \{(s,t) \notin Fail : a \in A, (\forall s' : (s,t) \xrightarrow{a}_{S\times T} (s',t') : (s',t') \neq (s,t), (s',t') \in Fail \cup May^k)\}$$

Tabla 3.2.: Definición del conjunto de estados *May*.

obtienen dos estados tales que $s \geqslant t$. En este caso se dice que el par (s,t) *podría pasar el test de la relación SIR*.

Para ambos casos, la no coincidencia en el estado (s,t) debe ser propagada hacia atrás, de forma tal de identificar los estados que no pueden ser relacionados por una relación SIR. Para ello utilizamos los conjuntos de estados: *Fail* y *May*. El conjunto *Fail* contiene aquellos pares de estados que no pueden ser relacionados por una relación SIR y no existe un conjunto de transiciones con acciones de entradas tal que al eliminarlas exista una relación. Por otro lado, *May* contiene los pares de estados que no están relacionados por una relación SIR pero existe una posibilidad de relacionarlos luego de eliminar algunas transiciones con acciones de entrada. Los estados que no pertenecen a $Fail \cup May$, pertenecerán al conjunto *Pass*. El conjunto *Pass* de pares de estados *es* una relación SIR.

Definición 3.4.4. *Sea $S \times T$ un producto sincronizado. Definimos los conjuntos Fail, May, Pass $\subseteq S \times T$ respectivamente por:*

- *$Fail = \cup_{i=0}^{\infty} Fail^i$ donde $Fail^i$ está definido en la Tabla 3.1. Si $q \in Fail$, decimos que el par q* falla el test de relación SIR.
- *$May = \cup_{i=0}^{\infty} May^i$ donde May^i está definido en la Tabla 3.2. Si $q \in May$, decimos que el par q* puede pasar el test de relación SIR.

- $Pass = S \times T - (May \cup Fail)$. *Si* $q \in Pass$*, decimos que el par* q pasa el test de relación SIR

El producto semi-sincronizado $S \times T$ *falla (puede pasar, pasa) el test de relación SIR si su estado inicial lo falla (lo puede pasar, lo pasa).*

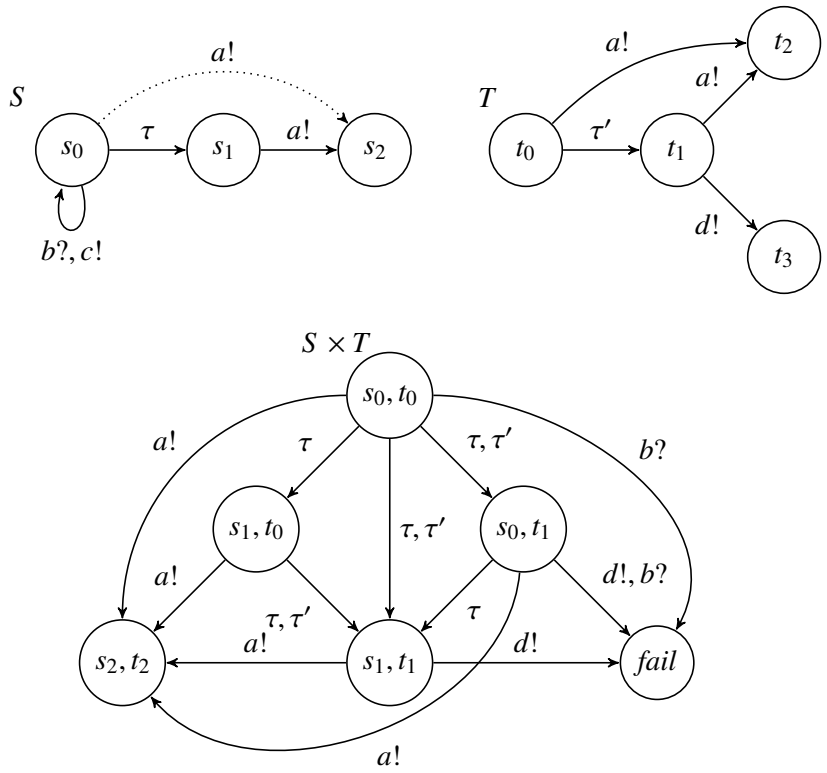


Figura 3.8.: Un ejemplo de producto sincronizado

Ejemplo 3.4.1. *En la Figura 3.8 se grafican dos IA* S *y* T *y su producto semi-sincronizado* $S \times T$*. La transición graficada en lineas de punto en* S *es la transición agregada por la saturación. Las transiciones self loops con acciones* τ *y* τ' *no se grafican. El producto semi-sincronizado* $S \times T$ *puede pasar el test de relación; los estados se clasifican de la siguiente manera:*

$$Fail = \{fail, (s_1, t_1), (s_0, t_1)\} \qquad May = \{(s_0, t_0), (s_1, t_0)\} \qquad Pass = \{(s_2, t_2)\}$$

Notar que, para clasificar un estado, no sólo se utiliza el producto sincronizado sino que también se utilizan las transiciones de los sistemas que se sincronizaron. Esto es claro al ver las definiciones auxiliares de Fail y May.

Los estados (s_0, t_1) *y* (s_1, t_1) *transicionan al estado fail porque* t_1 *ejecuta una transición con la acción* $d!$ *que los estados* s_0 *y* s_1 *no pueden imitar. Por esta razón son considerados estados*

que fallan el test. Estos son los únicos estados que fallan el test de relación. Por otro lado, un estado de S que está relacionado con un estado de T no está obligado a ejecutar acciones de salidas que el otro ejecuta. Por esta razón $s_0 \xrightarrow{c!} s_0$, $t_0 \xrightarrow{c!}$ *y* $t_1 \xrightarrow{c!}$ *no generan transiciones las* $(s_0, t_0) \xrightarrow{c!}$ *fail y* $(s_0, t_1) \xrightarrow{c!}$ *fail en el producto semi-sincronizado .*

Al analizar el estado (s_1, t_0) *y a la transición* $t_0 \xrightarrow{\tau'} t_1$ *se concluye que* (s_1, t_0) *puede pasar el test de relación porque la acción de la transición es* τ' *y todos los estados que se alcanzan, en este caso* (s_1, t_1)*, fallan el test. De igual forma, el estado* (s_0, t_0) *es catalogado como un estado que puede pasar el test. Estos son los únicos estados que pueden pasar el test de relación.*

Como los conjuntos Fail y May ya se encuentran totalmente definidos entonces el estado (s_2, t_2) *es el único estado que pasa el test.*

El siguiente Lema garantiza que un estado (s, t) pasa el test SIR sii $s \succcurlyeq t$

Teorema 3.4.1. *Dado un producto sincronizado* $S \times T$*, el par de estados* (s, t) *pasa el test SIR sii* $s \succcurlyeq t$*.*

Demostración. Dado que $(May \cup Fail) \cap Pass = \emptyset$, tenemos que demostrar que

(I) si $(s, t) \in May \cup Fail$ entonces $s \not\succcurlyeq t$ y

(II) si $(s, t) \in Pass$ entonces $s \succcurlyeq t$

La prueba de (I) es por inducción en k, primero con respecto a $Fail^k$, luego con respecto a May^k. Si $(s, t) \in Fail^0$ entonces $t \xrightarrow{a}$ y $s \xrightarrow{a}$ con $a \in A^O$, por lo tanto $s \not\succcurlyeq t$. Supongamos que $(s', t') \in Fail^k$ implica $s' \not\succcurlyeq t'$ y demostremos que $(s, t) \in Fail^{k+1}$ implica $s \not\succcurlyeq t$. Por definición de $Fail^{k+1}$ existe $a \in A^O \cup \{\tau\}$ tal que $t \xrightarrow{a} t'$ y $(\forall s' : (s, t) \xrightarrow{a} (s', t') : (s', t') \in Fail^k)$. Por hipótesis inductiva la condición (c) de Def. 2.1.11 no se satisface, entonces $s \not\succcurlyeq t$. La inducción en k para May^k procede de la misma manera.

Para probar (II) verifiquemos que *Pass* es una SIR, es decir que, para todo estado $(s, t) \in Pass$:

1. si $s \xrightarrow{a} s'$ y $a \in A^I$ entonces existe t' tal que $(s, t) \xrightarrow{a} (s', t')$ y $(s', t') \in Pass$.
2. si $t \xrightarrow{a} t'$ entonces existe s' tal que $(s, t) \xrightarrow{a} (s', t')$ y $(s', t') \in Pass$.

En ambos casos se procede de la misma manera: se supone que la condición no es válida, luego se obtiene que $(s, t) \notin Pass$. Por construcción del producto sincronizado es directo ver que las condiciones de la definición Def. 2.1.11 se satisfacen. Por lo tanto *Pass* es una relación SIR. □

Este teorema permite verificar si un ISS $\mathcal{S}$ es SIR-SNNI, debido a que $\mathcal{S}$ es SIR-SNNI si y sólo si $S \setminus A^h \succcurlyeq S/A^h$. Pero esto no se puede hacer de forma directa. Notemos que no se puede utilizar $\mathcal{S}\setminus A^h$ y $\mathcal{S}/A^h$ para crear el producto sincronizado porque en general $\mathcal{S}\setminus A^h$ no satisface $A^H = \{\tau\}$ y no está semi-saturado. Esto puede resolverse marcando el conjunto $\emptyset$ en $\mathcal{S}\setminus A^h$ y luego semi-saturando la componente, i.e. reemplazamos $S \setminus A^h$ por $\overline{(S \setminus A^h)}_{\emptyset}$. De forma similar, S/A^h no satisface $A^H = \{\tau, \tau'\}$. Dado que τ' se utiliza para representar las acciones internas que pueden ser eliminadas marcamos el conjunto $A^{h,I}$ en S/A^h. Entonces reemplazamos S/A^h por $(S/A^h)_{A^{h,I}}$.

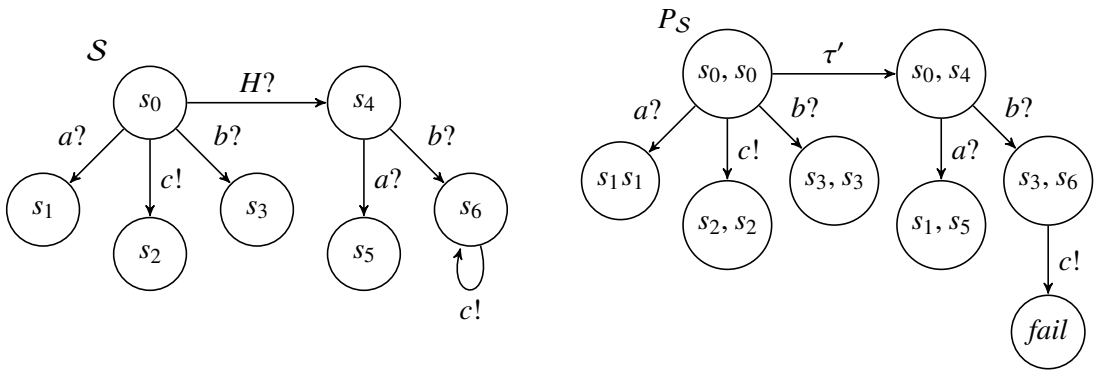


Figura 3.9.: Un ISS $\mathcal{S}$ y su producto sincronizado $P_{\mathcal{S}}$. $\mathcal{S}$ no es SIR-SNNI.

Por lo tanto, para verificar que $\mathcal{S}$ satisface SIR-SNNI podemos verificar si $\overline{S\setminus A^h}_{\emptyset} \times (S/A^h)_{A^{h,I}}$ pasa el test de relación SIR.

Aplicando un razonamiento similar, para verificar la propiedad SIR-NNI, se debe verificar si $\overline{((S\setminus A^{h,I})/A^{h,O})}_{\emptyset} \times (S/A^h)_{A^{h,I}}$ pasa el test de relación SIR.

Entonces como consecuencia del Teorema 3.4.1, tenemos un algoritmo de decisión para verificar si un sistema satisface la propiedad SIR-SNNI o SIR-NNI, lo cual queda establecido en el siguiente teorema.

Teorema 3.4.2. *Sea $\mathcal{S} = \langle S, A^h, A^l\rangle$ una ISS.*

1. *$\mathcal{S}$ satisface SIR-SNNI sii $\overline{(S\setminus A^h)}_{\emptyset} \times (S/A^h)_{A^{h,I}}$ pasa el test de relación SIR.*

2. *$\mathcal{S}$ satisface SIR-NNI sii $\overline{((S\setminus A^{h,I})/A^{h,O})}_{\emptyset} \times (S/A^h)_{A^{h,I}}$ pasa el test de relación SIR.*

Ejemplo 3.4.2. *En la Figura 3.9 se grafica una ISS $\mathcal{S}$ y a su derecha el producto semi-sincronizado $P_{\mathcal{S}} = \overline{(S\setminus A^h)}_{\emptyset} \times (S/A^h)_{A^{h,I}}$. En este caso $P_{\mathcal{S}}$ puede pasar el test SIR; los estados del producto se clasifican de la siguiente forma*

$$\mathit{Fail} = \{(s_3, s_6)\} \quad \mathit{May} = \{(s_0, s_0), (s_0, s_4)\} \quad \mathit{Pass} = \{(s_1, s_1), (s_2, s_2), (s_3, s_3), (s_1, s_5)\}$$

Por lo tanto por lo tanto $\mathcal{S}$ no satisface SIR-SNNI. Es claro que (s_3, s_6) falla el test porque s_6 puede ejecutar una transición con $c!$ y s_3 no. Luego este error se propaga al estado (s_0, s_4) debido a que los estados s_0 y s_4 ejecutan ambos una transición con acción $b?$; la única forma de sincronizar ambas transiciones es alcanzando un estado que falla el test, i.e. (s_3, s_6). Luego s_0 y s_4 también "fallan" el test de relación SIR, pero como la acción que alcanza el estado (s_3, s_6) es una acción de entrada ($b?$), que podría eliminarse para evitar la falla, se tiene $(s_0, s_4) \in \mathit{May}$. De forma similar se propaga la inconsistencia al estado (s_0, s_0).

Ejemplo 3.4.3. *En la Figura 3.10 se grafica el ILTS $\mathcal{T}$ y abajo el producto semi-sincronizado*

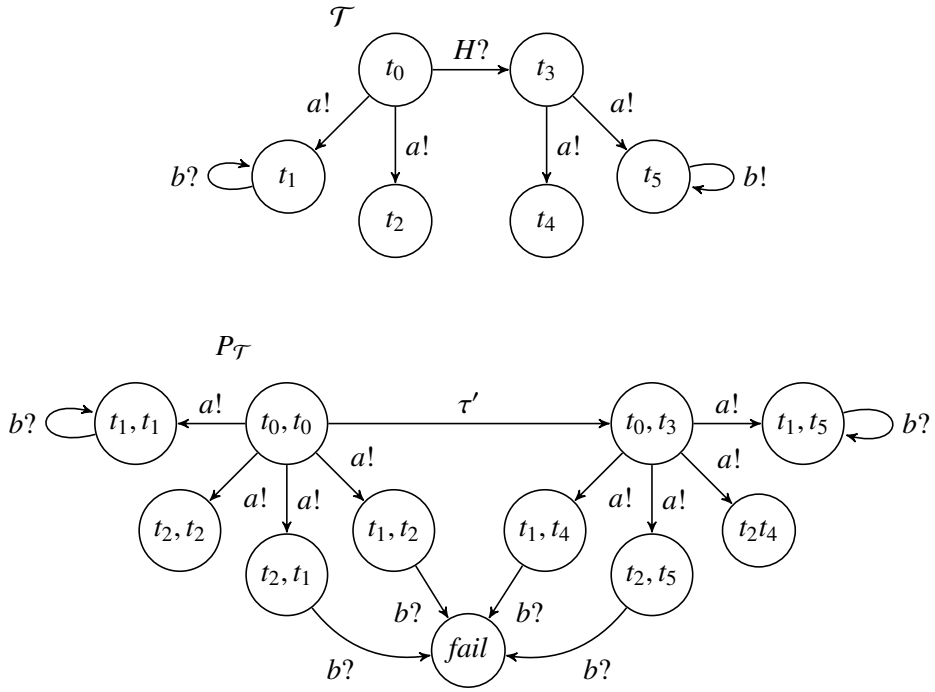


Figura 3.10.: Un ISS $\mathcal{T}$ y su producto sincronizado $P_{\mathcal{T}}$. $\mathcal{T}$ es SIR-SNNI.

$P_{\mathcal{T}} = \overline{(T\backslash A^h)}_{\emptyset} \times (T/A^h)_{A^{h,l}}$. *Los estados del producto se clasifican de la siguiente forma*

$$Fail = \{(t_2, t_1), (t_1, t_2), (t_1, t_4), (t_2, t_5), fail\} \qquad May = \emptyset$$

$$Pass = \{(t_0, t_0), (t_1, t_1), (t_2, t_2), (t_0, t_3), (t_2, t_4), (t_1, t_5)\}$$

Por lo tanto $P_{\mathcal{T}}$ pasa el test SIR. Notar que la existencia de un camino desde el estado (t_0, t_0) hasta el estado fail no es indicio de que la relación no exista. Tomemos por ejemplo el estado $(t_0, t_3) \in Pass$, el cual también pasa el test de relación y alcanza al estado fail. Si analizamos las transiciones que puede realizar el estado t_0 se ve que éstas siempre pueden relacionarse con una transición de t_3. Si se toma $t_0 \xrightarrow{a!} t_1$ el producto sincronizado indica que esta transición no puede relacionarse con la transición $t_3 \xrightarrow{a!} t_4$ pues (t_1, t_4) es un estado en Fail, pero si puede relacionarse con la transición $t_3 \xrightarrow{a!} t_5$ y alcanzar el estado (t_1, t_5) el cual no falla el test de relación. Un análisis similar se puede realizar con la otra transición de t_0 y con las transiciones del estado t_3. De esta forma se concluye que (t_0, t_3) pasa el test de relación SIR.

3.4.2. Síntesis de sistemas seguros

A continuación se demuestra que si un producto semi-sincronizado P puede pasar el test SIR entonces existe un conjunto de transiciones con acciones de entrada que al eliminarlas del IA original, el producto semi-sincronizado de la nueva ISS pasa el test. Este resultado será utilizado para derivar una interfaz segura.

Sea P un producto semi-sincronizado y sea $EC(P)$ (ver Tabla 3.3) el conjunto de transiciones que serán candidatas a ser eliminadas. La definición de $EC(P)$ es bastante intuitiva: el conjunto está compuesto por las transiciones que unen un estado que puede pasar el test de relación SIR con uno que lo falla. Esto puede pasar por cuatro diferentes razones. La primera es el caso en el que uno de los estados ejecuta una acción de entrada que el otro no puede ejecutar. Los siguientes dos casos son simétricos y consideran el caso en el que ambos estados pueden ejecutar una acción de entrada baja la cual tiene como destino un estado que falla el test. El último caso tiene en cuenta las transiciones ocultas, producto de la abstracción de una acción confidencial de entrada, que alcanzan un estado que falla el test. Notemos que si P puede pasar el test de relación entonces $EC(P_S) \neq \emptyset$.

El siguiente lema establece que si $P = S \times T$ es un producto semi-sincronizado que puede pasar el test de relación SIR y S' (resp. T') se obtiene de S (resp. T) luego de eliminar algunas transiciones de $EC(P)$ entonces el producto semi-sincronizado $P' = S' \times T'$ puede pasar o pasa el test de relación SIR.

Lema 3.4.1. *Sea $P = S \times T$ un producto semi-sincronizado que puede pasar el test de relación SIR y sean S' y T' los IA que se obtienen de S y T, respectivamente, luego de eliminar algunas transiciones de $EC(P)$. Entonces el producto semi-sincronizado $P' = S' \times T'$ puede pasar o pasa el test de relación SIR.*

Demostración. Sean $Fail_P$, $Pass_P$ y May_P los estados que fallan, pasan y pueden pasar el test de relación con respecto a P. Igual notación vale para $Fail_{P'}$, $Pass_{P'}$ y $May_{P'}$ con respecto a P'. Para demostrar el Lema, alcanza con demostrar que

$$
\begin{aligned}
EC(P) = &\{q \xrightarrow{a} q' \mid (\exists \hat{q} : (q,\hat{q}) \in May^0_{q \xrightarrow{a} q'} \vee (\hat{q},q) \in May^0_{q \xrightarrow{a} q'})\} \cup \\
&\{q \xrightarrow{a} q' \mid a \in A^I, (\exists \hat{q},\hat{q}' : (q,\hat{q}) \in May^1_{q \xrightarrow{a} q'}, (q,\hat{q}) \xrightarrow{a} (q',\hat{q}') : (q',\hat{q}') \in Fail)\} \cup \\
&\{q \xrightarrow{a} q' \mid a \in A^I, (\exists \hat{q},\hat{q}' : (\hat{q},q) \in May^1_{q \xrightarrow{a} q'}, (\hat{q},q) \xrightarrow{a} (\hat{q}',q') : (\hat{q}',q') \in Fail)\} \cup \\
&\{q \xrightarrow{a} q' \mid a \in A^{h,I}, (\exists \hat{q} : (\hat{q},q) \in May^1_{q \xrightarrow{\tau'} q'}, (\forall \hat{q}' : (\hat{q},q) \xrightarrow{\tau'} (\hat{q}',q') : (\hat{q}',q') \in Fail))\}
\end{aligned}
$$

Tabla 3.3.: Conjunto de transiciones candidatas para eliminar.

(I) $Fail_{P'} = Fail_P$

(II) $Pass_{P'} \cup May_{P'} = Pass_P \cup May_P$.

Notemos que si bien $Fail_{P'} = Fail_P$, muchos de estos estados ya no serán alcanzables desde el estado inicial de P'. Además, dado que *Fail* y $Pass \cup May$ forman una partición de estados, basta con demostrar (I) para garantizar (II). Demostrar (I) es directo pues las condiciones para que un estado pertenezca a *Fail* sólo dependen de transiciones con acciones de salida o τ' y estas acciones no se han modificado.

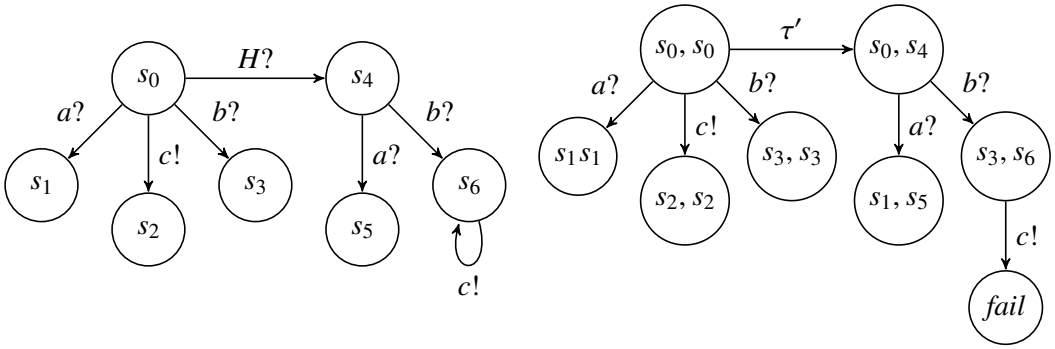
□

En base a este Lema definimos el siguiente procedimiento el cual sintetiza una interfaz segura. La finalización del mismo está garantizada porque el conjunto de transiciones de entradas es finito.

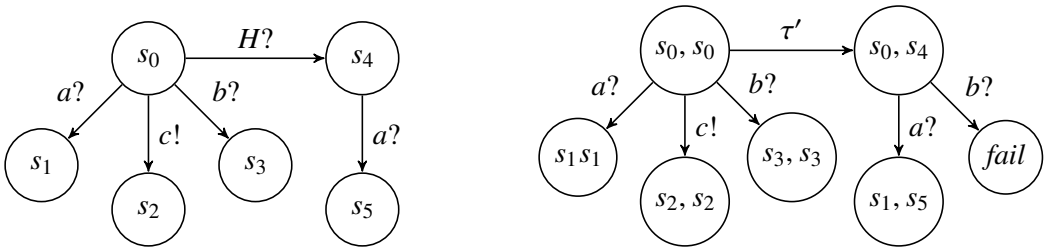
1. Sea $\mathcal{S}$ tal que su producto semi-sincronizado $P_{\mathcal{S}} = \overline{S \setminus A^h}_{\emptyset} \times (S/A^h)_{A^{h,I}}$ puede pasar el test de relación.
2. Eliminar de $\mathcal{S}$ alguna transición de $EC(P_{\mathcal{S}})$.
3. Reconstruir $P_{\mathcal{S}}$ en función del nuevo $\mathcal{S}$
4. Si $P_{\mathcal{S}}$ pasa el test terminar, caso contrario volver a 2.

Ejemplo 3.4.4. *En la Figura 3.11 se muestra el algoritmo de refinamiento aplicado a la ISS $\mathcal{S}$ de la Figura 3.9. Luego de la construcción del primer producto sincronizado, el conjunto de transiciones candidatas para eliminar está compuesto por $EC(P_{\mathcal{S}}) = \{s_0 \xrightarrow{b} s_3, s_4 \xrightarrow{b?} s_6\}$. La elección de la transición a eliminar es no determinista, en este ejemplo se elige eliminar la transición $s_4 \xrightarrow{b?} s_6$. El IA obtenido no satisface aun SIR-SNNI, entonces continuamos eliminando transiciones. Luego de eliminar la transición $s_0 \xrightarrow{b} s_3$ sí se obtiene un sistema seguro.*

$\mathcal{S}$ y su producto sincronizado:



$\mathcal{S}$ luego de eliminar $\{s_4 \xrightarrow{b?} s_6\}$:



$\mathcal{S}$ luego de eliminar $\{s_4 \xrightarrow{b?} s_6, s_0 \xrightarrow{b?} s_3\}$:

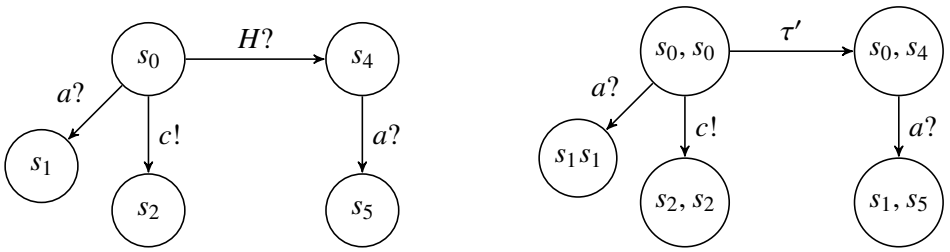


Figura 3.11.: Ejemplo del algoritmo de derivación.

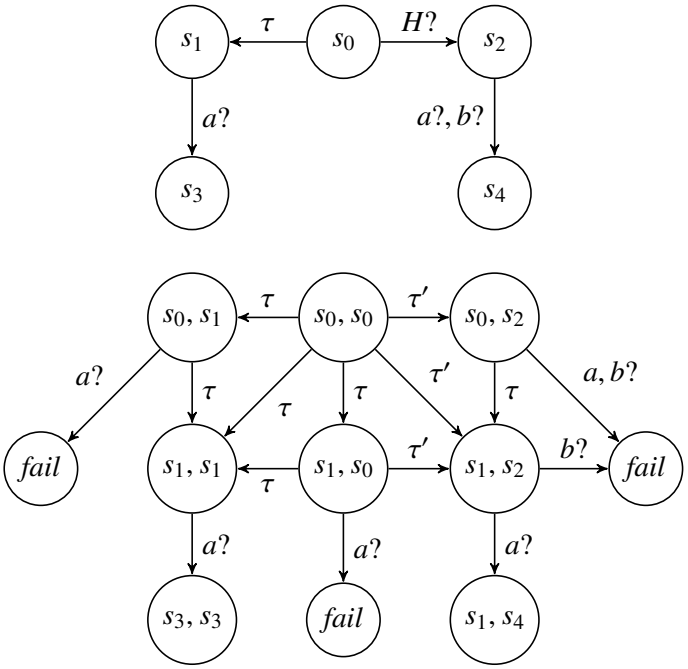
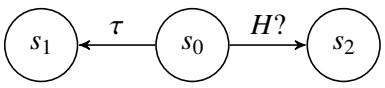


Figura 3.12.: El $\mathcal{S}$ y su producto semi-sincronizado $P_{\mathcal{S}}$

Notemos que la elección no determinista de la transición que se elimina permite que el algoritmo sintetice interfaces distintas desde una misma interfaz. En la Figura 3.12, se presenta una interfaz $\mathcal{S}$ y su producto sincronizado P_S, el cual puede pasar el test. Es claro que si se elimina la transición $s_2 \xrightarrow{b?} s_4$ de $\mathcal{S}$ se obtiene una interfaz segura. Sin embargo, el conjunto $EC(P_{\mathcal{S}})$ es tal que $\{s_1 \xrightarrow{a?} s_3, s_2 \xrightarrow{a?} s_4\} \subseteq EC(P_{\mathcal{S}})$. Si una de estas transiciones se elimina, entonces, la otra también debe ser eliminada. No sólo esto, esto no genera una interfaz segura por lo que finalmente se deberá eliminar la transición $s_2 \xrightarrow{b?} s_4$ y se obtendría la siguiente interfaz:



Si bien la interfaz es segura la misma no provee ningún servicio al usuario bajo y por lo tanto podría considerarse inútil. Para solucionar esto, al algoritmo se le debería introducir heurísticas que se encarguen de garantizar que la interfaz que se sintetiza, además de segura, satisface los requisitos extras de funcionalidad. Dado que estos requisitos están ligados directamente al contexto de la interfaz, no presentaremos posible soluciones para el problema.

3.5. Observaciones finales

3.5.1. Trabajos relacionados

En [41] se presenta por primera vez no interferencia sobre un modelo basado en sistema de transiciones. A partir de ahí se han introducido muchas variantes de la propiedad las cuales se han estudiado en distintos modelos. En este contexto no es novedad la diferentes naturalezas entre las acciones de entrada y de salida. Por esta razón la variante simple y la variante strong de no interferencia. Pero esta diferencia no ha sido explotada a nivel de semánticas como se ha hecho en el presente Capítulo.

El enfoque basado en álgebras de procesos ha sido bastante estudiado [31, 32, 69]. En [31, 32] se utilizan las dos variantes de no interferencia con semánticas para LTS, mientras que en [69], luego de marcar la diferencia los dos tipos de acciones, ésta se desestima.

Por otro lado, en [61] se argumenta que a pesar de los avances realizados en el contexto de álgebras de procesos, este avance puede ser desestimado por los sectores que están acostumbrado a trabajar con modelos basados en estados. Por esta razón se establecen transformaciones entre los modelos estudiados en cada enfoque y se estudia como se comportan distintas variantes de no interferencia. Es importante remarcar que las dos máquinas de estados que se presentan en [61] no son similares a las utilizadas en el presente trabajo. Un tipo de máquina está *basada en estados*, mientras que la otra está *basada en acciones*. En la primera, cada usuario realiza una observación particular en cada estado, mientras que en la segunda cada usuario que ejecuta una acción, ve una observación como resultado de la acción. Esta observación es sólo vista por el ejecutor de la acción. Por otro lado, para representar las semánticas de las álgebras de procesos, como es usual, se usa un LTS. Luego, las ISS se encuentran más cerca de la semántica del álgebra de procesos que a los sistemas basados en estados. Al estudiar la variante de no interference denominada *restrictiveness* [60], la cual se basa en definir una relación denominada *unwinding*, se argumenta que las acciones de salidas confidenciales no pueden ser controladas por lo tanto deben ser interpretadas como acciones internas. Este análisis se basa en [68], donde las acciones altas de salidas son interpretados como *señales de eventos*. Esta observación es la misma que se utiliza en el capítulo anterior para diferenciar entre acciones de entrada y de salida y definir todo un conjunto nuevo de semánticas y es la misma que justifica las dos variantes de no interferencia. Luego de la observación, y al igual que en la variante simple de no interferencia, las acciones confidenciales de entrada son restringidas mientras que las de salidas son ocultadas, pero todo esto se formaliza sin la inclusión de los operadores de restricción y abstracción.

Con respecto a la síntesis de sistemas seguros, ésta es un área que no ha sido muy desarrollada. En [18] se resuelve este problema para no interferencia basada en trazas, con la particularidad de que no hay diferencia entre acciones de entrada y acciones de salida. Las acciones que se pueden restringir son las acciones altas, pues estas son definidas como controlables. En [35] y [12] se extiende el algoritmo para sintetizar interfaces seguras para *Timed Automata* [4, 5, 13], una variante de interfaces de autómata donde se modela tiempo. Además, en [12] el conjunto de acciones controlables está compuesto por acciones bajas y acciones altas.

Los algoritmos aquí presentados no pueden generalizarse de forma directa dado que se encuentran estrechamente ligados a la caracterización relacional de la relación SIR. Sin embargo pueden pueden adaptarse para otras variantes de no interferencia basadas en otras nociones de

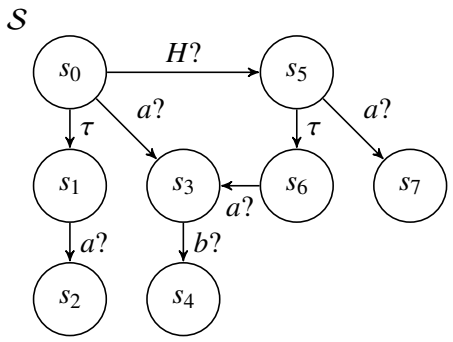


Figura 3.13.: $\mathcal{S}$ no es SIR-NNI/SIR-SNNI pero es $(\bigwedge_{RT,\not\sharp})$-NNI/$(\bigwedge_{RT,\not\sharp})$-SNNI.

observabilidad. En [56] se presenta una versión de los algoritmos para el chequeo y síntesis de sistemas BNNI/BSNNI. Si bien los algoritmos presentados en [56] y los algoritmos aquí presentados se basan en el algoritmo para verificar bisimulación de [30] los resultados en el caso de síntesis varían un poco. El algoritmo de síntesis de [56] presenta cuatros posibles resultados: los tres que están en la presente versión y un cuarto resultado denominado *indeterminado*. Para este resultado no es posible saber si se puede o no sintetizar una interfaz segura. La diferencia se debe al comportamiento simétrico de la bisimulación débil: para evitar la ejecución de acciones que no se sincronizan se puede eliminar transiciones de entrada altas. Eliminar estas transiciones, la cuales son abstraídas antes de verificar la bisimulación débil, puede producir que al saturar la nueva interfaz los estados de la misma ejecuten menos transiciones. Esto tiene como consecuencia la posible aparición de nuevos problemas de sincronización.

3.5.2. Conclusiones

Definir las propiedades de no interferencia basadas en nociones de observabilidad hace explícita la maquinaria con la que cuenta el atacante al momento de estudiar el sistema. No tener en cuenta esto puede generar errores de interpretación e inexactitudes al modelar el sistema y su entorno. Un ejemplo de esto ocurre en [57] al justificar la necesidad de las variantes de no interferencia SIR-NNI/SIR-SNNI. En [57] argumentamos que las propiedades BSNNI/BNNI [56] permiten ciertas filtraciones de información y para subsanar este problema se introducen las propiedades SIR-NNI/SIR-SNNI. Pero esto es natural que suceda pues las propiedades deben usarse en distintos contextos. Esto es claro al ver las nociones de observabilidad que se utilizan para modelar las propiedades.

$$WB = \{a^O, \top, \tau, \varepsilon, \vee, \rightleftarrows, \wedge, \neg, \} \qquad SIR = \{a^O, \top, \bigvee, \tau, \rightleftarrows, RT, \bigwedge, \eta, b\}$$

Las variantes BSNNI/BNNI pueden utilizarse en un contexto donde la comunicación se realiza utilizando hand shaking, mientras que las variantes SIR-NNI/SIR-SNNI deben ser utilizadas en un contexto donde no. Esto no es tenido en cuenta en [57].

Este enfoque también brinda un marco mucho más versátil para refinar las semánticas utilizadas. Por ejemplo, la presentación original de la relación SIR (Def 2.1.11) no hace evi-

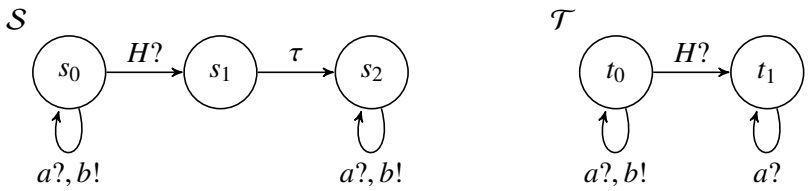


Figura 3.14.: El $\mathcal{S}$ no es SIR-NNI/SIR-SNNI, mientras que $\mathcal{T}$ lo es. Sin embargo puede argumentarse que existe una filtración de información en $\mathcal{T}$: la ejecución de la acción $b!$ es un indicio de que $H?$ no ha sido ejecutada.

dente que el usuario cuenta con una gran maquinaria para realizar copias. Sin embargo, esto queda claro al ver la representación de la relación SIR utilizando nociones de observabilidad: $\{\bigwedge, \eta, b\} \subseteq SIR$. Esta gran maquinaria quizás es demasiado para algunos contextos, por lo tanto ésta se podría reducir eliminando los tipos η y b. Sea $\bigwedge_{RT,\not\Leftrightarrow}$ la noción de observabilidad que se obtiene luego de eliminar los tipos η y b i.e. $\bigwedge_{RT,\not\Leftrightarrow} = \{a^O, \top, \bigvee, \tau, \rightleftarrows, RT, \bigwedge\}$. Esta nueva noción es más débil que SIR dado que no permite distinguir las transiciones internas previas a la ejecución de una acción de entrada. Un ejemplo de esto se presenta en la Figura 3.13: $\mathcal{S}$ es $(\bigwedge_{RT,\not\Leftrightarrow})$-NNI/$(\bigwedge_{RT,\not\Leftrightarrow})$-SNNI, pero no es SIR-NNI/SIR-SNNI.

Así cómo es posible debilitar una relación para que la propiedad se ajuste mejor a los requerimientos, también es posible fortalecer la relación para obtener el mismo resultado. La interfaz $\mathcal{S}$ en la Figura 3.14 no es $(\bigwedge_{RT,\not\Leftrightarrow})$-SNNI: la observación $\emptyset\top$ no es posible antes de la ejecución de la acción alta $H?$ pero si después de ésta, i.e. $\emptyset\top \in O_{\mathrm{SIR}}(\mathcal{S}/A^h) - O_{\mathrm{SIR}}(\mathcal{S}\backslash A^h)$. Por otro lado, la interfaz $\mathcal{T}$ en la Figura 3.14 sí es $(\bigwedge_{RT,\not\Leftrightarrow})$-SNNI. Si bien $\mathcal{T}$ es catalogado como seguro utilizando la propiedad $\bigwedge_{RT,\not\Leftrightarrow}$-SNNI se puede argumentar que existe una filtración de información: la ejecución de la acción $b!$ en $\mathcal{T}$ implica que la ejecución $H?$ no ha sido ejecutada. Para solucionar esto sólo basta extender la noción de observabilidad agregando el tipo $\neg$ y de esta forma obtener la noción de observabilidad

$$\sim_{RT,\not\Leftrightarrow} = \{a^O, \top, \bigvee, \tau, \rightleftarrows, RT, \bigwedge, \neg\}$$

Esta nueva semántica permite detectar la filtración de información que existe en $\mathcal{T}$: $\mathcal{T}$ no es seguro $(\sim_{RT,\not\Leftrightarrow})$-SNNI debido a que $t_0 \not\sim_{RT,\not\Leftrightarrow} t_1$. Esta nueva semántica es más fuerte que la bisimulación débil. Por ejemplo $\mathcal{S}$ es BSNNI pero no es $(\sim_{RT,\not\Leftrightarrow})$-SNNI dado que $s_0 \not\sim_{RT,\not\Leftrightarrow} s_1$.

Capítulo 4

Especificación de sistemas de transiciones probabilísticas

En los capítulos anteriores nos enfocamos en el estudio de la semántica de los sistemas. En particular, en el capítulo 2 estudiamos la semántica desde un punto de vista del poder de observación de quién interactúa con el proceso. En el capítulo 3 utilizamos estas nociones para identificar si un proceso es seguro o no. En un caso utilizamos un IA, en el otro un ISS, los cuales son variantes de sistemas de transiciones etiquetadas.

Si bien los sistemas de transiciones son los objetos matemáticos que se utilizan habitualmente para comprender el comportamiento de un proceso, no es el lenguaje habitual de modelado. Para especificar o diseñar procesos se utilizan lenguajes de especificación o modelado, que permiten la construcción de un sistema a través de la composición de sistemas más pequeños. En este caso, la semántica de los procesos, i.e. su sistema de transiciones, se construye siguiendo la estructura del lenguaje a través de reglas de derivación guiadas por los operadores que conforman dicho lenguaje. Esta técnica se denomina semántica operacional estructurada [65, 66].

En el contexto de esta tesis nos limitaremos a trabajar con lenguajes simples definidos a través de la signatura de un álgebra. De esta manera, un término algebraico representa un proceso. Éstos se construyen de forma recursiva a partir de una *signatura* la cual está constituida por un conjunto de *(nombres de) funciones*. Cada función posee una aridad específica y de éstas se distinguen las funciones de aridad cero, las cuales son denominadas *constantes*. Ejemplos de este tipo de lenguajes son ACP [6], CSP [47, 48] y CCS [62]. En este contexto, la semántica operacional estructurada define el comportamiento de un término, que denota la aplicación de una función, en base al comportamiento de los parámetros que utiliza la función. En [3] se presenta una introducción completa a este tema.

Utilicemos un ejemplo para clarificar el enfoque. Sea una signatura compuesta por las constantes a, a' y la función f de aridad uno. Esta signatura permitirá generar el siguiente conjunto infinito de términos:

$$\{a, a', f(a), f(a'), f(f(a)), f(f(a')), f(f(f(a))), f(f(f(a'))) \ldots\}$$

Cada uno de estos términos representa un estado en un sistema de transición. Dado los términos, sólo resta definir las transiciones que puede realizar cada uno. Para esto se utiliza un conjunto de reglas como las que se presentan a continuación:

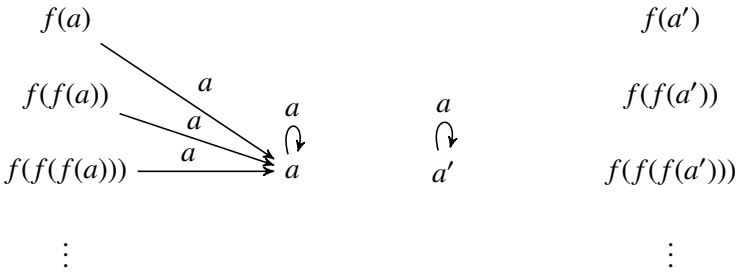
$$\frac{}{a \xrightarrow{a} a} \qquad \frac{}{a' \xrightarrow{a} a'} \qquad \frac{x \xrightarrow{a} a}{f(x) \xrightarrow{a} a}$$

La parte superior de una regla se llama *premisa*, mientras que la inferior *conclusión*. En la premisa se especifica que transiciones debe realizar el sistema para poder realizar la transición que se expresa en la conclusión. Las primeras dos reglas no tienen premisas; a este tipo de regla se la denomina *axioma*. Los dos axiomas que se presentan describen el siguiente comportamiento en los estados a y a':

La última regla no sólo posee premisas sino que también posee otro elemento del que haremos uso y que aún no ha sido introducido. Este elemento son las *variables*: la última regla utiliza una variable x, tanto en la premisa como en la conclusión. Las reglas que poseen variables no pueden ser utilizadas directamente. Previo a esto, las variables deben ser reemplazadas por términos sin variables. Esto se realiza mediante una función denominada *substitución*. Las substituciones pueden definirse de forma arbitraria, por lo tanto, de la última regla y utilizando distintas substituciones podemos obtener las siguientes reglas cerradas, i.e. sin variables:

$$\frac{a \xrightarrow{a} a}{f(a) \xrightarrow{a} a}, \quad \frac{a' \xrightarrow{a} a}{f(a') \xrightarrow{a} a}, \quad \frac{f(a) \xrightarrow{a} a}{f(f(a)) \xrightarrow{a} a}, \quad \frac{f(a') \xrightarrow{a} a}{f(f(a')) \xrightarrow{a} a}, \quad \frac{f(f(a)) \xrightarrow{a} a}{f(f(f(a))) \xrightarrow{a} a}, \quad \cdots$$

Cómo ya se menciono, siempre que el sistema de transición pueda realizar las transiciones que se indican en la premisa, el sistema deberá también realizar la transición que se indica en la conclusión. Por lo tanto, la signatura y las reglas presentadas pueden utilizarse para representar el siguiente sistema de transición:



La signatura y las reglas definen una especificación de un sistema de transición, donde la signatura especifica los estados y las reglas especifica las transiciones del sistema.

La necesidad de estudiar aspectos funcionales y no funcionales del comportamiento de los sistemas, por ejemplo performance o confiabilidad, llevó a desarrollar sistemas más complejos. Entre los sistemas más complejos se destacan los *sistemas de transiciones probabilistas*. En estos sistemas, luego de ejecutar una transición no se alcanza un estado sino que se alcanzan un conjunto de estados, cada estado con una probabilidad particular. Este tipo de sistemas se pueden representar utilizando distintas variantes [11,21,40,45,55,67,71,72], donde cada variante modela distintas características del sistema de transición. Este capítulo está dedicado a la especificación de sistemas de transiciones probabilistas al estilo de *Segala* [71,72]. Este tipo de modelo soporta no determinismo con transiciones de la forma $t \xrightarrow{a} \pi$ donde t es un término que representa un estado, a una acción y π es una distribución de probabilidad sobre estados.

En este nuevo contexto, la semántica operacional estructurada se extiende para contemplar los nuevos aspectos cuantitativos. Por ejemplo, en la siguiente sección se presentarán reglas con la siguiente forma:

$$\frac{f(a,x) \xrightarrow{a} \mu \quad \mu(\{y_0, y_1\}) > 0{,}5 \quad y_0 \overset{b}{\not\rightarrow}}{x \xrightarrow{a} 0{,}5\mu + 0{,}5\pi}$$

En esta regla, las expresiones x, y_0, y_1 y $f(a,x)$ son expresiones que se reducirán a términos que representan estados del sistema. Las expresiones están compuestas por las variables x, y_0 e y_1, y por elementos de la signatura que contiene a f y a. Por otro lado, las expresiones μ y $0{,}5\mu + 0{,}5\pi$ son expresiones que se reducirán a distribuciones sobre términos cerrados, i.e. que no poseen variables. En estas expresiones, μ es una variable de distribución que sólo puede ser reemplazada por expresiones que finalmente se reducen a una distribución de probabilidad; por otro lado π es una distribución de probabilidad particular. Estos términos serán denominados *términos de distribuciones*.

Las reglas que se utilizarán no sólo soportan *premisas positivas* ($f(a,x) \xrightarrow{a} \mu$) sino que también soportan *premisas negativas* ($y_0 \overset{b}{\not\rightarrow}$) y *premisas cuantitativas* ($\mu(\{y_0, y_1\}) > 0{,}5$). El utilizar premisas negativas enriquece el poder de expresión de las especificaciones pero al mismo tiempo introduce algunos problemas para asignar un sistema de transición particular a una especificación. Distintas técnicas se han presentado para sortear este problema en un contexto sin probabilidades [16, 39]. En este Capítulo se extenderá la técnica basada en *pruebas bien soportadas* [39] a un contexto con probabilidades. Para esto se necesitará extender la noción de sistema de transición: un sistema de transición estará compuesto por 3-uplas de la forma (t, a, π) y por 2-uplas (t, b). Las 3-uplas, como es usual, denotarán que el estado t puede ejecutar la acción a y el estado siguiente se elige con una distribución π, i.e. $t \xrightarrow{a} \pi$; mientras que las 2-uplas denotarán explícitamente que el estado t no ejecuta la acción b, i.e. $t \overset{b}{\not\rightarrow}$. Esto cambia con respecto a algunos enfoques donde $t \overset{b}{\not\rightarrow}$, denota que no existe π' tal que $t \xrightarrow{b} \pi'$. Los sistemas de transiciones que utilizaremos son denominados *sistemas de transiciones con tres valores* porque dado un estado t y una acción a existen tres posibilidades: existe π tal que (t, a, π) es una 3-upla del sistema, (t, a) es una 2-upla del sistema o ninguna de las dos opciones anteriores es válida. Por último, la inclusión de premisas cuantitativas también incrementan el poder expresivo de las especificaciones pero sin agregar ninguna complejidad extra.

A partir de aquí dejaremos de lado la diferenciación entre acciones de entrada y de salida,

dado que las acciones pasarán a ser un objeto meramente sintáctico.

4.1. Preliminares

En este sección introducimos la definiciones de los objetos matemáticos que se utilizaran a los largo de este capítulo y los subsiguientes. Estos objetos son tres: *términos*, *términos de distribuciones* y *substituciones*.

Los términos son expresiones algebraicas que se construyen a partir de una signatura. Los términos de distribuciones son también expresiones algebraicas pero estas se construyen siempre de la misma manera; se utilizan para representar distribuciones de probabilidad, las cuales definen la probabilidad de alcanzar un estado particular luego de la ejecución de una acción. Ambos tipos de términos pueden contener variables, en cuyo caso se denominan *abiertos*, o estar completamente definidos (sin variables) en cuyo caso se denominan *cerrados*.

Las substituciones son funciones que reemplazan variables por términos. Estos términos pueden ser cerrados o abiertos. Cómo se mostró en la introducción del capítulo, estas se utilizarán para instanciar las reglas con las que se derivan el sistema de transición.

Términos

Un lenguaje de modelado o de especificación está definido a través de una *signatura*. La signatura define los distintos constructores del lenguaje a través de un conjunto F de símbolos de función, y de una función r que asigna a cada símbolo de función su aridad.

Supongamos la existencia de un conjunto infinito $\mathcal{V}$ de variables (de término). Los nombres $x, y, z, x', x_0, x_1, \ldots$ serán utilizados para denotar variables en $\mathcal{V}$.

Definición 4.1.1. *Una* signatura *es una estructura* $\Sigma = (F, \mathsf{r})$*, donde*

(i) F *es el conjunto de* nombres de funciones *tal que* $F \cap \mathcal{V} = \emptyset$.

(ii) $\mathsf{r} : F \to \mathbb{N}_0$ *es la* función de grado*;* f *es una* constante *si* $f \in F$ *y* $\mathsf{r}(f) = 0$.

A partir de un conjunto de variables y una signatura se definen de forma recursiva los *términos*.

Definición 4.1.2. *Sea* $W \subseteq \mathcal{V}$ *un conjunto de variables. El conjunto* términos *sobre* W*, notación* $T(\Sigma, W)$*, es el menor conjunto tal que:*

- $W \subseteq T(\Sigma, W)$*,* y
- *si* $f \in F$ *y* $t_1, \cdots, t_{\mathsf{r}(f)} \in T(\Sigma, W)$*, entonces* $f(t_1, \cdots, t_{\mathsf{r}(f)}) \in T(\Sigma, W)$.

El conjunto $T(\Sigma, \emptyset)$ es abreviado por $T(\Sigma)$; los elementos de $T(\Sigma)$ son llamados *términos cerrados*. El conjunto $T(\Sigma, \mathcal{V})$ es abreviado por $\mathbb{T}(\Sigma)$; los elementos de $\mathbb{T}(\Sigma)$ son llamados *términos abiertos*. $Var(t) \subseteq \mathcal{V}$ es el conjunto de variables que ocurren en el término abierto t.

Ejemplo 4.1.1. *Introducimos una signatura de un álgebra probabilística que incluye la mayoría de los operadores más representativos. Para esto suponemos la existencia de un conjunto de etiquetas* $\mathcal{L}$*. Entonces, nuestra signatura (la cual será usada a lo largo del trabajo) contiene:*

- *dos constantes,* $\mathbf{0}$ *(stop) y* ε *(skip);*
- *una familia de operadores de prefijo* $a.([p_1]\_ \oplus \cdots \oplus [p_n]\_)$ *con aridad n,* $a \in \mathcal{L}$*,* $n \geq 1$*,* $p_1, \ldots, p_n \in (0, 1]$ *tales que* $\sum_{i=1}^{n} p_i = 1$ *(dado los términos* $t_1, \ldots, t_n$*, escribiremos* $a.\sum_{i=1}^{n}[p_i]t_i$ *para simplificar la notación);*
- *los operadores binarios* $\_ + \_$ *(composición alternativa o suma),* $\_ ; \_$ *(composición secuencial), y, para cada* $B \subseteq \mathcal{L}$*,* $\_ \|_B \_$ *(composición paralela); y*
- *un operador unario* $\mathsf{U}(\_)$ *que denominaremos* unreach.

El proceso $\mathbf{0}$ *denota el proceso que ha llegado a un punto en el cual no puede continuar ejecutando acciones. El proceso* ε *indica la terminación satisfactoria de la ejecución. El término* $a.\sum_{i=1}^{n}[p_i]t_i$ *denota el proceso que puede realizar la acción a y continuar con la ejecución del proceso* t_i *con probabilidad* p_i*. El operador* $t \|_B t'$ *se comporta como el operador de paralelización de CSP. El operador unreach* $\mathsf{U}(t)$ *puede ejecutar una acción â y detenerse si existe una ejecución probabilista (o scheduler) desde t en la cual la acción â nunca pueda ejecutarse (o más formalmente, no se ejecuta con probabilidad 1). El término* $t + t'$ *indica la ejecución de* t *o* t' *(pero no la de ambos) y* $t; t'$ *representa la ejecución de t seguida de* t'*.* □

Por último, introducimos las definición de *contexto*. Decimos que una expresión $C[x_1, \ldots, x_n]$ es un *contexto* si $C[x_1, \ldots, x_n]$ es un término en el cual al menos las variables $x_1, \ldots, x_n$ aparecen. Como es habitual, $C[t_1, \ldots, t_n]$ denota el término que se obtiene luego de reemplazar cada ocurrencia de las variables x_i por los términos t_i.

Términos de Distribuciones

Sea $\Delta(T(\Sigma))$ el conjunto de todas las funciones de probabilidad (discretas) sobre $T(\Sigma)$. Sean $\pi, \pi', \pi_0, \pi_1, \ldots$ elementos sobre $\Delta(T(\Sigma))$. Dada una función de distribución $\pi \in \Delta(T(\Sigma))$ y $T \subseteq T(\Sigma)$ un conjunto de términos cerrados, definimos $\pi(T) = \sum_{t \in T} \pi(t)$. Para un término cerrado $t \in T(\Sigma)$, denotamos con δ_t la distribución de Dirac (asociada a t), la cual está definida por

$$\delta_t(t') = \begin{cases} 1 & \text{si } t = t' \\ 0 & \text{si } t \neq t' \end{cases}$$

Por otro lado, el producto $\prod_{i=1}^{n} \pi_i$ de distribuciones está definido por

$$(\prod_{i=1}^{n} \pi_i)(t_1, \ldots, t_n) = \prod_{i=1}^{n} \pi_i(t_i)$$

En particular, si $n = 0$, $(\prod_{j \in \emptyset} \nu_j) = \delta_{()}$ es la distribución que asigna 1 a la tupla vacía. Sea $f : T(\Sigma)^n \to T(\Sigma)$ y recordemos que $f^{-1}(t') = \{\vec{t} \in T(\Sigma)^n \mid f(\vec{t}) = t'\}$. Entonces $(\prod_{i=1}^{n} \pi_i) \circ f^{-1}$ es una distribución de probabilidad bien definida sobre términos cerrados.

Dado un término abierto $t \in \mathbb{T}(\Sigma)$ diremos que δ_t es una *distribución de Dirac instanciable*. Es decir, δ_t es un símbolo que toma el valor $\delta_{t'}$ cuando las variables en t son sustituidas de forma tal que t se transforma en el término cerrado $t' \in T(\Sigma)$. Sea $\mathcal{D} = \{\delta_t : t \in \mathbb{T}(\Sigma)\}$ el conjunto de distribuciones de Dirac instanciables.

En su expresión más básica, un *término de distribución* puede ser una variable (que sólo podrá ser reemplazada por otro términos de distribución) o una distribución de Dirac instanciable. Además existirán dos formas de construir nuevos términos de distribuciones. La primera corresponde a la combinación convexa de términos de distribuciones. La segunda corresponde a la productoria de términos de distribuciones.

Como ya mencionamos, necesitamos variables las cuales sólo pueden ser reemplazadas por otro término de distribución, estas variables se denominarán *variables de distribución*. Sea $\mathcal{M}$ un conjunto infinito de variables de distribución y sean $\mu, \mu', \mu_0, \mu_1, \ldots$ elementos del conjunto $\mathcal{M}$.

Definición 4.1.3 (Términos de Distribuciones)**.** *Sea $D \subseteq \mathcal{M}$ un conjunto de variables de distribuciones y sea $V \subseteq \mathcal{V}$ un conjunto de variables de términos. El conjunto de* términos de distribuciones *sobre D y V, notación $\mathsf{DT}(\Sigma, D, V)$, es el menor conjunto que satisface:*

1. $D \cup \{\delta_t : Var(t) \subseteq V\} \subseteq \mathsf{DT}(\Sigma, D, V)$,
2. $\sum_{i \in I} p_i \theta_i \in \mathsf{DT}(\Sigma, D, V)$ *si* $\theta_i \in \mathsf{DT}(\Sigma, D, V)$ *y* $p_i \in (0, 1]$ *con* $\sum_{i \in I} p_i = 1$, *y*
3. $(\prod_{1 \leq n \leq N} \theta_i) \circ g^{-1} \in \mathsf{DT}(\Sigma, D, V)$ *si* $\theta_n \in \mathsf{DT}(\Sigma, D, V)$ *y* $g : T(\Sigma)^N \to T(\Sigma)$ *donde la función g se define mediante un contexto C, i.e.* $g(z_0, \ldots, z_N) = C[z_0, \ldots, z_n]$.

El conjunto $\mathsf{DT}(\Sigma, \emptyset, \emptyset)$, abreviado por $\mathsf{DT}(\Sigma)$, es el conjunto de las distribuciones discretas sobre términos cerrados. El conjunto $\mathsf{DT}(\Sigma, \mathcal{M}, \mathcal{V})$ es abreviado por $\mathbb{DT}(\Sigma)$.

La Def. 4.1.3 es bastante directa, el único punto que merece una pequeña atención es la función g^{-1} en el punto 3. La función inversa de g se utiliza para descomponer un término, que proviene de un contexto específico, en la tupla de términos que reemplazados en el contexto forman el término original. Estas tuplas son luego "medidas" por la productoria de distribuciones. De esta manera, la composición permite definir una nueva distribución sobre un término compuesto a partir de las distribuciones de las componentes. Esto quedará más claro en el siguiente ejemplo.

Ejemplo 4.1.2. *Sea, $t_0, t_1, t_2 \in T(\Sigma)$. A continuación se definen distribuciones sobre estos términos utilizando los términos de distribuciones.*

- *Supongamos que π_0 y π_1 son dos distribuciones sobre términos (cerrados) tales que:*

$$\begin{array}{lll} \pi_0(t_0) = 0{,}2 & \pi_0(t_1) = 0{,}3 & \pi_0(t_2) = 0{,}5 \\ \pi_1(t_0) = 0{,}3 & \pi_1(t_1) = 0{,}3 & \pi_1(t_2) = 0{,}4 \end{array}$$

  Luego π_0 y π_1 pueden expresarse como términos de distribuciones (cerrados) mediante las distribuciones de Dirac y la combinación convexa:

$$\pi_0 = 0{,}2 \cdot \delta_{t_0} + 0{,}3 \cdot \delta_{t_1} + 0{,}5 \cdot \delta_{t_2} \qquad \pi_1 = 0{,}3 \cdot \delta_{t_0} + 0{,}3 \cdot \delta_{t_1} + 0{,}4 \cdot \delta_{t_2}$$

- *El término de distribución abierto $\theta = 0{,}5 \cdot \pi_0 + 0{,}5 \cdot \mu$ denota una distribución que se comporta con una probabilidad de* 0,5 *cómo la distribución π_0 y con una probabilidad de* 0,5 *de manera aun no definida, por esta razón se utiliza la variable μ.*

- *El término de distribución abierto* $(\theta \times \pi_1) \circ g^{-1}$ *donde* $g(t, t') = t \parallel_B t'$*, denota una distribución sobre los término cerrados que a un término* $t \parallel_B t'$ *le asigna el producto entre la probabilidad de* t *con respecto a la distribución (a definir)* θ *del ejemplo anterior y la probabilidad de* t' *con respecto a la distribución* π_1*. Por otro lado, a un término que no satisface esta forma le asigna cero. Por ejemplo, calculemos la probabilidad de los términos* $t_0 \parallel_B t_1$ *y* t_2*:*

$$\begin{aligned}(\theta \times \pi_1) \circ g^{-1}(t_0 \parallel_B t_1) &= (\theta \times \pi_1)(\{(t_0, t_1)\}) \\ &= \theta(t_0) \cdot \pi_1(t_1) \\ (\theta \times \pi_1) \circ g^{-1}(t_2) &= (\theta \times \pi_1)(\emptyset) \\ &= 0\end{aligned}$$

□

Para simplificar la notación y dar una noción más algebraica, en casos como el anterior y cuando sea conveniente, escribiremos por ejemplo $\theta \parallel_B \pi_1$ en lugar de $(\theta \times \pi_1) \circ g^{-1}$

El siguiente lema se demuestra en forma directa. El mismo brinda una presentación más compacta de un término de distribución que no es una variable de distribución, ni una distribución de Dirac instanciable. Esto simplificará el análisis de caso inductivos en algunas pruebas.

Lema 4.1.1. *Sea* $\theta \in \mathsf{DT}(\Sigma, D, V)$ *tal que* $\theta \notin D \cup \{\delta_t : Var(t) \subseteq V\}$ *entonces*

$$\theta = \textstyle\sum_{i \in I} p_i (\prod_{n_i \in N_i} \theta_{n_i}) \circ g_i^{-1}$$

donde los $p_i \in (0, 1]$ *son tales que* $\sum_{i \in I} p_i = 1$*, cada* g_i *es una función tal que* $g_i : T(\Sigma)^{N_i} \to T(\Sigma)$*, y* $\theta_{n_i} \in \mathsf{DT}(\Sigma, D, V)$

Substituciones

Una *substitución* es un mapeo que asigna términos a variables. En nuestro caso particular necesitamos extender esta noción a variables probabilísticas.

Definición 4.1.4. *Una* substitución ρ *es un mapeo* $(\mathcal{V} \cup \mathcal{M}) \to (T(\Sigma, W) \cup \mathsf{DT}(\Sigma, D))$ *tal que* $\rho(x) \in T(\Sigma, W)$ *siempre que* $x \in \mathcal{V}$*, y* $\rho(\mu) \in \mathsf{DT}(\Sigma, D)$ *siempre que* $\mu \in \mathcal{M}$*.*

Una substitución ρ se extiende a términos y a conjuntos de éstos de la forma usual y a distribuciones de Dirac instanciables y a términos de distribución de la siguiente manera:

$$\rho(\delta_t) = \delta_{\rho(t)} \qquad \rho(\textstyle\sum_{i \in I} p_i \theta_i) = \sum_{i \in I} p_i \rho(\theta_i) \qquad \rho((\prod_{1 \leq i \leq n} \theta_n) \circ g^{-1}) = (\prod_{1 \leq i \leq n} \rho(\theta_n)) \circ g^{-1}$$

Diremos que la substitución ρ es *cerrada* si $\rho(x) \in T(\Sigma)$ siempre que $x \in \mathcal{V}$ y $\rho(\mu) \in \Delta(\Sigma)$ siempre que $\mu \in \mathcal{M}$. Caso contrario diremos que es *abierta*. Notemos que si la substitución ρ es cerrada, para todo $\theta \in \mathsf{DT}(\Sigma, D)$, $\rho(\theta) \in \Delta(\Sigma)$.

4.2. Especificación de sistemas de transiciones probabilistas

Un sistema de transiciones probabilistas sobre los términos cerrados de una signatura dada, quedan definidos por un sistema de transición. Este sistema de transición está compuesta por 3-uplas en $T(\Sigma) \times A \times \Delta(T(\Sigma))$ y 2-uplas en $T(\Sigma) \times A$. Una 3-upla (t, a, π) denota que el estado t puede ejecutar la acción a y el estado siguiente se elige con una distribución π; mientras que una 2-upla (t, b) denota explícitamente que t no puede ejecutar la acción b. Este enfoque es similar al de autómatas probabilísticos [71] pero con el agregado de poder especificar explícitamente las acciones que no son realizables por un proceso dado. De esta forma el sistema de transición está definido como un sistema trivaluado donde ciertas transiciones quedan sin especificar: si el sistema de transición no contiene el par (t, a), ni una terna (t, a, π), para alguna distribución π, luego el efecto de realizar a en el proceso t queda sin definir. Finalmente nuestro objetivo será definir sistemas completos (i.e. sistemas donde la realización o no de una acción a por parte de t está siempre definida). Notar que en este caso, el sistema de transición queda completamente definido por el conjunto de 3-uplas.

Definición 4.2.1. *Sea Σ una signatura y A un conjunto de etiquetas. Una* sistema de transiciones probabilistas *(PTS del inglés,* Probabilistic Transitions System*) es un conjunto $\rightarrow \subseteq PTr(\Sigma, A)$, donde $PTr(\Sigma, A) = (T(\Sigma) \times A \times \Delta(T(\Sigma))) \cup (T(\Sigma) \times A)$. Se denotará la tupla $(t, a, \pi) \in \rightarrow$ con $t \xrightarrow{a} \pi$ y la tupla $(t, a) \in \rightarrow$ con $t \xrightarrow{a}\!\!\!\!\!/\;\;$.*

Una especificación de un sistema de transición está compuesto por una signatura, un conjunto de acciones y un conjunto de reglas. La función más básica de una especificación es caracterizar un sistema de transición particular (Def. 4.2.3). El enfoque provee una caracterización algebraica para la especificación, el cual es similar al utilizado en [16, 43, 44].

Por otro lado, una especificación más la maquinaria adecuada, permitirán derivar un sistema de transiciones que cumplan la caracterización que la especificación expresa, i.e. el sistema de transiciones satisface las reglas. Esto se desarrolla en la Sección 4.3.

Definición 4.2.2. *Una* especificación de un sistema de transiciones probabilistas *(PTSS, del inglés* Probabilistic Transition System Specification*) es una estructura $P = (\Sigma, A, R)$ donde $\Sigma = (F, r)$ es una signatura, A es un conjunto de acciones y R es un conjunto de reglas de la forma:*

$$\frac{\{t_k \xrightarrow{a_k} \mu_k : k \in K\} \cup \{t_l \xrightarrow{b_l}\!\!\!\!\!/\;\;\; : l \in L\} \cup \{\theta_j(W_j) \gtrless_j q_j : j \in J\}}{t \xrightarrow{a} \theta}$$

en el cual K, L, J son conjuntos de índices (posiblemente vacíos), $t, t_k, t_l \in \mathbb{T}(\Sigma)$, $a, a_k, b_l \in A$, $\mu_k \in \mathcal{M}$, $W_j \subseteq \mathcal{V}$, $\gtrless_j \in \{>, \geq, <, \leq\}$, $q_j \in [0, 1]$ y $\theta_j, \theta \in \mathbb{DT}(\Sigma)$

Una expresión de la forma $t \xrightarrow{a} \theta$, se denomina *literal positivo*. Similarmente llamamos *literal negativo* y *literal cuantitativo* a las expresiones $t \xrightarrow{a}\!\!\!\!\!/\;\;$ y $\theta(W) \gtrless p$, respectivamente ($t \in \mathbb{T}(\Sigma), a \in A, \theta \in \mathbb{DT}(\Sigma), W \subseteq Var \cup T(\Sigma)$ y $p \in [0, 1]$). Un literal ψ es *no-cuantitativo* si este es un literal positivo o negativo. Para un literal no-cuantitativo ψ, $src(\psi)$ denota el término a la izquierda de la flecha; para un literal positivo ψ, $trg(\psi)$ denota el término a la derecha de la flecha. Para toda regla $r \in R$, los literales sobre la linea se denominan *premisas*. El conjunto de premisas

de r se denota con prem(r); el literal bajo la linea es llamado *conclusión*, notación conc(r). Denotamos con pprem(r), nprem(r) y qprem(r) el conjunto de premisas positivas, negativas y cuantitativas de la regla r respectivamente. Una regla r es llamada *positiva* si nprem(r) = $\emptyset$. Un PTSS es llamado *positivo* si sólo posee reglas positivas. Una regla r sin premisas es un *axioma*. En general, permitiremos que el conjunto de premisas positivas, negativas y cuantitativas sean conjuntos infinitos.

Las substituciones se utilizarán para definir instancias de las reglas del PTSS, las cuales permitirán definir los sistemas de transiciones probabilistas. Dada una substitución ρ, la misma se extiende a literales de la siguiente manera:

$$\rho(t \xrightarrow{a} \theta) = \rho(t) \xrightarrow{a} \rho(\theta) \qquad \rho(t \not\xrightarrow{a}) = \rho(t) \not\xrightarrow{a} \qquad \rho(\theta(W) \gtrless p) = \rho(\theta)(\rho(W)) \gtrless p$$

Luego, la noción de substitución se extiende a reglas de la forma esperada. Decimos que r' es una instancia (cerrada) de la regla r si existe una substitución (cerrada) ρ tal que $r' = \rho(r)$.

Decimos que ρ es una *substitución genuina de r* si para toda premisa cuantitativa $\rho(\theta(W) \gtrless p)$ de r se satisface $\rho(\theta(w)) > 0$ para todo $w \in W$. Entonces, si ρ es genuina, todo término en $\rho(W)$ se encuentra en el soporte de $\rho(\theta)$. Las substituciones genuinas evitan introducir términos espurios en la premisa cuantitativa. Solamente utilizaremos este tipo de substituciones a lo largo del trabajo.

La intuición detrás de una especificación es bastante simple. Si un sistema de transición está asociado a un PTSS P se espera

1. que respete las reglas de P, esto es, siempre que las premisas de una instancia cerrada de una regla en P pertenezcan a el sistema de transición, entonces también su conclusión pertenecerá; y

2. que no incluya más transiciones que las que explícitamente se justifican i.e., una transición está definida sí y sólo sí existe una instancia cerrada de una regla tal que sus premisas se encuentran en el sistema de transición.

La primera noción corresponde a la noción de modelo, y la segunda a la noción de transición soportada. Antes de definir formalmente estas nociones, introducimos nueva notación. Dado un sistema de transición $\rightarrow \subseteq PTr(\Sigma, A)$, un literal positivo $t \xrightarrow{a} \pi$ *vale en* $\rightarrow$, notación $\rightarrow \models t \xrightarrow{a} \pi$, si $(t, a, \pi) \in \rightarrow$. Un literal negativo $t \not\xrightarrow{a}$ *vale en* $\rightarrow$, notación $\rightarrow \models t \not\xrightarrow{a}$, si $(t, a) \in \rightarrow$. Un literal cuantitativo $\pi(T) \gtrless p$ *vale en* $\rightarrow$, notación $\rightarrow \models \pi(T) \gtrless p$ precisamente cuando $\pi(T) \gtrless p$. Notemos que la validez de un literal cuantitativo no depende del sistema de transición. De todas maneras utilizamos esta notación porque será conveniente al momento de generalizar. Dado un conjunto de literales H, escribimos $\rightarrow \models H$ si $\forall \phi \in H : \rightarrow \models \phi$.

Definición 4.2.3. *Sea $P = (\Sigma, A, R)$ un PTSS y sea $\rightarrow \subseteq PTr(\Sigma, A)$ un sistema de transiciones probabilistas, entonces $\rightarrow$ es* un modelo soportado de *P si:*

$$\psi \in \rightarrow \quad \Leftrightarrow \quad \text{existe una regla } \frac{H}{\chi} \in R \text{ y una substitución genuina } \rho \text{ tal que } \rho(\chi) = \psi \text{ y} \rightarrow \models \rho(H)$$

Cuando se cumple la implicación $\Leftarrow$ decimos que $\rightarrow$ es un modelo *de P. Cuando se cumple la implicación $\Rightarrow$ decimos que $\rightarrow$ es* soportado por *P.*

$$\varepsilon \xrightarrow{\surd} \delta_{\mathbf{0}} \qquad a.\sum_{i=1}^{n}[p_i]x_i \xrightarrow{a} \sum_{i=1}^{n} p_i\delta_{x_i} \qquad \frac{x \xrightarrow{a} \mu}{x+y \xrightarrow{a} \mu} \qquad \frac{y \xrightarrow{a} \mu}{x+y \xrightarrow{a} \mu}$$

$$\frac{x \xrightarrow{a} \mu}{x;y \xrightarrow{a} \mu;\delta_y}\ a \neq \surd \qquad \frac{x \xrightarrow{\surd} \mu \quad y \xrightarrow{a} \mu'}{x;y \xrightarrow{a} \mu'} \qquad \frac{x \xrightarrow{a} \mu \quad y \xrightarrow{a} \mu'}{x \parallel_B y \xrightarrow{a} \mu \parallel_B \mu'}\ a \in B\backslash\{\surd\}$$

$$\frac{x \xrightarrow{a} \mu}{x \parallel_B y \xrightarrow{a} \mu \parallel_B \delta_y}\ a \notin B \cup \{\surd\} \qquad \frac{y \xrightarrow{a} \mu}{x \parallel_B y \xrightarrow{a} \delta_x \parallel_B \mu}\ a \notin B \cup \{\surd\} \qquad \frac{x \xrightarrow{\surd} \mu \quad y \xrightarrow{\surd} \mu'}{x \parallel_B y \xrightarrow{\surd} \delta_{\mathbf{0}}}$$

$$\frac{x \xrightarrow{\hat{a}}\!\!\!\!\!\!/\ \ \rightarrow}{\mathsf{U}(x) \xrightarrow{\hat{a}} \delta_{\mathbf{0}}} \qquad \frac{x \xrightarrow{b} \mu \quad \mu(Y) \geq 1 \quad \{\mathsf{U}(y) \xrightarrow{\hat{a}} \mu'_y \mid y \in Y\}}{\mathsf{U}(x) \xrightarrow{\hat{a}} \delta_{\mathbf{0}}}\ b \neq a, x \notin Y$$

Tabla 4.1.: Reglas para el álgebra probabilística del Ejemplo 4.1.1 ($Y \subseteq \mathcal{V}$ es un conjunto infinito contable)

Ejemplo 4.2.1. *Las reglas para el álgebra de procesos del Ejemplo 4.1.1 están definidas en el Tabla 4.1. En éstas se considera un conjunto de etiquetas $A = \mathcal{L} \cup \{\surd\}$ donde $\surd \notin \mathcal{L}$ y $\hat{a} \in \mathcal{L}$.* □

4.3. Derivación de sistemas de transiciones

Una especificación de un sistema de transición con premisas negativas no necesariamente define un sistema de transición [16, 39, 43]. Por otro lado, en algunos casos, un PTSS puede tener más de un modelo soportado. Por ejemplo, tomemos el PTSS con una constante f, un conjunto de etiquetas $\{a, b\}$ y las siguientes dos reglas:

$$\frac{f \xrightarrow{a} \mu}{f \xrightarrow{a} \delta_f} \qquad \frac{f \xrightarrow{a}\!\!\!\!\!\!/\ \ \rightarrow}{f \xrightarrow{b} \delta_f}$$

El mismo tiene dos modelos soportados: $\{f \xrightarrow{a} \delta_f\}$ y $\{f \xrightarrow{b} \delta_f\}$.

La manera de asignar modelos a especificaciones con premisas negativas ha sido estudiado en profundidad en un contexto no probabilista [16, 39]. Estos resultados son fácilmente adaptables al contexto con probabilidades. En particular, en [22] utilizamos la técnica basada en estratificación. En esta tesis utilizamos la técnica basada en *pruebas bien soportadas* (en inglés *well supported proofs*) que es estrictamente más general que la basada en estratificación. Esta técnica utiliza las nociones *prueba* y *prueba bien soportada*. La primera es una noción de prueba que permite derivar reglas con conclusiones positivas. La segunda es una noción de prueba que permite derivar literales no-cuantitativos, i.e. literales positivos y negativos. En el caso de los literales negativos, estos serán derivables en el momento en que pueda afirmarse con total se-

guridad que no existe una prueba de un literal positivo que lo niegue. Otra diferencia entre las pruebas y las pruebas bien soportadas, es que las segundas no utilizan premisas.

Un PTSS $P = (\Sigma, A, R)$ es *significativo*, i.e. puede utilizarse para definir un sistema de transición, si para todo $t \in T(\Sigma), a \in A$ se puede derivar una prueba bien soportada de $t \overset{a}{\not\rightarrow}$ o $t \overset{a}{\rightarrow} \pi$ para algún $\pi \in \mathsf{DT}(\Sigma)$ utilizando el conjunto de reglas R. El sistema de transición basado en pruebas bien soportadas será el conjunto de literales no-cuantitativos que pueden ser demostrados.

Definición 4.3.1. *Sea $P = (\Sigma, A, R)$ un PTSS. Sea ψ un literal positivo y H un conjunto de literales (no necesariamente cerrados). Una* prueba *de una regla $\frac{H}{\psi}$ de P es un árbol bien fundado en el cual cada nodo está etiquetado por un literal tal que:*

1. *la raíz está etiquetada con ψ y*
2. *si un nodo q está etiquetado con un literal χ y K es el conjunto de etiquetas de los nodos que se encuentran directamente sobre q, entonces alguna de las siguientes condiciones se satisface*
   a) *$K = \emptyset$ y, $\chi \in H$ o χ es una premisa cuantitativa (cerrada) válida,*
   b) *$\frac{K}{\chi}$ es una instancia de substitución de una regla en R.*

Si una prueba de $\frac{H}{\psi}$ de P existe, entonces $\frac{H}{\psi}$ es demostrable*, notación $P \vdash \frac{H}{\psi}$.*

Para simplificar la notación, si ψ es un literal cuantitativo cerrado, $P \vdash \psi$ denota que ψ es válido. Dado dos literales $t \overset{a}{\rightarrow} \pi$ y $t \overset{a}{\not\rightarrow}$, diremos que uno niega al otro. Dado un literal ψ, denotaremos con $\neg\psi$ a un literal que niega ψ. Sea $\mathfrak{p}$ una prueba de la regla $\frac{H}{\psi}$, con $h(\mathfrak{p})$ se denota la altura del árbol de prueba $\mathfrak{p}$.

Definición 4.3.2. *Una* prueba bien soportada *de un literal no-cuantitativo ψ de un PTSS $P = (\Sigma, A, R)$ es un árbol bien fundado en el que lo nodos están etiquetados con literales cerrados tal que:*

1. *la raíz está etiquetada con ψ, y*
2. *si el nodo q está etiquetado con un literal no-cuantitativo χ y K es el conjunto de etiquetas de los nodos que se encuentran directamente sobre q, entonces:*
   a) *si χ es un literal positivo entonces $\frac{K}{\chi}$ es una instancia de substitución de R,*
   b) *si χ es un literal negativo entonces para toda prueba de $P \vdash \frac{N}{\neg\chi}$, con N un conjunto de literales negativos, un literal en K niega a un literal en N.*
3. *si q es un nodo etiquetado con un literal cuantitativo, entonces es válido y no existen nodos sobre q.*

Un literal ψ es ws-demostrable*, notación $P \vdash_{ws} \psi$, si existe una prueba bien soportada de ψ en P.*

Al igual que en el caso de pruebas, si ψ es una premisa cuantitativa cerrada entonces $P \vdash_{ws} \psi$ denotará que la premisa cuantitativa es válida.

El punto (2.b) de la Def 4.3.2 establece que un literal negativo $t \overset{a}{\not\to}$ es derivable sólo cuando es seguro que un literal $t \overset{a}{\longrightarrow} \pi$ para cualquier $\pi \in \mathsf{DT}(\Sigma)$ es imposible de derivar. Esta práctica se referencia a veces como *negación como falla* (*negation as a failure*) [19]

Ejemplo 4.3.1. *Sea un PTSS $P = \langle \Sigma, A, R \rangle$ con Σ una signatura con constantes a y b, una función unaria f y una función binaria g. Además, $A = \{a, b\}$ y R está compuesto por las siguiente reglas:*

$$a \overset{a}{\longrightarrow} \pi_a \qquad b \overset{b}{\longrightarrow} \delta_b \qquad \frac{x \overset{a}{\longrightarrow} \mu \quad y \overset{b}{\not\to} \quad \mu(\{y\}) > 0}{f(x) \overset{a}{\longrightarrow} \delta_b} \qquad \frac{f(x_0) \overset{a}{\longrightarrow} \mu \quad f(x_1) \overset{a}{\not\to}}{g(x_0, x_1) \overset{a}{\longrightarrow} \mu}$$

con $\pi_a = 0{,}5\delta_a + 0{,}5\delta_b$. Luego el término $g(a, b) \overset{a}{\longrightarrow} \delta_b$ es derivable mediante la siguiente prueba bien soportada

$$\dfrac{\dfrac{a \overset{a}{\longrightarrow} \pi_a \quad a \overset{b}{\not\to} \quad \pi_a(a) > 0}{f(a) \overset{a}{\longrightarrow} \delta_b} \qquad \dfrac{b \overset{b}{\longrightarrow} \delta_b}{f(b) \overset{a}{\not\to}}}{g(a, b) \overset{a}{\longrightarrow} \delta_b}$$

El literal $a \overset{b}{\not\to}$ es derivable porque no existe H y π tal que $P \vdash \frac{H}{a \overset{b}{\longrightarrow} \pi}$ con H un conjunto de premisas negativas; esto se deduce a partir de las reglas. Por otro lado, para derivar $f(b) \overset{a}{\not\to}$ sí existe un conjunto de premisas negativas H y una distribución π tal que $P \vdash \frac{H}{f(b) \overset{a}{\longrightarrow} \pi}$. En este caso, $H = \{b \overset{b}{\not\to}\}$ y la regla es derivable utilizando la tercera regla. Por esta razón, para derivar $f(b) \overset{a}{\not\to}$ se debe demostrar que existe una prueba que niegue $b \overset{b}{\not\to}$. Esto es directo usando el axioma $b \overset{b}{\longrightarrow} \delta_b$ (rama derecha de la prueba). El resto del árbol de prueba sigue una construcción estándar.

El método basado en pruebas bien soportadas es *consistente* en el sentido en que, dado un PTSS *P*, no es posible demotrar mediante pruebas bien soportadas ψ y $\neg\psi$. El Lema 4.3.1 establece este resultado. Su prueba es similar al mismo resultado en el contexto no probabilístico [39, Proposición 10].

Lema 4.3.1. *Sea P un PTSS. No existe ψ tal que $P \vdash_{ws} \psi$ y $P \vdash_{ws} \neg\psi$.*

Un PTSS *P* es *completo* si para todo término *t* y etiqueta *a*, $P \vdash_{ws} t \overset{a}{\longrightarrow} \pi$ para alguna distribución π o $P \vdash_{ws} t \overset{a}{\not\to}$. Los únicos PTSS que serán de nuestro interés son aquellos que son completos. Estos serán los utilizados para derivar sistemas de transiciones, los cuales serán modelos bien soportados del PTSS en cuestión.

Definición 4.3.3. *Un PTSS P es* significativo *(basado en pruebas bien soportadas) si es completo.*

Definición 4.3.4. *El sistema de transición basado en pruebas bien soportadas asociado a un PTSS (completo) P, notación $\rightarrow_{ws}$, es el conjunto de literales ws-demostrables en P.*

Diremos que dos PTSS son *equivalentes* si ambos pueden demostrar el mismo conjunto de literales ws-demostrables. El siguiente lema brinda condiciones suficientes para asegurar que dos PTSS son equivalentes. El mismo será usado frecuentemente a lo largo del trabajo.

Lema 4.3.2. *Sea P y P′ dos PTSS sobre la misma signatura tal que $P \vdash \frac{H}{c}$ sii $P' \vdash \frac{H}{c}$ para todo regla cerrada $\frac{H}{c}$ con H conteniendo solo premisas negativas. Entonces $P \vdash_{ws} \psi$ sii $P' \vdash_{ws} \psi$ para todo literal cerrado ψ.*

Demostración. Supongamos $P \vdash \frac{N}{\phi} \Leftrightarrow P' \vdash \frac{N}{\phi}$. Sólo demostraremos que $P \vdash_{ws} \psi \Rightarrow P' \vdash_{ws} \psi$ para todo literal positivo o negativo ψ. El otro caso es simétrico. Procedemos por inducción completa en la altura del árbol de prueba. El caso base es directo.

Supongamos que ψ es un literal negativo y que K es el conjunto de literales inmediatamente sobre ψ en el árbol de prueba. Luego $P \vdash_{ws} \psi_k$ para todo $\psi_k \in K$ y por hipótesis inductiva $P' \vdash_{ws} \psi_k$. Sean $\mathfrak{p}_q$ los arboles de prueba de cada ψ_k en P'. Por otro lado, por Def. 4.3.2, para todo conjunto N de literales negativos tal que $P \vdash \frac{N}{\psi'}$ con ψ' negando a ψ, existe un ψ_k que niega un literal en N. Dado que, por hipótesis $P \vdash \frac{N}{\psi'} \Leftrightarrow P' \vdash \frac{N}{\psi'}$, para toda regla cerrada con N un conjunto de literales negativos, el árbol de prueba que determina que $P' \vdash_{ws} \psi$ se construye etiquetando con ψ la raíz y tomando el conjunto $\{\mathfrak{p}_q \mid \psi_k \in K\}$ como sus descendientes.

Supongamos ahora que ψ es un literal positivo. Sea $\mathfrak{p}_\psi$ una prueba bien soportada de ψ y sea N_ψ los literales negativos que aparecen como etiquetas en la prueba $\mathfrak{p}_\psi$. Sea p'_ψ la prueba obtenida al eliminar de la prueba bien soportada p_ψ los nodos sobre los literales N_ψ. Entonces, $P \vdash \frac{N_\psi}{\psi}$, porque p'_ψ es una prueba de la regla. Por hipótesis $P' \vdash \frac{N_\psi}{\psi}$. Repitiendo el razonamiento del caso anterior, para todo $\psi' \in N_\psi$, si $P \vdash_{ws} \psi'$ entonces $P' \vdash_{ws} \psi'$. Utilizando las pruebas bien soportadas de $P' \vdash_{ws} \psi'$ junto a la prueba p'_ψ se puede construir en prueba bien soportada de ψ en P', i.e. $P' \vdash_{ws} \psi$. □

4.4. Observaciones finales

En este capítulo introducimos los PTSS, los cuales permiten definir la especificación de un PTS, i.e. un sistema de transiciones probabilista al estilo de Segala. Es más, introducimos un método para asociar a un PTSS completo un modelo soportado. Este método se basa en pruebas bien soportadas. Finalmente el Lema 4.3.2 brinda condiciones suficientes para garantizar que dos PTSS son equivalentes, i.e. a ambas especificaciones se le asigna el mismo modelo soportado.

En [16] y [39] se estudia en profundidad como asignar un sistema de transición a una especificación con premisas negativas. En [22] extendemos el método basado en estratificación al contexto probabilístico. En [59] y aquí presentamos la extensión del método basado en pruebas bien soportadas.

Estas extensiones pueden realizarse sin mayores dificultades debido a que el único problema a resolver es definir cuando una premisa negativa es válida. En general, esto se resuelve definiendo un "orden" para derivar los literales. Este orden garantiza que para toda regla r y substitución cerrada ρ tal que la regla $\rho(r)$ posee una premisa negativa $\rho(t \overset{a}{\not\rightarrow})$ entonces para toda distribución

π, si $\rho(t \xrightarrow{a} \pi)$ es derivable, esta transición debe derivarse antes que $\rho(\text{conc}(r))$. Por lo tanto, al momento de derivar $\rho(\text{conc}(r))$ utilizando la regla $\rho(r)$ se verifica sin inconveniente la validez de $\rho(t \not\xrightarrow{a})$. En el caso del método basado en estratificación este orden es explicito y está dado por una función denominada *estratificación*. En el caso del método basado en pruebas bien soportadas es implícito y está codificado en las mismas pruebas bien soportadas: un literal negativo es demostrable sólo cuando se puede garantizar que no existe una prueba de un literal positivo que lo niegue (Def. 4.3.2, item 2.b). Los primeros literales negativos que se pueden derivar son aquellos para los cuales las reglas no permiten derivar un literal que los niegue (en el Ejemplo 4.3.1 $a \not\xrightarrow{b}$ porque no hay regla con una conclusión de la forma $a \xrightarrow{a} \theta$). Por otro lado, los primeros literales positivos, son aquellos que se puede derivar utilizando, o no, los literales negativos "recién derivados". Estos nuevos literales positivos permiten falsificar nuevas pruebas y por lo tanto derivar nuevos literales negativos (otra vez Def. 4.3.2, item 2.b). A su vez, estos nuevos literales negativos permiten derivar nuevos literales positivos. Así se continua sucesivamente.

Por último, las premisas cuantitativas no afectan el orden que se utilice, pues son sólo condiciones extras para derivar una transición.

Capítulo 5

El formato $ntμfθ/ntμxθ$

Debido a su popularidad, el enfoque estructural para dar semántica operacional a los lenguajes [66] se convirtió en un tema de estudio en sí mismo. Este estudio se centro principalmente en la obtención de meta-resultados y en la definición de *formatos de reglas*. En [63] se realiza una recopilación de los mismos.

Un formato de regla impone restricciones sintácticas a las reglas que pueden utilizarse y de esta forma se logra que los sistemas especificados satisfagan propiedades que serán de nuestro interés. Esto se debe a que no todo conjunto de regla garantiza buenas propiedades sobre la semántica de las funciones que se definen. Por ejemplo, en la Figura 5.1 recordamos el sistema de transición especificado en la introducción del Capítulo 4. Notemos que a y a'son isomorfos, luego para toda noción de observabilidad N se satisface que $O_N(a) = O_N(a')$. Supongamos ahora que $f(a)$ es el resultado de utilizar un sistema $f(\cdot)$ junto al componente a. Dado que a y a' presentan el mismo comportamiento sería deseable que $f(a)$ y $f(a')$ también presenten el mismo comportamiento, es decir, se espera que la equivalencia inducida por la noción de observabilidad N sea una congruencia para f. De la Figura 5.1 se desprende que esto no es así, i.e. $O_N(f(a)) \neq O_N(f(a'))$.

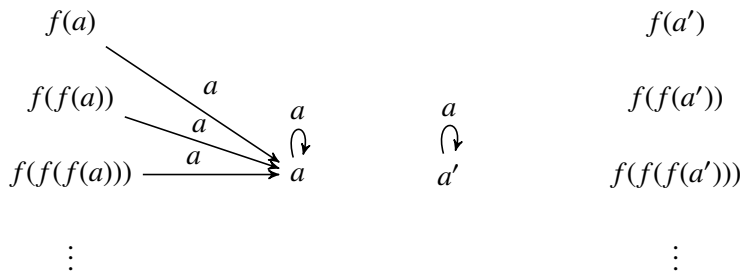


Figura 5.1.: El ejemplo de la introducción del Capítulo 4

En el presente capítulo se introducirá el formato *ntμfθ/ntμxθ*, el cual garantiza que la bisimulación, en un contexto probabilista, es una congruencia para todo operador que se define utilizando el formato. Formalmente, si $\sim$ denota la equivalencia inducida por la bisimulación, el formato garantiza que si $u_i \sim v_i$ con $u_i, v_i \in T(\Sigma)$ para $i \in \{1, \ldots, n\}$, y f es una función n-aria, tal que su semántica operacional fue definida usando el formato *ntμfθ/ntμxθ* entonces:

$$f(u_1, \ldots, u_n) \sim f(v_1, \ldots, v_n)$$

Definir un formato de regla consiste en buscar la menor cantidad de restricciones que garanticen la propiedad deseada. Naturalmente las restricciones dependerán de la propiedad a asegurar y a mayor restricciones menor poder expresivo. Luego un formato no sólo debe garantizar una propiedad sino también permitir definir semánticas lo suficientemente expresivas. En particular, el formato *ntμfθ/ntμxθ* es bastante expresivo dado que soporta reglas con las siguientes características:

- premisas positivas de la forma $t \xrightarrow{a} \mu$ con $t \in \mathbb{T}(\Sigma)$, $a \in A$ y $\mu \in \mathcal{M}$;
- premisas negativas de la forma $t \not\xrightarrow{a}$ con $t \in \mathbb{T}(\Sigma)$ y $a \in A$;
- premisas cuantitativas de la forma $\theta(Y) \trianglerighteq q$ con $\theta \in \mathbb{DT}(\Sigma)$ y $\trianglerighteq \in \{>, \geq\}$;
- conclusiones de la forma $x \xrightarrow{a} \theta$ o $f(x_1, \ldots, x_{\mathrm{r}f}) \xrightarrow{a} \theta$ con $f \in \Sigma$, $x, x_1, \ldots, x_{\mathrm{r}f} \in \mathit{Var}$ y $\theta \in \mathbb{DT}(\Sigma)$.

Existen otros formatos que garantizan que la bisimulación es una congruencia, tanto para un contexto con, o sin, probabilidades. En un contexto no probabilístico se encuentran: *De Simone format* [27], *GSOS* [15], *tyft/tyxt* [44], *ntyft/ntyxt* [16,42,43], *path format* [7] y *panth format* [76]. Por otro lado, en un contexto probabilístico se encuentran los formatos: RTSS [53], PGSOS [10], y Segala-GSOS [10].

El formato *ntμfθ/ntμxθ* podría considerarse una extensión del formato *ntyft/ntyxt*. Este último, por razones obvias, no soporta premisas cuantitativas. Sí soporta premisas negativas y positivas como las que soporta el formato *ntμfθ/ntμxθ* pero en un contexto no probabilista. Finalmente, el formato *ntμfθ/ntμxθ*, al igual que el formato *ntyft/ntyxt*, permite realizar *double testing* y *lookahead*.

Double testing: Una regla puede realizar "*double testing*" si permite premisas no cuantitativas con la misma fuente. Por ejemplo una regla puede tener como premisas los literales $t \xrightarrow{a} \mu$ y $t \xrightarrow{b} \mu'$.

Lookahead: Una regla puede realizar "*lookahead*" si el estado destino de una premisa positiva puede aparecer en la fuente de otra premisa positiva. Un ejemplo para el formato *ntyft/ntyxt* sería una regla con premisas $x \xrightarrow{a} y$ e $y \xrightarrow{b} z$. En el caso del formato *ntμfθ/ntμxθ* un ejemplo sería una regla con premisas $x \xrightarrow{a} \mu, \mu(Y) \trianglerighteq q$ y $y \xrightarrow{b} \mu'$ con $y \in Y$. Notemos que siempre se debe usar una premisa cuantitativa para hablar de los estados alcanzables desde otro estado.

5.1. Bisimulación y congruencia

La bisimulación sobre sistemas de transiciones probabilistas fue introducida por Larsen y Skou en [54, 55]. En esta tesis utilizamos una definición más moderna, aunque equivalente. Ésta se presenta en [23] y utiliza conjuntos cerrados en vez de clases de equivalencia.

Definición 5.1.1. *Dada una relación $R \subseteq T(\Sigma) \times T(\Sigma)$, un conjunto $Q \subseteq T(\Sigma)$ es R-cerrado si para todo $t \in Q$ y $t' \in T(\Sigma)$, $t\ R\ t'$ implica $t' \in Q$ (i.e. $R(Q) \subseteq Q$). Si el conjunto Q es R-cerrado lo denotaremos con R-cerrado(Q).*

Es fácil verificar que si dos relaciones $\mathsf{R}, \mathsf{R}' \subseteq T(\Sigma) \times T(\Sigma)$ son tales que $\mathsf{R}' \subseteq \mathsf{R}$, entonces para todo conjunto $Q \subseteq T(\Sigma)$, R-cerrado(Q) implica R'-cerrado(Q).

Definición 5.1.2. *Una relación $R \subseteq T(\Sigma) \times T(\Sigma)$ es una bisimulación si R es simétrica y para todo $t, t' \in T(\Sigma)$, $\pi \in \Delta(T(\Sigma))$, $a \in A$,*

$$t\ R\ t' \text{ y } t \xrightarrow{a} \pi \text{ implica que existe } \pi' \in \Delta(T(\Sigma)) \text{ tal que } t' \xrightarrow{a} \pi' \text{ y } \pi\ R\ \pi',$$

donde $\pi\ R\ \pi'$ si y sólo si $\forall Q \subseteq T(\Sigma) : R\text{-cerrado}(Q) \Rightarrow \pi(Q) = \pi'(Q)$. Definimos a la relación $\sim$ como la menor relación que incluye todas las bisimulaciones. Es ya bien sabido que $\sim$ es también una bisimulación y una relación de equivalencia (Ver por ejemplo [23]).

La definición se basa en conjuntos cerrados y no en clases de equivalencia porque para basarse en clases de equivalencia se debe requerir que R sea una relación de equivalencia. Luego esta definición presenta menos restricciones para la relación R.

Notemos que la definición de bisimulación no establece ninguna condición sobre las acciones que no puede realizar un estado. Esto se debe a que se está trabajando con PTSS completos y que la bisimulación es una relación simétrica, por lo tanto las condiciones impuestas son suficientes para garantizas que si $t\ R\ t'$ y $t \xrightarrow{a}$ entonces $t' \xrightarrow{a}$.

Ejemplo 5.1.1. *Consideremos los sistemas de transiciones probabilistas de la Figura 5.2. Se puede ver que la relación $\{(p_i, q_i) \mid i = 1, \ldots 5\} \cup \{(p_2, q'_2), (q'_2, p_2)\}$ es una bisimulación, por consiguiente $p_1 \sim q_1$.*

Una relación de equivalencia R es una congruencia en una signatura Σ si un término puede ser reemplazado por R-equivalente en un contexto, de manera que el resultado del reemplazo sea R-equivalente al original.

Definición 5.1.3. *Sea $\Sigma = (F, \mathsf{r})$ una signatura y sea $f \in F$. Una relación de equivalencia $R \subseteq T(\Sigma) \times T(\Sigma)$ es una* congruencia *con respecto f si dados $u_1, \ldots, u_{\mathsf{r}(f)}, v_1, \ldots, v_{\mathsf{r}(f)} \in T(\Sigma)$*

$$\bigwedge_{i=1}^{i=\mathsf{r}(f)} u_i\, R\, v_i \Rightarrow f(u_1, \ldots, u_{\mathsf{r}(f)})\, R\, f(v_1, \ldots, v_{\mathsf{r}(f)})$$

Si bien esta definición se aplica sólo en las operaciones de la signatura Σ, su generalización a contextos arbitrarios de Σ es evidente (y equivalente).

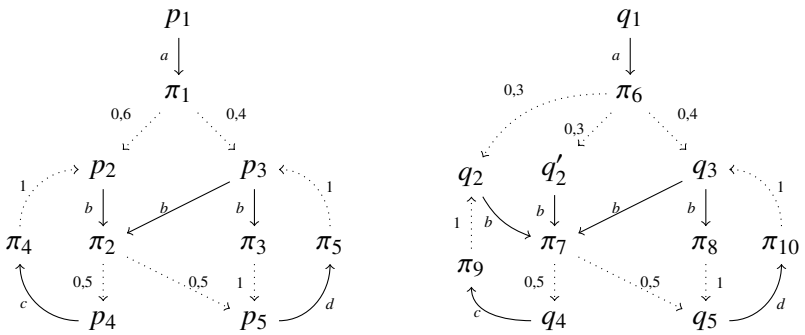


Figura 5.2.: Se puede ver que la relación $\{(p_i, q_i) \mid i = 1, \ldots 5\} \cup \{(p_2, q'_2), (q'_2, p_2)\}$ es una bisimulación, por consiguiente $p_1 \sim q_1$

5.2. El formato $nt\mu f\theta/nt\mu x\theta$

Antes de introducir el formato $nt\mu f\theta/nt\mu x\theta$ de forma completa, presentaremos una propuesta más simple. La misma puede verse como una extensión del formato *ntyft/ntyxt* para que soporte probabilidades y considere premisas cuantitativas con una forma muy restrictiva. Al mismo tiempo, también podría considerarse como una generalización del formato Segala-GSOS [10] con términos en la fuente de las premisas no cuantitativas, así como con lookahead. Denominamos a este formato *$nt\mu f\theta$ simple*.

Definición 5.2.1 ($nt\mu f\theta$ simple)**.** *Sea $P = (\Sigma, A, R)$ un PTSS. Una regla $r \in R$ está en $nt\mu f\theta$* formato *si tiene la siguiente forma*

$$\frac{\{t_m \xrightarrow{a_m} \mu_m : m \in M\} \cup \{t_n \xrightarrow{b_n} \not\rightarrow : n \in N\} \cup \{\mu_l(z_l) > 0 : l \in L\}}{f(x_1, \ldots, x_{\mathsf{r}(f)}) \xrightarrow{a} \theta} \tag{F}$$

donde M, N, y L son conjuntos de índices, μ_m, z_l, x_k ($1 \leq k \leq \mathsf{r}(f)$) son todas variables diferentes, $f \in F$, $t_m, t_n \in \mathbb{T}(\Sigma)$, y $\theta \in \mathbb{DT}(\Sigma)$. P está en formato $nt\mu f\theta$ simple *si todas sus reglas están en formato $nt\mu f\theta$ simple.*

Se puede demostrar que la bisimulación es una congruencia para cualquier operador definido utilizando el formato $nt\mu f\theta$ simple. De hecho esto se puede concluir como un corolario del Teorema 5.3.2. Por consiguiente, no nos detendremos con resultados técnicos sobre este formato y sólo lo utilizaremos para explicar los ingredientes básicos que luego se harán presentes nuevamente en el formato $nt\mu f\theta/nt\mu x\theta$.

A continuación presentamos ejemplos los cuales justifican las restricciones impuestas sobre este formato. Consideremos una signatura con un operador unario f y tres constantes b, c y d, junto con una etiqueta a. Consideremos además los siguientes axiomas:

$$c \xrightarrow{a} \delta_c \qquad d \xrightarrow{a} \pi_d \text{ con } \pi_d = 0{,}5 \cdot \delta_c + 0{,}5 \cdot \delta_d$$

No existe regla asociada a la constante b. Notemos que $c \sim d$. Usando esta especificación como base, definiremos distintas reglas para f las cuales justificaran las restricciones impuestas.

La necesidad de que la fuente de la conclusión satisfaga una forma particular ha sido justificada por varios ejemplos en [43, 44] para el formato *tyft/tyxt*. Adaptamos un ejemplo de [43] para motivar la necesidad. Consideremos el axioma

$$f(b) \xrightarrow{a} \delta_{f(b)}$$

Los términos b y $f(f(b))$ no realizan transiciones debido a que no aparecen en la fuente de los axiomas definidos. Como ambos términos no ejecutan transiciones, $f(f(b)) \sim b$. Sin embargo $f(f(f(b)))$ y $f(b)$ no son bisimilares dado que $f(b)$ puede ejecutar una transición con a y $f(f(f(b)))$ no.

El siguiente ejemplo muestra que el destino de una premisa positiva no puede ser una distribución particular. Consideremos la regla

$$\frac{x \xrightarrow{a} \delta_c}{f(x) \xrightarrow{a} \delta_c}$$

Como consecuencia, a pesar de que $c \sim d$, $f(c)$ y $f(d)$ no son bisimilares dado que $d \xrightarrow{a} \delta_c$ no es una transición válida en el único modelo soportado. Un efecto similar tiene la regla

$$\frac{x \xrightarrow{a} \mu \;\; \mu(d) > 0}{f(x) \xrightarrow{a} \delta_c}$$

la cual muestra que los literales cuantitativos no pueden inquirir sobre términos arbitrarios: notemos que $f(c)$ y $f(d)$ no son bisimilares dado que $c \xrightarrow{a} \delta_c$ y $\delta_c(d) = 0$.

Los literales cuantitativos no pueden realizar comparaciones utilizando cotas superiores (o igualdad). Consideremos la regla

$$\frac{x \xrightarrow{a} \mu \;\; y \xrightarrow{a} \mu' \;\; \mu(y) \leq 0{,}5}{f(x) \xrightarrow{a} \delta_c}$$

Los términos $f(c)$ y $f(d)$ no son bisimilares porque $f(d) \xrightarrow{a} \delta_c$ pero $f(c) \not\xrightarrow{a}$ dado que no existe un término t que pueda substituir *genuinamente* a y (i.e., tal que $\delta_c(t) > 0$) y $\delta_c(t) \leq 0{,}5$.

Notemos que el requerimiento de que la substitución sea genuina permite reemplazar "> 0" por "≥ 0". Por otro lado, permitir que los literales cuantitativos comparen valores distintos de 0 es problemático. Consideremos la regla

$$\frac{x \xrightarrow{a} \mu \;\; y \xrightarrow{a} \mu' \;\; \mu(y) \geq 1}{f(x) \xrightarrow{a} \delta_c}$$

Una vez más $f(c)$ y $f(d)$ no son bisimilares dado que $d \xrightarrow{a} \pi_d$ y no existe un término t tal que $\pi_d(t) \geq 1$.

Este ejemplo sugiere que las premisas cuantitativas deberían ser de la forma $\mu(Y) > p$ o $\mu(Y) \geq p$ donde Y es un conjunto de variables. De esta forma, la regla previa podría ser reescrita como

$$\frac{x \xrightarrow{a} \mu \;\; y \xrightarrow{a} \mu' \;\; \mu(\{y,z\}) \geq 1}{f(x) \xrightarrow{a} \delta_c} \tag{I}$$

De todas formas, el mismo problema se repite si introducimos a la especificación una nueva constante e con un axioma $e \xrightarrow{a} (0{,}4 \cdot \delta_c + 0{,}3 \cdot \delta_d + 0{,}3 \cdot \delta_e)$. Es más, generalizando este ejemplo se puede concluir que el conjunto Y necesita ser *infinito*. Puntualmente, supongamos un conjunto infinito de nuevas constantes $\{e_n\}_{n \in \mathbb{N}_0}$ tales que $e_n \xrightarrow{a} (\sum_{i=\mathbb{N}_0}^{n} \frac{1}{2^{i+1}} \cdot \delta_{e_i}) + \frac{1}{2^{n+1}} \delta_{e_{n+1}}$; luego para todo $n \in \mathbb{N}_0$, $e_n \sim c$. Si Y no es infinito entonces $f(e_{n'}) \xrightarrow{a} \delta_c$ no será una transición válida para $n' \geq |Y|$ debido a que la premisa cuantitativa será imposible de satisfacer y por lo tanto $f(n') \not\sim f(c)$. Además, es necesario que todo término que substituya una variable en Y tenga un comportamiento simétrico. Con simétrico nos referimos a que las condiciones impuestas a una variable en Y deben ser impuesta a todas las otras variables del conjunto. Por ejemplo, el requisito de simetría no está presente en la regla (I). Al término cerrado con el que se substituye z no se le requiere que realice una acción a, i.e, la premisa $z \xrightarrow{a} \mu''$ no pertenece a la regla. Sin la condición de simetría la bisimulación no se conserva. Se presenta un ejemplo de esto luego de introducir el formato.

Luego de estas consideraciones, extendemos el formato $nt\mu f\theta$ simple con premisas cuantitativas de la forma $\theta(Y) > p$ o $\theta(Y) \geq p$, con Y un conjunto infinito de variables de término. Denominaremos este formato $nt\mu f\theta/nt\mu x\theta$ siguiendo la nomenclatura de [43, 44].

Para asegurar el comportamiento simétrico introducimos la definición de Diag. Denotaremos el l-ésimo elemento de la tupla $\vec{y}$ con $\vec{y}(l)$. Para un conjunto de tuplas $T = \{\vec{y_i} \mid i \in I\}$ denotaremos la l-ésima proyección con $\pi_l(T) = \{\vec{y_i}(l) \mid i \in I\}$.

Definición 5.2.2. *Sea $\{Y_l\}_{l \in L}$ una familia de conjuntos de variables de término con la misma cardinalidad. Fijemos el conjunto $\mathsf{Diag}\{Y_l\}_{l \in L} \subseteq \prod_{l \in L} Y_l$ de forma tal que:*

(i) para todo $l \in L$, $\pi_l(\mathsf{Diag}\{Y_l\}_{l \in L}) = Y_l$;

(ii) para todo $\vec{y}, \vec{y'} \in \mathsf{Diag}\{Y_l\}_{l \in L}$, $(\exists l \in L : \vec{y}(l) = \vec{y'}(l)) \Rightarrow \vec{y} = \vec{y'}$.

La propiedad (ii) asegura que diferentes vectores $\vec{y}, \vec{y'} \in \mathsf{Diag}\{Y_l\}_{l \in L}$ difieren en todas las posiciones y por la propiedad (i) toda variable de toda Y_l es usada en un vector $\vec{y} \in \mathsf{Diag}\{Y_l\}_{l \in L}$. El término Diag es debido a la palabra"diagonal", siguiendo la intuición que cada vector $\vec{y}$ representa una coordenada en el espacio $\prod_{l \in L} Y_l$, entonces $\mathsf{Diag}\{Y_l\}_{l \in L}$ puede ser visto como una linea en la diagonal principal del espacio. Por lo tanto, si $Y_l = \{y_l^0, y_l^1, y_l^2, \ldots\}$, una posible definición para $\mathsf{Diag}\{Y_l\}_{l \in L}$ es

$$\mathsf{Diag}\{Y_l\}_{l \in L} = \{(y_0^0, y_1^0, \ldots, y_L^0), (y_0^1, y_1^1, \ldots, y_L^1), (y_0^2, y_1^2, \ldots, y_L^2), \ldots\}$$

Además utilizaremos la siguiente notación: si $t \in \mathbb{T}(\Sigma)$ y $Var(t) \subseteq \{w_1, \ldots, w_H\}$, $t(w'_1, \ldots, w'_H)$ es el mismo término t en donde todas las ocurrencias de la variable w_h (si es que aparece en t) es reemplazada por la variable w'_h, para $1 \leq h \leq H$.

Definición 5.2.3. *Sea $P = (\Sigma, A, R)$ un PTSS. Una regla $r \in R$ está en* formato *$nt\mu f\theta$ si tiene la siguiente forma*

$$\frac{\bigcup_{m\in M}\{t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m : \vec{z} \in \mathcal{Z}\} \cup \bigcup_{n\in N}\{t_n(\vec{z}) \xrightarrow{b_n} \not\rightarrow : \vec{z} \in \mathcal{Z}\} \cup \{\theta_l(Y_l) \trianglerighteq_{l,k} p_{l,k} : l \in L, k \in K_l\}}{f(x_1, \ldots, x_{r(f)}) \xrightarrow{a} \theta}$$

con $\trianglerighteq_{l,k} \in \{>, \geq\}$ para todo $l \in L$ y $k \in K_l$, y las siguientes condiciones se satisfacen:

1. *Cada conjunto Y_l es contable infinito, para todo $l \in L$, y la cardinalidad de L tiene que ser estrictamente menor a la cardinalidad de los conjuntos Y_l's.*
2. *$\mathcal{Z} = \mathsf{Diag}\{Y_l\}_{l\in L} \times \prod_{w\in W}\{w\}$, con $W \subseteq \mathcal{V} \setminus \bigcup_{l\in L} Y_l$.*
3. *Todas las variables $\mu^{\vec{z}}_m$, con $m \in M$ y $\vec{z} \in \mathcal{Z}$, son diferentes.*
4. *Para todo $\vec{z}, \vec{z'} \in \mathcal{Z}$, $m \in M$, si $\mu^{\vec{z}}_m, \mu^{\vec{z'}}_m \in Var(\theta) \cup (\cup_{l\in L} Var(\theta_l))$ entonces $\vec{z} = \vec{z'}$.*
5. *Para todo $l \in L$, $Y_l \cap \{x_1, \ldots, x_{r(f)}\} = \emptyset$, y $Y_l \cap Y_{l'} = \emptyset$ para todo $l' \in L$, $l \neq l'$.*
6. *Las variables $x_1, \ldots, x_{r(f)}$ son todas diferentes.*
7. *Para todo $l \in L$, $Var(\theta_l) \cap (\{x_1, \ldots, x_{r(f)}\} \cup \bigcup_{l'\in L} Y_{l'}) = \emptyset$.*

Una regla $r \in R$ está en formato *$nt\mu x\theta$ si tiene la forma anterior pero la conclusión satisface la siguiente forma $x \xrightarrow{a} \theta$ y, además, se satisfacen las mismas condiciones excepto que en los items (5) y (7) en vez de escribir $\{x_1, \ldots, x_{r(f)}\}$, escribimos $\{x\}$. P está en* formato *$nt\mu f\theta$ si todas sus reglas están en formato $nt\mu f\theta$ Similarmente, P está en* formato *$nt\mu x\theta$ si todas sus reglas lo están. P está en* formato *$nt\mu f\theta/nt\mu x\theta$ si todas sus reglas están en alguno de los formatos $nt\mu f\theta$ o $nt\mu x\theta$.*

Las variables $x_1, \ldots, x_{r(f)}$ que aparecen en la fuente de la conclusión son variables ligadas. Las variables en $\bigcup_{l\in L} Y_l$ y las que aparecen en las distribuciones de Dirac instanciables son también variables ligadas cuando aparecen en una premisa cuantitativa. Por lo tanto éstas tienen que ser todas diferentes. Esto se establece en las condiciones 3, 5, y 7. Las variables de distribución en $\{\mu^{\vec{z}}_m \mid m \in M \wedge \vec{z} \in \mathcal{Z}\}$ están ligadas cuando aparecen en el destino de una premisa positiva. Entonces éstas también necesitan ser diferentes, esto se establece en la condición 6. Los conjuntos Y_l, como ya fue explicado, deben ser infinitos, esto se estable en la condición 1. La condición 4, sumado al conjunto de premisas $\{t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m : \vec{z} \in \mathcal{Z}\}$ aseguran el comportamiento simétrico de los términos $t_m(\vec{z})$ para todo posible instanciación de las variables $\vec{z}$. El siguiente ejemplo muestra el porqué de este requerimiento: supongamos que el comportamiento simétrico en las variables Y_l no es requerido, entonces se podría diferenciar distribuciones que son equivalentes. Por ejemplo, consideremos una signatura con constantes c, d, y $\{n, n' \mid n \in \mathbb{N}_0\}$, operador unario f y reglas

$$n \xrightarrow{n} \delta_n \qquad n' \xrightarrow{n} \delta_n \qquad c \xrightarrow{a} \pi \qquad d \xrightarrow{a} \pi'$$

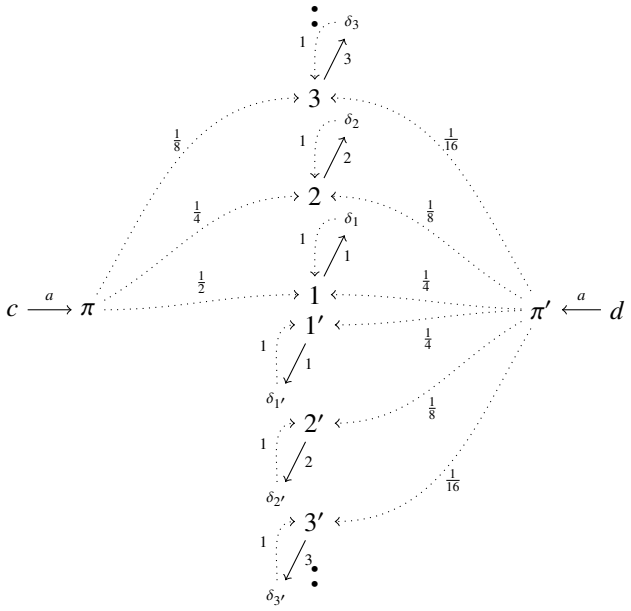


Figura 5.3.: Las variables en las premisas cuantitativas deben tener un comportamiento simétrico.

con $\pi = \sum_{i\in\mathbb{N}_0} \frac{1}{2^{i+1}} \cdot \delta_n$ y $\pi' = \sum_{i\in\mathbb{N}_0} (\frac{1}{2^{i+2}} \cdot \delta_n + \frac{1}{2^{i+2}} \cdot \delta_{n'})$ (la Figura 5.3 es una representación gráfica de los axiomas), y

$$\frac{x \xrightarrow{a} \mu \qquad \{y_k \xrightarrow{k} \mu_k \mid k \in \mathbb{N}_0\} \qquad \mu(\{y_k\}_{k\in\mathbb{N}_0}) \geq 1}{f(x) \xrightarrow{b} \mu}$$

Notar que $c \sim d$. A pesar de esto, $f(c) \xrightarrow{b} \delta_c$ pero $f(d) \not\xrightarrow{b}$ dado que $d \xrightarrow{a} \pi'$ y no existe forma de asignar los términos n y n' a dos variables diferentes y_{k_1} e y_{k_2} (para todo $n \in \mathbb{N}_0$). Por lo tanto $\pi'(\rho(\{y_k\}_{k\in\mathbb{N}_0})) = 0{,}5$ para cualquier sustitución ρ que satisfaga las premisas positivas.

Finalmente la condición 2 es sólo notación.

5.3. El teorema de congruencia

Para simplificar la demostración del teorema de congruencia primero se aplica una reducción del formato. De esta forma se demuestra que para todo PTSS *bien fundado* (Def. 5.3.1) en formato *ntμfθ/ntμxθ* existe un PTSS *puro* (Def. 5.3.3) en formato *ntμfθ* (Corolarios 5.3.1 y 5.3.2). Luego se presenta la relación R_P (Def. 5.3.4), la cual por construcción es composicional y al mismo tiempo contiene a $\sim$, i.e. $\sim \subseteq \mathsf{R}_P$. La prueba del teorema de congruencia consiste en demostrar que R_P es una bisimulación y por lo tanto $\sim \supseteq \mathsf{R}_P$. Por consiguiente $\sim = \mathsf{R}_P$, por lo cual $\sim$ es composicional.

En el Capítulo 6 se demostrará que todo PTSS en formato $nt\mu f\theta/nt\mu x\theta$ puede reducirse a uno equivalente que es bien fundado, por lo cual este requisito no es fundamental para la validez de resultado.

Premisas bien fundadas y variables libres

En primer lugar introducimos el concepto de *premisas bien fundadas*. Un conjunto de premisas es bien fundado si puede definirse un orden bien fundado sobre la dependencia de las variables que aparecen en éstas. Este orden es importante pues nos permitirá definir substituciones de forma inductiva y realizar pruebas por inducción.

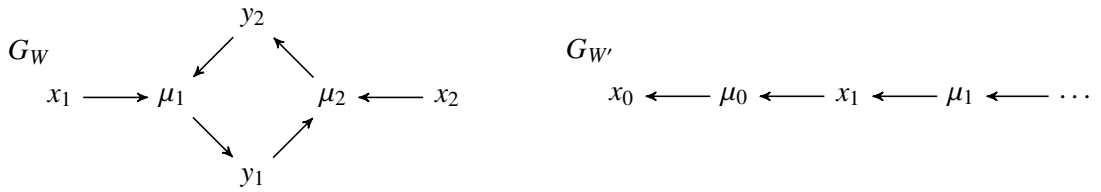
Definición 5.3.1. *Sea W un conjunto de premisas positivas y cuantitativas. El* grafo de dependencia de W *está definido como $G_W = (V, E)$ con*

$$V = \cup_{\psi \in W} Var(\psi)$$
$$E = \{\langle x, \mu\rangle \mid t \xrightarrow{a} \mu, x \in Var(t)\} \cup \{\langle \zeta, y\rangle \mid (\theta(Y) \trianglerighteq p) \in W, \zeta \in Var(\theta), y \in Y\}$$

El conjunto W es bien fundado *si cualquier cadena inversa (backward chain) de aristas en G_W es finita. Para todo $x \in V$, sea $n_{\mathrm{VDG}}(x) = \sup(\{n_{\mathrm{VDG}}(y) + 1 \mid (y, x) \in E\})$, donde $\sup(\emptyset) = 0$. Una regla es* bien fundada *si su conjunto de premisas positivas y cuantitativas es bien fundado. Un PTSS es* bien fundado *si todas sus reglas son bien fundadas.*

La función $n_{\mathrm{VDG}}(x)$ da la cantidad de nodos que existen en la cadena inversa de máxima longitud que inicia en x (en el grafo de dependencia). Luego, si el conjunto utilizado para crear el grafo de dependencia es bien fundado el valor $n_{\mathrm{VDG}}(x)$ está siempre bien definido. Esto permite realizar inducción sobre $n = n_{\mathrm{VDG}}(x)$.

Ejemplo 5.3.1. *Sean $W = \{f_1(x_1, y_2) \xrightarrow{a} \mu_1, \mu_1(\{y_1\}) > 0, f_2(x_2, y_1) \xrightarrow{a} \mu_2, \mu_2(\{y_2\}) > 0\}$ y $W' = \{x_{n+1} \xrightarrow{a} \mu_n, \mu_n(\{x_n\}) > 0 \mid n \in \mathbb{N}\}$. Sus grafos de dependencia G_W y $G_{W'}$ pueden graficarse de la siguiente manera.*



El conjunto W no es bien fundado, pues existe un ciclo en G_W; W' tampoco es bien fundado pues para toda variable x_n o μ_n existe una cadena inversa infinita en $G_{W'}$ que parte de dicha variable.

Sea r una regla bien fundada y supongamos que se quiere utilizar la misma para derivar una transición de un término t. Sea $W = \mathrm{pprem}(r) \cup \mathrm{qprem}(r)$ y G_W su respectivo grafo de dependencia. G_W, mediante la función n_{VDG}, establece el orden en el que se deben instanciar las variables de las premisas para verificar la validez de las mismas y al mismo tiempo, si todas las premisas se satisfacen, derivar una transición ψ tal que $src(\psi) = t$.

$$r = \frac{x_1 \xrightarrow{a} \mu \qquad (p_0\nu + p_1\mu + p_2\delta_{z_0})(Y) > 0 \qquad \{g(z_1, x_2, y) \xrightarrow{a} \mu^y \mid y \in Y\}}{f(x_1, x_2, x_3) \xrightarrow{a} \mu + \nu + \delta_{z_2}}$$

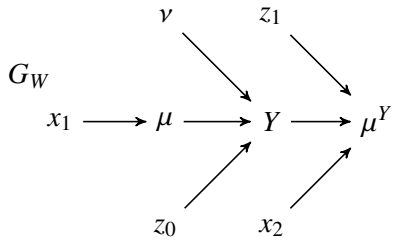


Figura 5.4.: Una regla r y una representación compacta del grafo de dependencia del conjunto $W = \text{prem}(r)$

El la Figura 5.4 se presenta una regla r en formato $nt\mu f\theta$ y una representación compacta del grafo de dependencia de $W = \text{prem}(r)$. En este último, $Y \to \mu^Y$ denota el conjunto de aristas $\{y \to \mu^y \mid y \in Y\}$, mientras que $\nu \to Y$ denota el conjunto de aristas $\{\nu \to y \mid y \in Y\}$. La misma notación vale para $\mu \to Y$, $z_0 \to Y$, $z_1 \to \mu^Y$ y $x_2 \to \mu^Y$ donde $\mu^Y = \{\mu^y \mid y \in Y\}$. El conjunto W está bien fundado.

Supongamos que se quiere utilizar la regla r para derivar el literal $\psi = f(t_1, t_2, t_3) \xrightarrow{a} \pi$ para algún $\pi \in \mathsf{DT}(\Sigma)$. Por lo tanto se necesita una substitución ρ para instanciar la regla r. Como se quiere derivar ψ, esto implica que ρ es tal que $\rho(x_i) = t_i$ para $i = 1, 2, 3$. Dado que $\rho(x_1)$ ya está definido y $x \xrightarrow{a} \mu$ es una premisa positiva, esto restringe los posibles valores de $\rho(\mu)$, pues $\rho(x_1 \xrightarrow{a} \mu)$ debe ser una premisa válida. Supongamos que se puede definir $\rho(\mu)$ tal que $\rho(x_1 \xrightarrow{a} \mu)$ es válida. Entonces tiene sentido continuar con las otras premisas. Para evaluar la premisa cuantitativa $(p_0\nu + p_1\mu + p_2\delta_{z_0})(Y) > 0$, dado que la substitución debe ser genuina y $\rho(\mu)$ ya está definido, se debe definir primero $\rho(\nu)$ y $\rho(z_0)$ y luego $\rho(Y)$. Notar que este orden está codificado por las transiciones del grafo de dependencia y por lo tanto por la función n_{VDG}. De esta forma, mientras se puedan instanciar las variables obteniendo premisas válidas, se continúa hasta definir la substitución para todas las variables.

Entonces, dada una regla existen dos tipos de variables: las que dependen de otras en el grafo de dependencia y las que no. En el ejemplo, las variables $\{\mu\} \cup Y \cup \mu^Y$ se encuentran en una sucesión de dependencias mientras que $\{x_1, x_2, x_3, z_0, z_1, z_2, \nu\}$ no dependen de ninguna otra variable (se incluyeron las variables x_3 y z_2 para considerar todas las variables de la regla). Notemos que la instanciación de las variables que se encuentran en el primer conjunto dependen de la instanciación que se realiza en las variables que aparecen en la fuente de la conclusión. A su vez, estas últimas se definen a partir del término para el cual se está derivando la transición. En el ejemplo, los posibles valores de ρ para $\mu \cup Y \cup \mu^Y$ dependen de los posibles valores de ρ para $\{x_1, x_2, x_3\}$. Estos valores, a su vez, dependen del término cerrado $f(t_1, t_2, t_3)$. Por otro lado, los valores de ρ para $\{z_0, z_1, z_2, \nu\}$ son independientes de $f(t_1, t_2, t_3)$. Las variables para las cuales su

instanciación no dependen de la instanciación de otras variables ni del término para el cual se está derivando la transición, se denominan *variables libres*

Definición 5.3.2. *Una variable x ocurre* libre *en una regla r si ocurre en r pero no en la fuente de la conclusión ni tampoco en W_j con $\theta_j(W_j) \gtrless_j q_j \in qprem(r)$. Una variable de distribución μ ocurre* libre *en una regla r si ocurre en r pero no en el destino de una premisa positiva.*

Definición 5.3.3. *Una regla r es* pura *si es bien fundada y no contiene variables libres. Un PTSS P es* puro *si todas sus reglas son puras.*

Reducción al formato $nt\mu f\theta$ puro

El Lema 5.3.1 establece que para todo PTSS en formato $nt\mu f\theta/nt\mu x\theta$ existe otro PTSS en formato $nt\mu f\theta$ que permite demostrar exactamente el mismo conjunto de reglas. Como consecuencia, ambos PTSS definen exactamente el mismo modelo bien soportado. Esto último se establece en el Corolario 5.3.1

Lema 5.3.1. *Sea $P = (\Sigma, A, R)$ un PTSS en formato $nt\mu f\theta/nt\mu x\theta$. Entonces existe un PTSS $P' = (\Sigma, A, R')$ en formato $nt\mu f\theta$ tal que para toda regla $\frac{H}{\phi}$, $P \vdash \frac{H}{\phi}$ sii $P' \vdash \frac{H}{\phi}$.*

Demostración. Para cada símbolo de función $f \in F$ se define una substitución abierta ρ_f de la siguiente manera:

$$\begin{array}{ll} \rho_f(x) = f(z_1, \ldots, z_{\mathrm{r}(f)}) & \text{si } x \text{ está en la fuente de la conclusión de } r \text{ y} \\ & z_1, \ldots, z_{\mathrm{r}(f)} \text{ son variables que no ocurren en } r. \\ \rho_f(x) = x & \text{caso contrario.} \end{array}$$

Sea R' un conjunto de reglas tal que si $r \in R$ está en formato $nt\mu f\theta$, entonces $r \in R'$. Además, para toda regla r en formato $nt\mu x\theta$ y todo símbolo de función f, agregamos la regla r' en R' donde $r' = \rho_f(r)$.

Sólo demostraremos que si $P \vdash \frac{H}{\psi}$ entonces $P' \vdash \frac{H}{\psi}$, el otro caso es similar. Sea $\mathfrak{p}(\frac{H}{\psi})$ una prueba de $P \vdash \frac{H}{\psi}$. Procedemos por inducción completa en la altura $h(\mathfrak{p}(\frac{H}{\psi}))$. Supongamos que para toda regla $\frac{H}{\phi}$ con prueba $\mathfrak{p}(\frac{H}{\phi})$ tal que $h(\mathfrak{p}(\frac{H}{\phi})) < h(\mathfrak{p}(\frac{H}{\psi}))$, $P \vdash \frac{H}{\phi}$ implica $P' \vdash \frac{H}{\phi}$. La demostración está terminada si probamos que $P \vdash \frac{H}{\psi}$ implica $P' \vdash \frac{H}{\psi}$.

Supongamos $\psi = f(t_1, \ldots, t_n) \xrightarrow{a} \pi$. Si $P \vdash \frac{H}{\psi}$ entonces la raíz del árbol de prueba $\mathfrak{p}(\frac{H}{\psi})$ está etiquetado con ψ. Sea K el conjunto de etiquetas de los nodos que se encuentran directamente sobre la raíz, entonces existe una regla r y una substitución cerrada ρ tal que $\rho(r) = \frac{K}{\psi}$ es una instancia de substitución válida de r. Para todo $\psi' \in K$, si ψ' no es un literal cuantitativo entonces $P \vdash \frac{H}{\psi'}$ con una prueba $\mathfrak{p}(\frac{H}{\psi}')$ tal que $h(\mathfrak{p}(\frac{H}{\psi'})) < h(\mathfrak{p}(\frac{H}{\psi}))$. Por hipótesis inductiva $P' \vdash \frac{H}{\psi'}$. Si r está en formato $nt\mu f\theta$ entonces $r \in R'$ y $P' \vdash \frac{H}{\psi}$. Por otro lado, si r está en formato $nt\mu x\theta$ con $\mathrm{conc}(r) = x \xrightarrow{a} \theta$, entonces tomemos la regla $r' = \rho_f(x) \in R'$ con $\rho_f(x) = f(z_1, \ldots, z_{\mathrm{r}(f)})$ y definamos la substitución ρ' como $\rho'(z_k) = t_k$, para $1 \leq k \leq \mathrm{r}(f)$ y $\rho'(x) = \rho(x)$ para toda otra variable (de término o distribución). Notemos que $\rho'(r') = \rho'(\rho_f(r)) = \rho(r)$. Entonces, si $\psi' \in \rho'(\mathrm{prem}(r'))$ no es un literal cuantitativo, $P' \vdash \frac{H}{\psi'}$. Más aún $\rho'(r')$ es una instancia de substitución válida de R', con lo cual $P' \vdash \frac{H}{\mathrm{conc}(\rho'(r'))} = \frac{H}{\psi}$. □

Corolario 5.3.1. *Sea $P = (\Sigma, A, R)$ un PTSS en formato ntμfθ/ntμxθ. Entonces existe un PTSS $P' = (\Sigma, A, R')$ en formato ntμfθ que es equivalente a P.*

Demostración. Por el Lema 5.3.1, existe P' en formato *ntμfθ/ntμxθ* tal que $P \vdash \frac{H}{\phi}$ sii $P' \vdash \frac{H}{\phi}$ para toda regla $\frac{H}{\phi}$. Luego, por el Lema 4.3.2, $P \vdash_{ws} \phi$ sii $P' \vdash_{ws} \phi$, para todo literal ϕ. Es decir, P y P' son equivalentes. □

El Lema 5.3.2 establece que para todo PTSS bien fundado existe otro puro (i.e. bien fundado y sin variables libres) que permite demostrar exactamente el mismo conjunto de reglas. Por el Lema 4.3.2 los PTSS son equivalentes.

Lema 5.3.2. *Sea $P = (\Sigma, A, R)$ un PTSS bien fundado en formato ntμfθ/ntμxθ. Entonces existe un PTSS $P' = (\Sigma, A, R')$ en formato ntμfθ/ntμxθ puro tal que para toda regla $\frac{H}{\phi}$ $P \vdash \frac{H}{\phi}$ sii $P' \vdash \frac{H}{\phi}$. Más aún, si P está en formato ntμfθ entonces P' está en formato ntμfθ.*

Demostración. Para toda regla $r \in R$, si r no tiene variables libres entonces $r \in R'$. Si $r \in R$ tiene variables libres, entonces para cada $r \in R$ y para cada substitución $\hat{\rho}$ tal que

1. $\hat{\rho}(x) = t_x \in T(\Sigma)$, para toda variable de término libre x en r,
2. $\hat{\rho}(\mu) = \pi_\mu \in \mathsf{DT}(\Sigma)$, para toda variable de distribución libre μ en r, y
3. $\hat{\rho}(x) = x$ caso contrario,

definimos $r' = \hat{\rho}(r) \in R'$. Notemos que las reglas en R' no tienen variables libres y están en formato *ntμfθ/ntμxθ*. Observar, además, que si las reglas en R están en formato *ntμfθ* entonces las reglas en R' están también en formato *ntμfθ*. El resto de la prueba continúa de la misma forma que la prueba del Lema 5.3.1. □

Corolario 5.3.2. *Sea $P = (\Sigma, A, R)$ un PTSS bien fundando en formato ntμfθ/ntμxθ. Entonces existe un PTSS $P' = (\Sigma, A, R')$ puro en formato ntμfθ/ntμxθ el cual es equivalente a P. Más aún, si P está en formato ntμfθ entonces P' está en formato ntμfθ.*

Utilizando los resultados en esta sección es directo demostrar que para todo PTSS bien fundado en formato *ntμfθ/ntμxθ* existe un PTSS puro en formato *ntμfθ* equivalente.

La relación candidata

En esta sección definimos la relación R_P, Def. 5.3.4, la cual es preservada por todas las operaciones del PTSS P. Esto se garantiza por construcción. La demostración del Teorema 5.3.2 se centra en probar que R_P es, además, una bisimulación.

Definición 5.3.4. *Sea $\Sigma = (F, \mathsf{r})$ una signatura y $P = (\Sigma, A, R)$ un PTSS con un sistema de transición asociado. La relación $\mathsf{R}_P \subseteq T(\Sigma) \times T(\Sigma)$ es la menor relación que satisface:*

(i) $\sim \subseteq \mathsf{R}_P$, *y*

(ii) para todo $f \in F$, $\left(\bigwedge_{k=1}^{\mathsf{r}(f)} u_k \; \mathsf{R}_P \; v_k\right) \Rightarrow f(u_1, \ldots, u_{\mathsf{r}(f)}) \; \mathsf{R}_P \; f(v_1, \ldots, v_{\mathsf{r}(f)})$.

La condición (II) garantiza que, por construcción, R_P es preservada por las operaciones en la signatura de P. En la demostración del Teorema 5.3.2 se procede de la siguiente forma: primero se demuestra que R_P es una bisimulación, luego por definición de $\sim$, $\sim \supseteq \mathsf{R}_P$. Además, por (I), se obtiene que $\sim = \mathsf{R}_P$ y como consecuencia $\sim$ es una congruencia.

A continuación damos algunos resultados sobre R_P que serán de utilidad en la demostración del Teorema 5.3.2. La prueba del siguiente lema es directa por inducción en la estructura de t.

Lema 5.3.3. *Sea $t \in \mathbb{T}(\Sigma)$ un término abierto y sean $\rho, \rho' : \mathcal{V} \cup \mathcal{M} \to T(\Sigma) \cup \mathcal{DT}(\Sigma)$ dos substituciones tales que $\rho(x)\ \mathsf{R}_P\ \rho'(x)$ para todo $x \in \mathcal{V}$. Entonces $\rho(t)\ \mathsf{R}_P\ \rho'(t)$.*

Lema 5.3.4. *Sea $\theta \in \mathbb{DT}(\Sigma)$ y sea $\rho, \rho' : \mathcal{V} \cup \mathcal{M} \to T(\Sigma) \cup \mathcal{DT}(\Sigma)$ dos substituciones tales que $\rho(\zeta)\ \mathsf{R}_P\ \rho'(\zeta)$ para toda variable $\zeta \in Var(\theta)$. Entonces $\rho(\theta)\ \mathsf{R}_P\ \rho'(\theta)$.*

Demostración. La prueba es por inducción estructural en $\theta \in \mathbb{DT}(\Sigma)$. El caso base se divide en dos subcasos. Si θ es una variable de distribución entonces la prueba es directa. En el otro caso, θ es una distribución de Dirac instanciable, $\theta = \delta_t$. Por Lema 5.3.3, $\rho(t)\ \mathsf{R}_P\ \rho'(t)$, luego para Q-cerrado, $\rho(t) \in Q$ sii $\rho'(t) \in Q$. Entonces $\delta_{\rho(t)}\ \mathsf{R}_P\ \delta_{\rho'(t)}$.

Para el caso inductivo, supongamos que $\rho(\zeta)\ \mathsf{R}\ \rho'(\zeta)$ para todo $\zeta \in Var(\sum_{i\in I} p_i(\prod_{n_i\in N_i} \theta_{n_i}) \circ g_i^{-1})$. Debemos demostrar que

$$\rho\left(\sum_{i\in I} p_i(\prod_{n_i\in N_i} \theta_{n_i}) \circ g_i^{-1}\right)\ \mathsf{R}_P\ \rho'\left(\sum_{i\in I} p_i(\prod_{n_i\in N_i} \theta_{n_i}) \circ g_i^{-1}\right) \tag{5.1}$$

Por hipótesis inductiva, $\rho(\theta_{n_i}) = \rho'(\theta_{n_i})$. Por Lema 5.3.3, para todo contexto g_i y $u_h, u'_h \in T(\Sigma)$ tal que $u_h\ \mathsf{R}_P\ u'_h$, $1 \le h \le N_i$, $g_i(u_1, \ldots, u_{N_i})\ \mathsf{R}_P\ g_i(u'_1, \ldots, u'_{N_i})$. Como consecuencia, para todo d tal que R_P-cerrado(d), $g_i^{-1}(d) = \biguplus_{k\in K} d_1^k \times \ldots \times d_{N_i}^k$, donde R_P-cerrado(d_h^k), para todo $k \in K$ y $1 \le h \le N_i$. Entonces, para cada $i \in I$ y conjunto cerrado R_P-cerrado(d) calculamos

$$\begin{aligned}
((\prod_{n_i\in N_i} \rho(\theta_{n_i})) \circ g_i^{-1})(d) &= (\prod_{n_i\in N_i} \rho(\theta_{n_i}))\left(\biguplus_{k\in K} d_1^k \times \ldots \times d_{N_i}^k\right) \\
&= \sum_{k\in K} (\prod_{n_i\in N_i} \rho(\theta_{n_i}))(d_1^k \times \ldots \times d_{N_i}^k) \\
&= \sum_{k\in K} \rho(\theta_1)(d_1^k) \cdot \ldots \cdot \rho(\theta_{N_i})(d_{N_i}^k) \\
&= \sum_{k\in K} \rho'(\theta_1)(d_1^k) \cdot \ldots \cdot \rho'(\theta_{N_i})(d_{N_i}^k) \\
&= ((\prod_{n_i\in N_i} \rho'(\theta_{n_i})) \circ g_i^{-1})(d)
\end{aligned}$$

□

Finalmente recordamos el siguiente resultado de [79]. El mismo nos permitirá particionar los elementos de $T(\Sigma)$ en conjuntos disjuntos utilizando la relación R_P.

Teorema 5.3.1. *[79, Theo. 3.4] Sea $R \subseteq U \times U$ una relación binaria simétrica, entonces el poset $\langle \{X : R\text{-closed}(X)\}, \subseteq \rangle$ es un álgebra de Bool completa.*

Es directo demostrar que la relación R_P es simétrica. Luego el poset $\langle \{X : \mathsf{R}_P\text{ -closed}(X)\}, \subseteq \rangle$ es un álgebra de Bool completa. Dada un álgebra de Bool $\langle \{X : \mathsf{R}_P\text{-cerrado}(X)\}, \subseteq \rangle$, denotamos con $A(\mathsf{R}_P)$ al conjunto de átomos de ésta. Con $[t]_{\mathsf{R}_P}$ denotamos al átomo de $A(\mathsf{R}_P)$ que contiene a t.

El teorema de congruencia

Sea ϕ y ϕ' dos literales positivos o dos literales negativos. Con $\phi\ \mathsf{R}_P\ \phi'$ se denotará que $src(\phi)\ \mathsf{R}_P\ src(\phi')$ y si son dos literales positivos que ambos términos ejecutan la misma acción; si son negativos, que ambos no la ejecutan.

Recordemos que se desea demostrar que para todo $f \in F, u_1, \ldots u_{\mathsf{r}(f)}, v_1, \ldots, v_{\mathsf{r}(f)} \in T(\Sigma)$:

$$1 \leq k \leq \mathsf{r}(f) : u_k \sim v_k \Rightarrow f(u_1, \ldots, u_{\mathsf{r}(f)}) \sim f(v_1, \ldots, v_{\mathsf{r}(f)})$$

Para demostrar que las propiedades de transferencia valen entre $f(u_1, \ldots, u_{\mathsf{r}(f)})$ y $f(v_1, \ldots, v_{\mathsf{r}(f)})$ se debe demostrar que si $f(u_1, \ldots, u_{\mathsf{r}(f)}) \xrightarrow{a} \pi$ entonces existe π' tal que $f(v_1, \ldots, v_{\mathsf{r}(f)}) \xrightarrow{a} \pi'$ con $\pi \sim \pi'$ y el caso simétrico. Además se debe demostrar que si $f(u_1, \ldots, u_{\mathsf{r}(f)}) \overset{a}{\not\rightarrow}$ entonces $f(v_1, \ldots, v_{\mathsf{r}(f)}) \overset{a}{\not\rightarrow}$ y viceversa.

Para el caso de los literales positivos la demostración procede por inducción en la altura de la prueba bien soportada. La complejidad de esta parte de la prueba está relacionada con garantizar que las premisas cuantitativas se satisfagan.

Para el caso de los literales negativos el problema es más complejo. En la Def. 4.3.2, para derivar un literal negativo se requiere una cuantificación universal con respecto a un conjunto de reglas derivables desde la especificación. Como los términos u_k y v_k pueden ser distintos a nivel sintáctico, en principio, no existe una relación directa entre las pruebas de la regla $\frac{H_u}{\psi_{f(u_1,\ldots,u_{\mathsf{r}(f)})}}$ y las pruebas de la regla $\frac{H_v}{\psi_{f(v_1,\ldots,v_{\mathsf{r}(f)})}}$ donde ambas reglas tienen una conclusión positiva, $src(\psi_{f(u_1,\ldots,u_{\mathsf{r}(f)})}) = f(u_1, \ldots, u_{\mathsf{r}(f)})$ y $src(\psi_{f(v_1,\ldots,v_{\mathsf{r}(f)})}) = f(v_1, \ldots, v_{\mathsf{r}(f)})$.

Sin embargo esta relación existe y es expresada por el Lema 5.3.5. Éste garantiza que si $t\ \mathsf{R}_P\ t'$ y $H/t \xrightarrow{a} \pi$ es demostrable entonces existe H' y π' tal que $H'/t' \xrightarrow{a} \pi'$ es demostrable y $\pi\ \mathsf{R}_P\ \pi'$, i.e $t \xrightarrow{a} \pi\ \mathsf{R}_P\ t' \xrightarrow{a} \pi'$. Esto se puede interpretar como una variante de la propiedad de transferencia para reglas demostrables. Además, como estas pruebas se utilizan para derivar literales negativos, se necesita asegurar que si para una premisa h de $\frac{H_u}{\psi_{f(u_1,\ldots,u_{\mathsf{r}(f)})}}$ existe una prueba bien soportada de $\neg h$, entonces también existe una prueba bien soportada de $\neg h'$ tal que h' es una premisa de $\frac{H_v}{\psi_{f(v_1,\ldots,v_{\mathsf{r}(f)})}}$. Para demostrar esto se define un mapeo desde los literales en H_v a los literales que se utilizan en la prueba $\frac{H_u}{\psi_{f(u_1,\ldots,u_{\mathsf{r}(f)})}}$. Este mapeo se establece en los items 1 y 2 del Lema 5.3.5. El mapeo, intuitivamente, establece que toda hipótesis de H_v, a nivel "semántico", debe ser utilizada en algún momento en la prueba $\frac{H_u}{\psi_{f(u_1,\ldots,u_{\mathsf{r}(f)})}}$. En este caso la semántica está definida por la relación R_P.

Dado que la estructura de prueba utilizada es similar a la utilizada en la prueba del Teorema 5.3.2 la demostración del Lema 5.3.5 se encuentra en el Apéndice A.2.

Lema 5.3.5. *Sea P un PTSS puro en formato ntμfθ/ntμxθ. Sea $H \subseteq T(\Sigma)$ un conjunto de literales negativos cerrados, $t, t' \in T(\Sigma)$, $a \in A$, $\pi \in \mathsf{DT}(\Sigma)$ tales que $P \vdash (\frac{H}{t \xrightarrow{a} \pi})$ con prueba $\mathfrak{p}(\frac{H}{t \xrightarrow{a} \pi})$ y $t\ \mathsf{R}_P\ t'$, entonces existen $H' \subseteq T(\Sigma)$, un conjunto de literales negativos, y $\pi' \in \mathsf{DT}(\Sigma)$ tales que $P' \vdash (\frac{H'}{t' \xrightarrow{a} \pi'})$ con prueba $\mathfrak{p}(\frac{H'}{t' \xrightarrow{a} \pi'})$, $\pi\ \mathsf{R}_P\ \pi'$ y para todo $h' \in H'$*

1. *existe $h \in H$ tal que $h\ \mathsf{R}_P\ h'$ o*

2. *existen literales positivos* $\eta \in \mathfrak{p}(\frac{H}{t \xrightarrow{a} \pi})$, $\eta' \in \mathfrak{p}(\frac{H'}{t' \xrightarrow{a} \pi'})$ *y conjuntos* $H_\eta \subseteq H$, $H_{\eta'} \subseteq H'$ *tales que* $src(\eta) \sim src(\eta')$, $h' \in H_{\eta'}$, $P \vdash \frac{H_\eta}{\eta}$ *y* $P \vdash \frac{H_{\eta'}}{\eta'}$.

Ahora estamos en condiciones de probar que para todo operador definido utilizando el formato $nt\mu f\theta/nt\mu x\theta$, la bisimulación es una congruencia.

Teorema 5.3.2. *Sea P un PTSS bien fundado en formato* $nt\mu f\theta/nt\mu x\theta$. *Entonces* $\sim$ *es una congruencia para todo operador definido en P.*

Demostración. Por los Corolarios 5.3.1 y 5.3.2 podemos asumir que P es puro en formato $nt\mu f\theta$. Debemos demostrar que para todo $f \in F, u_1, \ldots u_{\mathsf{r}(f)}, v_1, \ldots, v_{\mathsf{r}(f)} \in T(\Sigma)$:

$$1 \leq k \leq \mathsf{r}(f) : u_k \sim v_k \Rightarrow f(u_1, \ldots, u_{\mathsf{r}(f)}) \sim f(v_1, \ldots, v_{\mathsf{r}(f)}) \tag{5.2}$$

y para esto demostraremos que R_P (ver Def. 5.3.4) es una bisimulación. Si este es el caso, entonces $\mathsf{R}_P \subseteq \sim$. Dado que $\sim \subseteq \mathsf{R}_P$, por definición de R_P, podemos concluir que $\mathsf{R}_P = \sim$ y por lo tanto $\sim$ es una congruencia como consecuencia de que R_P es preservada por las funciones de la signatura.

Para demostrar que R_P es una bisimulación debemos demostrar que R_P satisface la propiedad de transferencia, i.e., que para todo $t, t' \in T(\Sigma)$, $a \in A$, y $\pi \in \mathsf{DT}(\Sigma)$,

$$t \; \mathsf{R}_P \; t' \text{ y } t \xrightarrow{a} \pi \;\; \Rightarrow \;\; \exists \pi' : t' \xrightarrow{a} \pi' \text{ y } \pi \; \mathsf{R}_P \; \pi'.$$

Si $t \; \mathsf{R}_P \; t'$ vale porque se satisface la condición (i) de la Def. 5.3.4, entonces $t \sim t'$ y por lo tanto $t \xrightarrow{a} \pi$ implica que existe $\pi' \in \mathsf{DT}(\Sigma)$ tal que $t' \xrightarrow{a} \pi'$ y $\pi \sim \pi'$. Dado que $\sim \subseteq \mathsf{R}_P$, R_P-cerrado(c) implica $\sim$-cerrado(c), para todo $c \subseteq T(\Sigma)$. Por lo tanto $\pi \; \mathsf{R}_P \; \pi'$, lo cual demuestra la propiedad de transferencia para el caso (i) de la Def. 5.3.4.

Supongamos ahora que $t \; \mathsf{R}_P \; t'$ porque se satisface la condición (ii) de la Def. 5.3.4, luego $t = f(u_1, \ldots, u_{\mathsf{r}(f)})$, $t' = f(v_1, \ldots, v_{\mathsf{r}(f)})$, con $f \in F$ y $u_k \; \mathsf{R}_P \; v_k$ para $1 \leq k \leq \mathsf{r}(f)$. Para demostrar la propiedad de transferencia vamos a demostrar que si $P \vdash_{ws} \psi$ con una prueba bien soportada $\mathfrak{p}(\psi)$:

(H-I) $\psi = f(u_1, \ldots, u_{\mathsf{r}(f)}) \xrightarrow{a} \pi \Rightarrow \exists \pi' : P \vdash_{ws} f(v_1, \ldots, v_{\mathsf{r}(f)}) \xrightarrow{a} \pi', \pi \; \mathsf{R}_P \; \pi'$.

(H-II) $\psi = f(u_1, \ldots, u_{\mathsf{r}(f)}) \not\xrightarrow{a} \Rightarrow P \vdash_{ws} f(v_1, \ldots, v_{\mathsf{r}(f)}) \not\xrightarrow{a}$.

Procedemos por inducción completa en la altura $h(\mathfrak{p}(\psi)) = \gamma$ de la prueba. Supongamos que la hipótesis vale para todo $\gamma' < \gamma$. Sea $P \vdash_{ws} \psi$ con una prueba $\mathfrak{p}(\psi)$ tal que $\psi = f(u_1, \ldots, u_{\mathsf{r}(f)}) \xrightarrow{a} \pi$ y $h(\mathfrak{p}(\psi)) = \gamma$. Entonces, por Def. 4.3.2, existe una regla r tal que

$$r = \frac{\bigcup_{m \in M} \{t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}} : \vec{z} \in \mathcal{Z}\} \cup \bigcup_{n \in N} \{t_n(\vec{z}) \not\xrightarrow{b_n} : \vec{z} \in \mathcal{Z}\} \{\theta_l(Y_l) \trianglerighteq_{l,k} p_{l,k} : l \in L, k \in K_l\}}{f(x_1, \ldots, x_{\mathsf{r}(f)}) \xrightarrow{a} \theta}$$

y una substitución genuina ρ tal que para todo $1 \leq k \leq \mathsf{r}(f)$, $\vec{z} \in \mathcal{Z}$, $m \in M$, $n \in N$, $l \in L$ y $k \in K_l$

(ρ1) $\rho(x_k) = u_k$,

(ρ2) $\rho(\theta) = \pi$,

(ρ3) $P \vdash_{ws} \rho(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}})$ con una prueba bien soportada $\mathfrak{p}(\rho(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}}))$, subprueba de $\mathfrak{p}(f(u_1, \ldots, u_{\mathfrak{r}(f)}) \xrightarrow{a} \pi)$, tal que $h(\mathfrak{p}(\rho(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}})) < \gamma$,

(ρ4) $P \vdash_{ws} \rho(t_n(\vec{z}) \xrightarrow{b_n}\!\!\!\!\!\!/\;\;)$ con una prueba bien soportada $\mathfrak{p}(\rho(t_n(\vec{z}) \xrightarrow{b_n}\!\!\!\!\!\!/\;\;))$, la cual es subprueba de $\mathfrak{p}(f(u_1, \ldots, u_{\mathfrak{r}(f)}) \xrightarrow{a} \pi)$, tal que $h(\mathfrak{p}(\rho(t_n(\vec{z}) \xrightarrow{b_n}\!\!\!\!\!\!/\;\;))) < \gamma$,

(ρ5) $\rho(\theta_l(Y_l)) \trianglerighteq_{l,k} p_{l,k}$.

Ahora necesitamos demostrar la existencia de una substitución genuina ρ' tal que, la cual usando la regla r y la hipótesis inductiva, permita concluir (H-ı).

El requisito que establece que la substitución tiene que ser genuina crea una dependencia entre las variables en $Var(\theta_l)$ y las variables en Y_l. No se puede definir la substitución para las variables Y_l antes de definir la substitución para las variables en $Var(\theta_l)$. Para satisfacer este requisito, la substitución ρ' se define utilizando el grafo de dependencia G de $\mathrm{pprem}(r) \cup \mathrm{qprem}(r)$.

Además, ρ' debe ser tal que las premisas cuantitativas sean satisfechas. Para esto, definamos $\rho(Y_l)/\mathsf{R}_P = \{\rho(Y_l) \cap [t]_{\mathsf{R}_P} \mid t \in T(\Sigma)\}$. Es decir, obtenemos una partición de $\rho(Y_l)$ inducida por los átomos del álgebra de Bool $\langle\{X :\mathsf{R}_P \text{-closed}(X)\}, \subseteq\rangle$. Para cada conjunto $d \in \rho(Y_l)/\mathsf{R}_P$ definamos $d^{\uparrow}$ como el único átomo del álgebra de Bool $d^{\uparrow} \in A(\mathsf{R}_P)$ tal que $d \subseteq d^{\uparrow}$. Ahora definamos Ξ, una partición $\mathcal{Z}$, tal que para todo $l \in L$ y $d_l \in \rho(Y_l)/\mathsf{R}_P$ hay un elemento de la partición $Z_{\prod_{l\in L} d_l} \in \Xi$ tal que

(Ξ1) $|Z_{\prod_{l\in L} d_l}| \geq \omega$ con $\omega = |\mathbb{N}|$.

(Ξ2) $\exists \vec{z} \in Z_{\prod_{l\in L} d_l} : \rho(\vec{z}) \in \prod_{l\in L} d_l \times \prod_{k=1}^{\mathfrak{r}(f)} \{u_k\}$

(Ξ3) si $\mu_m^{\vec{z}} \in Var(\theta)$ o $\mu_m^{\vec{z}} \in Var(\theta_l)$ para algún $l \in L$, $\vec{z} \in \mathcal{Z}$, $m \in M$ entonces si $\rho(\vec{z}) \in \prod_{l\in L} d_l \times \prod_{k=1}^{\mathfrak{r}(f)} \{u_k\}$ entonces $\vec{z} \in Z_{\prod_{l\in L} d_l}$.

(Ξ4) si $\vec{z}(i) \in Var(\theta)$ para $\vec{z} \in \mathcal{Z}$, $i \in \{1, \ldots, |z|\}$ entonces $\vec{z} \in Z_{\prod_{l\in L} d_l}$.

Notar que la cardinalidades de los conjuntos Y_l's, L (restricción 1 en la Def. 5.2.3) y M junto a que la cantidad de elementos que pueden satisfacer las condiciones (Ξ3) y (Ξ4) son finitos, garantizan que tal partición puede ser construida. La condición (Ξ1) asegurará que existen suficientes variables para definir la substitución ρ' de forma tal que las premisas cuantitativas sean satisfechas. La condición (Ξ2) será fundamental para relacionar los términos cerrados obtenidos mediante la substitución ρ' con los términos cerrados obtenidos mediante ρ a través de la relación R_P. Notemos que para cada partición de términos cerrados $\prod_{l\in L} d_l \times \prod_{k=1}^{\mathfrak{r}(f)} \{u_k\}$ existe una partición de variables $Z_{\prod_{l\in L} d_l}$ (notar el subíndice de $Z_{\prod_{l\in L} d_l}$) la cual contiene al menos un vector de variables que es instanciado por ρ a un valor en la primera partición, i.e. $\exists \vec{z} \in Z_{\prod_{l\in L} d_l}$. Luego ρ' instanciará en $Z_{\prod_{l\in L} d_l}$ los términos cerrados que se relacionan con los elementos de $\prod_{l\in L} d_l \times \prod_{k=1}^{\mathfrak{r}(f)} \{u_k\}$. Teniendo en cuenta esto, la condición (Ξ3), una vez definida ρ', implica $\rho(\vec{z})\ \mathsf{R}_P\ \rho'(\vec{z})$ para los $\vec{z}$ tales que $\mu^{\vec{z}}$ se usa en algún término de distribución. De forma similar, la condición (Ξ4) implicará, si $\vec{z}(i) \in Var(\theta)$, $\rho(\vec{z}(i)) = \rho'(\vec{z}(i))$. Ambos hechos serán importantes para demostrar los puntos relacionados a las premisas cuantitativas. Definamos ahora la substitución genuina ρ' tal que para toda variable $\zeta \in Var(r)$ se satisfacen las siguientes propiedades:

($\rho'1$) si $\zeta = x_k$ para algún $1 \leq k \leq \mathsf{r}(f)$ entonces $\rho(\zeta)\ \mathsf{R}_P\ \rho'(\zeta)$;

($\rho'2$) si $\zeta = \mu_m^{\vec{z}}$ con $m \in M$ y $\vec{z} \in \mathcal{Z}$, entonces $P \vdash_{ws} \rho'(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}})$. Más aún, si $\mu_m^{\vec{z}} \in Var(\theta)$ o $\mu_m^{\vec{z}}$ aparece en una premisa cuantitativa, entonces $\rho(\mu_m^{\vec{z}})\ \mathsf{R}_P\ \rho'(\mu_m^{\vec{z}})$.

($\rho'3$) si $\zeta = y \in Y_l$, entonces Y_l es tal que [1]:

a) $\rho'(\pi_l(Z_{\prod_{l \in L} d_l})) = d_l^{\uparrow} \cap \mathsf{support}(\rho'(\theta_l))$ (aquí, π_l indica la proyección de la l-ésima componente),

b) para todo $t_m(\vec{z})$ con $\vec{z} = \langle z_0, \ldots, z_{|\vec{z}|} \rangle$, existe $\vec{z'} = \langle z'_0, \ldots, z'_{|\vec{z}|} \rangle$ tal que para todo $i \in \{0, \ldots, |\vec{z}|\}$ si $n_{\mathrm{VDG}}(z_i) \leq n_{\mathrm{VDG}}(y)$ entonces $\rho(z'_i)\ \mathsf{R}_P\ \rho'(z_i)$.

c) si $\mu_m^{\vec{z}} \in Var(\theta)$ o $\mu_m^{\vec{z}} \in Var(\theta_l)$ para algún $l \in L$, $\vec{z} \in \mathcal{Z}$, $m \in M$ entonces si $\rho(\vec{z})\ \mathsf{R}_P\ \rho'(\vec{z})$.

d) si $y \in Var(\theta)$ entonces $\rho(y)\ \mathsf{R}_P\ \rho'(y)$

Definimos $\rho'(x)$ utilizando inducción en $n = n_{\mathrm{VDG}}(x)$ con respecto al grafo de dependencias del conjunto $\mathrm{pprem}(r) \cup \mathrm{qprem}(r)$. La definición será tal que las propiedades enunciadas son directas. Supongamos que ρ' está definida para las variables ζ' tales que $n_{\mathrm{VDG}}(\zeta') < n$. A continuación definimos ρ' para ζ tal que $n_{\mathrm{VDG}}(\zeta) = n$. Hay 3 casos que analizar $\zeta = x_i$, $\zeta = \mu_m^{\vec{z}}$ y $\zeta \in Y_{l'}$.

Si $\zeta = x_i$ para $i \in \{1, \ldots, \mathsf{r}(f)\}$. Entonces definamos $\rho'(\zeta) = v_i$. De esta forma se satisface la condición ($\rho'1$).

Si $\zeta = \mu_m^{\vec{z}}$, dado que $n_{\mathrm{VDG}}(\mu_m^{\vec{z}}) = n$, tenemos que para todo $x' \in Var(t_m(\vec{z}))$, $n_{\mathrm{VDG}}(x') < n$. Por la condición ($\rho'3b$) y la definición de R_P, podemos asegurar que existe un $\vec{z'}$ tal que $\rho'(t_m(\vec{z}))\ \mathsf{R}_P\ \rho(t_m(\vec{z'}))$. Más aún, si las condiciones de ($\rho'3c$) valen entonces $\rho(\vec{z})\ \mathsf{R}_P\ \rho(\vec{z'})$. Por ($\rho 3$), $P \vdash_{ws} \rho(t_m(\vec{z'})) \xrightarrow{a_m} \rho(\mu_m^{\vec{z'}})$, dado que $t_m(\vec{z'}) \xrightarrow{a_m} \mu_m^{\vec{z'}}$ es también una premisa positiva de r. Ahora, por definición de R_P, dos casos se presentan:

1. $\rho(t_m(\vec{z'})) \sim \rho'(t_m(\vec{z}))$. Entonces existe $\pi \in \mathsf{DT}(\Sigma)$ tal que $\rho'(t_m(\vec{z})) \xrightarrow{a_m} \pi$ y $\rho(\mu_m^{\vec{z'}}) \sim \pi$ (y por lo tanto también $\rho(\mu_m^{\vec{z'}})\ \mathsf{R}_P\ \pi$).

2. Existe un nombre de función $g \in F$ y términos $t_k, t'_k \in T(\Sigma)$, $1 \leq k \leq \mathsf{r}(g)$, tales que

   - $\rho(t_m(\vec{z'})) = g(t_1, \ldots, t_{\mathsf{r}(g)})$,
   - $\rho'(t_m(\vec{z})) = g(t'_1, \ldots, t'_{\mathsf{r}(g)})$, y
   - $t_k\ \mathsf{R}_P\ t'_k$ para todo $1 \leq k \leq \mathsf{r}(g)$.

   Además, sabemos que $h(\rho(t_m(\vec{z'})) \xrightarrow{a_m} \rho(\mu_m^{\vec{z'}})) < \gamma$, entonces por inducción en la altura de la prueba, usando particularmente (H-ı), existe una distribución $\pi \in \mathsf{DT}(\Sigma)$ tal que $P \vdash_{ws} \rho'(t_m(\vec{z})) \xrightarrow{a_m} \pi$ y $\rho(\mu_m^{\vec{z'}})\ \mathsf{R}_P\ \pi$.

Para ambos casos podemos definir $\rho'(\mu_m^{\vec{z}}) = \pi$ y obtener

[1] Notar que $\forall y' \in Y_l : n_{\mathrm{VDG}}(y) = n_{\mathrm{VDG}}(y')$, esto permite definir la substitución y la propiedad para todo el conjunto Y_l al mismo tiempo

- $P \vdash_{ws} \rho'(t_m(\vec{z})) \xrightarrow{a_m} \mu^{\vec{z}}_m)$, y
- $\rho(\mu^{\vec{z}'}_m)\ \mathsf{R}_P\ \rho'(\mu^{\vec{z}}_m)$; en particular $\rho(\mu^{\vec{z}}_m)\ \mathsf{R}_P\ \rho'(\mu^{\vec{z}}_m)$ si $\mu^{\vec{z}}_m \in Var(\theta)$ o $\mu^{\vec{z}}_m$ aparece en una premisa cuantitativa. Esto se justifica por la condición ($\rho'3c$).

Esta definición asegura la condición ($\rho'2$).

Sea $\zeta \in Y_{\hat{l}}$ para algún $\hat{l} \in L$. En este caso definimos la substitución para todo el conjunto $Y_{\hat{l}}$ al mismo tiempo. Para toda variable de término en $Y_{\hat{l}}$, definamos ρ' tal que para cada $Z_{\prod_{l\in L} d_l} \in \Xi$

$$\rho'(\pi_{\hat{l}}(Z_{\prod_{l\in L} d_l})) = d^{\uparrow}_{\hat{l}} \cap \mathsf{support}(\rho'(\theta_{\hat{l}}))$$

Esto es posible porque $|Z_{\prod_{l\in L} d_l}| \geq \omega$. Esto, sumado a las condiciones ($\Xi 2$), ($\Xi 3$) y ($\Xi 4$) implican que todas las condiciones de ($\rho'3$) se satisfacen.

La definición de ρ' asegura que $P \vdash_{ws} \rho'(t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m)$. Ahora demostraremos que las premisas negativas también son válidas, i.e. $P \vdash_{ws} \rho'(t_n(\vec{z}) \xrightarrow{b_n}\!\!\!\!\!\!/\ \ )$. Por definición de ρ' y por construcción de la partición Ξ, existe $\vec{z'} \in \mathcal{Z}$ tal que $\rho(t_n(\vec{z'}))\ \mathsf{R}_P\ \rho'(t_n(\vec{z}))$. Notar que $t_n(\vec{z'}) \xrightarrow{b_n}\!\!\!\!\!\!/$ es también una premisa en r y por lo tanto $P \vdash_{ws} \rho(t_n(\vec{z'}) \xrightarrow{b_n}\!\!\!\!\!\!/\ \ )$, por la condición ($\rho 4$). Por la definición de R_P, dos casos se presentan:

1. $\rho(t_m(\vec{z'})) \sim \rho'(t_m(\vec{z}))$, y por lo tanto $P \vdash_{ws} \rho'(t_n(\vec{z})) \xrightarrow{b_n}\!\!\!\!\!\!/$.
2. Existe un nombre de función $g \in F$ y términos $t_k, t'_k \in T(\Sigma)$, $1 \leq k \leq \mathsf{r}(g)$, tales que
   - $\rho(t_n(\vec{z'})) = g(t_1, \ldots, t_{\mathsf{r}(g)})$,
   - $\rho'(t_n(\vec{z})) = g(t'_1, \ldots, t'_{\mathsf{r}(g)})$, y
   - $t_k\ \mathsf{R}_P\ t'_k$ para todo $1 \leq k \leq \mathsf{r}(g)$.

   Es más, sabemos que $h(\rho(t_n(\vec{z'})) \xrightarrow{b_n}\!\!\!\!\!\!/\ \ ) < \gamma$, entonces por inducción en la altura de prueba, usando particularmente (H-ii), podemos concluir que $P \vdash_{ws} \rho'(t_n(\vec{z})) \xrightarrow{b_n}\!\!\!\!\!\!/$.

Por lo tanto todas las premisas negativas valen para la substitución ρ'. Es decir, para todo $n \in N$ y $\vec{z} \in \mathcal{Z}$,

($\rho'4$) $\quad P \vdash \rho'(t_n(\vec{z})) \xrightarrow{b_n}\!\!\!\!\!\!/$.

Sólo falta demostrar que las premisas cuantitativas $\rho'(\mathrm{qprem}(r))$ son válidas. Sea $\theta_l(Y_l) \trianglerighteq_{l,k} p_{l,k}$ una premisa cuantitativa de r. Si $\zeta \in Var(\theta_l)$, por la condición 7 de la Def. 5.2.3 y dado que la regla no tiene variables libres, ζ es una variable de distribución. Más aún ζ tiene que aparecer en el destino de una premisa positiva; entonces $\zeta = \mu^{\vec{z}}_m$ con $m \in M$ y $\vec{z} \in \mathcal{Z}$. Por la condición ($\rho'2$), $\rho(\mu^{\vec{z}}_m)\ \mathsf{R}_P\ \rho'(\mu^{\vec{z}}_m)$, y de aquí, por el Lema 5.3.4 tenemos que $\rho(\theta_l)\ \mathsf{R}_P\ \rho'(\theta_l)$. Además, por ($\rho 5$), $\rho(\theta_l(Y_l)) \trianglerighteq_{l,k} p_{l,k}$. El siguiente cálculo es suficiente para probar que las premisas cuantitativas de

r valen bajo ρ':

$$\begin{aligned}
\rho'(\theta_l)(\rho'(Y_l)) &= \sum_{d' \in \rho'(Y_l)/\mathsf{R}_P} \rho'(\theta_l)(d') \\
&= \sum_{d \in \rho(Y_l)/\mathsf{R}_P} \rho'(\theta_l)(d^{\uparrow}) && \text{(por } (\rho' 3a)\text{)} \\
&= \sum_{d \in \rho(Y_l)/\mathsf{R}_P} \rho(\theta_l)(d^{\uparrow}) && \text{(por Lema 5.3.4)} \\
&\geq \sum_{d \in \rho(Y_l)/\mathsf{R}_P} \rho(\theta_l)(d) && (d \subseteq d^{\uparrow}) \\
&= \rho(\theta_l)(\rho(Y_l)) \\
&\trianglerighteq_{l,k} p_{l,k} && \text{(dado que } \rho(\theta_l(Y_l)) \trianglerighteq_{l,k} p_{l,k}\text{)}
\end{aligned}$$

Por lo tanto, para todo $l \in L$, $k \in K_l$ y $\vec{z} \in \mathcal{Z}$, se cumple que $\rho'(\theta_l(Y_l)) \trianglerighteq_{l,k} p_{l,k}$. Dado que todas las premisas de $\rho'(r)$ valen, podemos concluir que $P \vdash H'/\rho'(\mathrm{conc}(r))$.

Solamente resta demostrar que $\rho(\theta)\ \mathsf{R}_P\ \rho'(\theta)$, pero por Lema 5.3.4 esto es directo. Recordemos que la regla no tiene variables libres, entonces si $\zeta \in Var(\theta)$ y ζ es una variable de distribución, tenemos que $\zeta = \mu^{\vec{z}}_m$ para algún $\vec{z} \in \mathcal{Z}$ y $m \in M$. Por condición $(\rho' 2)$, $\rho(\mu^{\vec{z}}_m)\ \mathsf{R}_P\ \rho'(\mu^{\vec{z}}_m)$. Por otro lado, si ζ es una variable de término, por las condiciones $(\rho' 1)$ y $(\rho' 3d)$, $\rho(\zeta)\ \mathsf{R}_P\ \rho'(\zeta)$. Es decir que las condiciones del Lema 5.3.4 son satisfechas. Esto concluye la prueba del item (H-I) de la inducción.

Demostremos el caso (H-II). Para este caso, $P \vdash_{ws} \psi$ con una prueba $\mathfrak{p}(\psi)$ tal que $\psi = f(u_1, \ldots, u_{\mathsf{r}(k)}) \overset{a}{\not\to}$ y $h(\mathfrak{p}(\psi)) = \gamma$. Por Def. 4.3.2, para todo conjunto de literales negativos H y $\pi \in \mathsf{DT}(\Sigma)$ tales que $P \vdash \frac{H}{f(u_1,\ldots,u_{\mathsf{r}(k)}) \overset{a}{\longrightarrow} \pi}$ existe una etiqueta h tal que $\neg h \in H$ y $P \vdash_{ws} h$.

Sea H_v un conjunto de literales negativos tal que $P \vdash \frac{H_v}{f(v_1,\ldots,v_{\mathsf{r}(f)}) \overset{a}{\longrightarrow} \pi}$ con una prueba bien soportada $\mathfrak{p}(\frac{H_v}{f(v_1,\ldots,v_{\mathsf{r}(f)}) \overset{a}{\longrightarrow} \pi})$. Por Lema 5.3.5, existe un conjunto de literales cerrados H_u y una distribución $\pi' \in \mathsf{DT}(\Sigma)$ tales que $P \vdash \frac{H_u}{f(u_1,\ldots,u_{\mathsf{r}(f)}) \overset{a}{\longrightarrow} \pi'}$ con una prueba bien soportada $\mathfrak{p}(\frac{H_u}{f(u_1,\ldots,u_{\mathsf{r}(f)}) \overset{a}{\longrightarrow} \pi'})$, $\pi\ \mathsf{R}_P\ \pi'$ y para todo $h_u \in H_u$

1. existe $h_v \in H_v$ tal que $h_v\ \mathsf{R}_P\ h_u$ o
2. existen literales positivos $\eta_v \in \mathfrak{p}(\frac{H_v}{f(v_1,\ldots,v_{\mathsf{r}(k)}) \overset{a}{\longrightarrow} \pi})$, $\eta_u \in \mathfrak{p}(\frac{H_u}{f(u_1,\ldots,u_{\mathsf{r}(k)}) \overset{a}{\longrightarrow} \pi'})$ y conjuntos $H_{\eta_v} \subseteq H_v$, $H_{\eta_u} \subseteq H_u$ tales que $src(\eta_v) \sim src(\eta_u)$, $h_u \in H_{\eta_u}$, $P \vdash \frac{H_{\eta_v}}{\eta_v}$ y $P \vdash \frac{H_{\eta_u}}{\eta_u}$.

Sean $H'_u, H''_u \subseteq H_u$ tales que todo literal en H'_u satisface el item (1) y $H''_u = H_u - H'_u$.

Si existe literal $h_u \in H'_u$ tal que $P \vdash_{ws} \neg h_u$ con prueba bien soportada $\mathfrak{p}(h_u)$, $h(\mathfrak{p}(h_u)) < \gamma$, por hipótesis inductiva $P \vdash_{ws} \neg h_v$, con lo cual se niega $h_v \in H_v$. Supongamos que esto no es cierto. Notemos que $H''_u \neq \emptyset$ porque sino ninguna premisa de $H_u = H'_u$ sería negable y por lo tanto no valdría $P \vdash_{ws} f(u_1, \ldots, u_{\mathsf{r}\,f}) \overset{a}{\not\to}$. Sea $\mathcal{U}$ el conjunto compuesto por los $\eta_u \in \mathfrak{p}(\frac{H_u}{f(u_1,\ldots,u_{\mathsf{r}(k)}) \overset{a}{\longrightarrow} \pi'})$ de (2). Dado que $P \vdash_{ws} f(u_1, \ldots, u_{\mathsf{r}\,f}) \overset{a}{\not\to}$ existe $\hat{\eta}_u \in \mathcal{U}$ tal que para todo conjunto de literales

negativos $\hat{H}_u$ tal que $P \vdash \frac{\hat{H}_u}{\hat{\eta}_u}$ existe literal positivo $\hat{h}_u$ tal que $\neg\hat{h}_u \in \hat{H}_u$ y $P \vdash_{ws} \hat{h}_u$ $(*)$. Por (2), existe $\hat{\eta}_v$ tal que $P \vdash \frac{H_{\hat{\eta}_v}}{\eta_v}$, $H_{\hat{\eta}_v} \subseteq H_v$ y $src(\hat{\eta}_v) \sim src(\hat{\eta}_u)$. Por $(*)$ debe existir $\hat{h}_v$ tal que $\neg\hat{h}_v \in \hat{H}_v$ y $P \vdash_{ws} \hat{h}_v$ (caso contrario $\hat{\eta}_v \not\sim \hat{\eta}_u$). Entonces siempre se puede negar una premisa de H_v.

□

5.4. Observaciones finales

5.4.1. Trabajos relacionados

Los formatos de reglas para sistemas de transiciones probabilistas han recibido relativamente poca atención. Entre éstos se encuentran RTSS [53], PGSOS [10], y Segala-GSOS [10].

En el formato RTSS [53], los literales utilizados en la relación de transición son de la forma $t \xrightarrow{a,p}_{\xi} t'$ donde p es la probabilidad de alcanzar t' luego de ejecutar la acción a y ξ es un índice utilizado para distinguir diferentes ocurrencias del mismo salto probabilístico. Por lo tanto, una transición probabilista completa $t \xrightarrow{a} \mu_{t,a}$ está definida para todo $t' \in T(\Sigma)$ por $\mu_{t,a}(t') = \sum_{t \xrightarrow{a,p}_{\xi} t'} p$. Las reglas en este formato tienen conclusiones de la forma $f(x_1, \ldots, x_{\mathrm{r}(f)}) \xrightarrow{a,p}_{\iota} t$, con p un término abierto que finalmente será un valor real y ι una variable de índice. Se soportan premisas negativas y positivas, pero en la fuente sólo se permiten utilizar variables incluidas en $\{x_1, \ldots, x_{\mathrm{r}(f)}\}$. Cada regla define sólo una transición. La versión más simple no soporta double testing, ni lookahead. Se presenta una variante que soporta double testing y otra que soporta lookahead de profundidad 1, La restricción sobre la profundidad es debido a que el caso general requiere cálculos más complejos.

El formato PGSOS [10] es análogo al formato RTSS con double testing. El mismo se deriva utilizando un enfoque basado en Teoría de Categorías, más precisamente un enfoque bi-algebraico.

El formato Segala-GSOS [10] también se desarrolla utilizando un enfoque bi-algebraico. A pesar de esto, la forma de representar transiciones es la misma que utiliza el formato *ntμfθ/ntμxθ*. Las conclusiones de las regla son de la forma $f(x_1, \ldots, x_{\mathrm{r}(f)}) \xrightarrow{a} \theta$ donde θ es un caso particular de un término de distribución. Soporta premisas negativas y positivas, pero en la fuente sólo se permiten utilizar variables de estado incluidas en $\{x_1, \ldots, x_{\mathrm{r}(f)}\}$. No soporta lookahead, aunque sí double testing y premisas cuantitativas que verifican si la probabilidad de alcanzar un estado particular es mayor a cero.

El formato *ntμfθ/ntμxθ* posee una expresividad superior: soporta términos en la fuente de premisas positivas y negativas contra premisas del mismo tipo con sólo variables y premisas cuantitativas que verifican valores arbitrarios (contra premisas cuantitativas que solo verifican si la probabilidad es mayor que cero sólo en el caso del formato Segala-GSOS). Además de esto, soporta lookahead arbitrario y double testing.

5.4.2. Conclusiones

En esta sección se presento el formato *ntμfθ/ntμxθ* y se demostró que la bisimulación es una congruencia para todo operador cuya semántica se define utilizando en formato.

En [22] se introduce el teorema de congruencia para una versión previa del formato *ntμfθ/ntμxθ*. Este formato se denominaba *ntμfv/ntμxv* y sólo soportaba una variable de distribución en las premisas cuantitativas y términos de distribuciones con una forma particular en el destino de la conclusión. No sólo esto, para derivar un modelo a partir de un PTSS se utilizaba el método basado en estratificación. Este trabajo seguía la estructura de [43], donde se demuestra el resultado análogo para el formato *ntyft/ntyxt*. Adaptar el resultado presentaba dos problemas. El primero era adaptar reducciones similares a las que se establecen en los corolarios 5.3.1 y 5.3.2 de este trabajo. La restricciones del formato sobre las premisas cuantitativas y el destino de la conclusión no permitían realizar las reducciones de forma directa. El segundo problema estaba asociado a la manipulación de los elementos probabilísticos del modelo, los cuales eran nuevos con respecto al trabajo anterior.

Este capítulo también sigue la estructura de [43] pero con una salvedad importante: dado un PTSS el PTS asociado se derivaba utilizando el método basado en pruebas bien soportadas. A pesar de este cambio, las reducciones expresadas en los corolarios 5.3.1 y 5.3.2 poseen más similitudes con las reducciones originales de [43] que con las presentadas para el formato *ntμfv/ntμxv*. Esto se debe a la introducción de los términos de distribuciones, los cuales permiten definir reglas con menos restricciones. Por otro lado, la demostración del Teorema 5.3.2 difiere bastante con respecto a las presentada en [43], dado que los métodos para derivar los PTSS son totalmente diferentes. Cabe remarcar que la manipulación de los elementos probabilísticos del modelo es la misma que se utiliza para demostrar el resultado análogo de [22].

Un aspecto positivo del formato *ntμfθ/ntμxθ* es que posee una estructura similar al formato *ntyft/ntyxt*, el cual ha sido ampliamente estudiado. Esta similitud permite la adaptación de muchos resultados del contexto no probabilístico al contexto probabilístico. El Teorema 5.3.2 es un ejemplo significativo de esto. En los siguientes capítulos veremos más resultados que avalan esta afirmación.

Capítulo 6

El formato $nt\mu f\theta/nt\mu x\theta$ reduce al formato pntree

Una regla está en formato *pntree*, si la regla está en $nt\mu f\theta/nt\mu x\theta$, es bien formada, no tiene variables libres y las fuentes de sus premisas positivas son simplemente una variable. En esta sección demostraremos que para toda especificación en formato $nt\mu f\theta/nt\mu x\theta$ existe una especificación equivalente en formato pntree: Teorema 6.2.1. Como corolario de este resultado se obtendrá que no es necesario que la especificación sea bien fundada para que la bisimulación sea una congruencia para toda función de la signatura. Es decir, el corolario generaliza el Teorema 5.3.2 a todo PTSS en formato $nt\mu f\theta/nt\mu x\theta$.

En la Sección 6.1 se introducen las *estructuras de pruebas*. Esta nueva estructura será esencial para demostrar el resultado. En la sección 6.2 se desarrolla la reducción.

6.1. Estructuras de prueba

Una *estructura de prueba* es un árbol de derivación en el cual las reglas no comparten variables. La relación entre la conclusión de una regla r y una premisa ψ en otra regla se representa mediante una función ϕ de reglas a literales, i.e. $\phi(r) = \psi$. Una substitución σ *coincide* con una estructura de prueba si $\phi(r) = \psi$ implica $\sigma(\phi(r)) = \sigma(\psi)$. Luego, a partir de una substitución que coincide con una estructura de prueba se puede crear un árbol de derivación. De esta forma, este árbol de derivación define una regla demostrable (Def. 4.3.1) en la cual las premisas de la regla son las hojas del árbol y la conclusión es la raíz. La ausencia de variables comunes entre las reglas permitirá definir una substitución sobre una estructura de prueba de forma incremental, evitando conflictos de redefiniciones en la substitución.

Las reglas demostrables serán usadas de la siguiente manera a lo largo de la demostración del resultado: dado un PTSS P tomaremos el conjunto de reglas demostrables en P con un formato particular. Estas reglas serán utilizadas para definir un nuevo PTSS P', luego se demostrará que P y P' son equivalentes.

Diremos que un PTSS es *pequeño* si para cada una de sus reglas, su conjunto de premisas no excede la cardinalidad del conjunto de variables V. De esta forma, un PTSS pequeño nos asegura tener suficientes variables para construir las estructuras de pruebas. Notar que todo PTSS puede ser pequeño incrementado la cantidad de variables en V.

Definición 6.1.1. *Una* estructura de prueba *es una tupla* $\langle B, r, \phi\rangle$ *tal que*

- $r \in B$ *y* B *es un conjunto de reglas las cuales no tienen variables en común,*
- ϕ *es una función inyectiva de* $B \setminus \{r\}$ *a las premisas positivas en las reglas de* B*, tales que cada cadena* $b_0, b_1, \ldots$ *de* B*, con* $\phi(b_{i+1})$ *una premisa de* b_i*, es una cadena finita.*

Sea $top(B, r, \phi)$ *el conjunto de todas las premisas de las reglas de* B *que están fuera de la imagen de* ϕ*. Sea* $qtop(B, r, \phi)$ *el conjunto de las premisas cuantitativas en* $top(B, r, \phi)$*.*

A continuación introducimos un orden parcial $<$ en una estructura de prueba, el cual permite realizar razonamientos inductivos sobre una estructura. El orden parcial $<$ está definido por $(B', r', \phi') < (B, r, \phi)$ si y solo si $B' \subset B$, ϕ' es ϕ restringido a $B' \setminus \{r'\}$, $\text{top}(B', r', \phi') \subseteq \text{top}(B, r, \phi)$, y existe una cadena $b_0, b_1, \ldots, b_n$ con $b_0 = r$, $b_n = r'$, $n > 0$ y $\phi(b_{i+1})$ es una premisa de b_i.

Una substitución σ *coincide* con la estructura de prueba (B, r, ϕ) si $\sigma(\text{conc}(b)) = \sigma(\phi(b))$ para toda regla $b \in B \setminus \{r\}$.

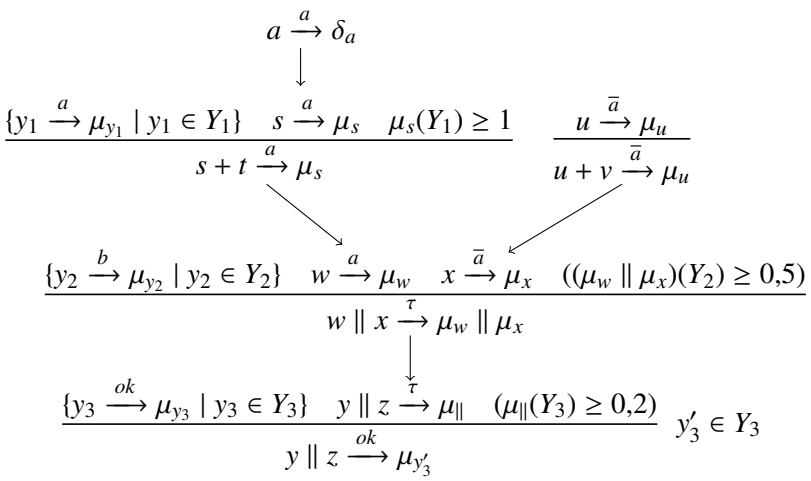
Definición 6.1.2. *Sea* $P = (\Sigma, A, R)$ *un PTSS pequeño. Sea* ψ *un literal positivo y sean* $H = H_p \cup H_n \cup H_q$ *un conjunto de literales tales que* H_p*,* H_n *y* H_q *son conjuntos de literales positivos, negativos y cuantitativos abiertos, respectivamente. Una regla* $\frac{H}{\psi}$ *es* demostrable (utilizando estructuras de prueba) *en* P*, notación* $P \vdash_{ep} \frac{H}{\psi}$*, si* $\psi \in H$ *o existe una estructura de prueba* (B, r, ϕ) *tal que cada regla en* B *es una regla en* R *módulo una conversión* α *y existe una substitución* σ *que coincide con* (B, r, ϕ) *tal que:*

1. *la raíz de la estructura de prueba instanciada usando* σ *es* ψ*, i.e.* $\sigma(conc(r)) = \psi$*, y*
2. *para todo* $\chi \in \sigma(top(B, r, \phi))$*,* $\chi \in H$ *o* χ *es una premisa cuantitativa válida.*

Ejemplo 6.1.1. *Sea* $P = \langle \Sigma, A, R\rangle$ *un PTSS tal que* $\{a, +, \|\} \subseteq \Sigma$*,* $\{a, \overline{a}, b, \tau, ok\} \subseteq A$ *y todas las reglas de la Fig. 6.1 pertenecen a* R*. Sea* (B, r, ϕ) *la estructura de prueba de la Fig. 6.1 donde el mapeo* ϕ *está representado por las flechas. Sea* σ *la substitución definida en la Fig. 6.1, con* $\sigma(\zeta) = \zeta$ *para cualquier variable (de término o de distribución) que no se encuentra definida en la figura. La siguiente regla es demostrable en* P*:*

$$\frac{\begin{array}{c} u \xrightarrow{\overline{a}} \mu_u \quad \{y_2 \xrightarrow{b} \mu_{y_2} \mid y_2 \in Y_2\} \quad ((\delta_a \parallel \mu_u)(Y_2) \geq 0{,}5) \\ \{y_3 \xrightarrow{ok} \mu_{y_3} \mid y_3 \in Y_3\} \quad ((\delta_a \parallel \mu_u)(Y_3) \geq 0{,}2) \end{array}}{(a + t) \parallel (u + v) \xrightarrow{ok} \mu_{y'_3}} \tag{6.1}$$

Dado que $\sigma(y_1) = a$ *para todo* $y_1 \in Y_1$ *y* $\sigma(\mu_s) = \delta_a$*, entonces* $\sigma(\mu_s(Y_1) \geq 1) = (\delta_a(\{a\}) \geq 1)$ *es cerrada y, más aún, es válida. Por esta razón no aparece en las premisas de la regla (6.1). Además la variable* $\mu_\parallel$ *es substituida por* $(\delta_a \parallel \mu_u)$*.*



$$\sigma(s) = a \qquad \sigma(\mu_s) = \delta_a$$

$$\sigma(y_1) = a \text{ para todo } y_1 \in Y_1 \qquad \sigma(\mu_{y_1}) = \delta_a \text{ para todo } y_1 \in Y_1$$

$$\sigma(w) = a + t \qquad \sigma(\mu_w) = \delta_a \qquad \sigma(x) = u + v \qquad \sigma(\mu_x) = \mu_u$$

$$\sigma(y) = a + t \qquad \sigma(z) = u + v \qquad \sigma(\mu_{\parallel}) = \delta_a \parallel \mu_u$$

Figura 6.1.: Ejemplo de una estructura de prueba. (Ver Ejemplo 6.1.1)

Las estructuras de pruebas tienen como desventajas que no permiten un razonamiento inductivo. Notemos que dadas n estructuras de pruebas, las mismas no se pueden utilizar para construir una nueva estructura de prueba debido a que se debe garantizar que todas las variables son diferentes. Para solucionar este problema se introduce el siguiente lema que establece que la definición de prueba utilizando estructuras de prueba (Def. 6.1.2) y la definición de prueba (Def. 4.3.1) son equivalentes. La prueba de este lema es análoga a la prueba del mismo resultado en un contexto no probabilístico (Ver [34, Proposición 2.9])

Lema 6.1.1. *Sea $P = \langle \Sigma, A, R \rangle$ un PTSS pequeño y sea $\frac{H}{\psi}$ una regla con ψ un literal positivo. Entonces $P \vdash \frac{H}{\psi}$ sii $P \vdash_{ep} \frac{H}{\psi}$.*

Luego no tendrá sentido diferenciar entre prueba y prueba utilizando estructuras de prueba. Lo mismo vale para demostrable y demostrable utilizando estructuras de prueba. Durante este Capitulo utilizaremos los términos "prueba" y "demostrable" y el símbolo "$\vdash$" para referirnos a las definiciones que utilizan estructuras de pruebas. A pesar de este cambio de denominación, la definición de prueba bien soportada (Def.4.3.2) y el Lema 4.3.2 siguen siendo válidos por el Lema 6.1.1.

Dado que el Lema 4.3.2 permite realizar razonamientos inductivos demostrar el siguiente resultado es directo.

Lema 6.1.2. *Sean P y P' dos PTSS tales que todas las reglas de P' son demostrables en P. Entonces toda regla demostrable en P' es demostrable en P.*

6.2. Reducción al formato pntree

El lema 6.2.1 establece un resultado que pertenece a la Teoría de Unificación sobre dominios infinitos [33] y será necesarios para la prueba del resultado principal de esta sección. El mismo se presenta en [34, Lemma 3.2]

Definición 6.2.1. *Una substitución σ es un* unificador *de una substitución ρ si $\sigma\rho = \sigma$. En es caso decimos que ρ es* unificable*.*

Lema 6.2.1. *Si una substitución ρ es unificable, entonces existe un unificador $\hat{\sigma}$ de ρ tal que:*

1. *cada unificador σ de ρ es también un unificador de $\hat{\sigma}$*
2. *si $\rho(\zeta) = \zeta$ entonces $\hat{\sigma}(\zeta) = \zeta$, para todo $\zeta \in \mathcal{V} \cup \mathcal{M}$, y*
3. *si $\rho^n(\zeta)$ es una variable para todo $n \geq 0$ entonces $\hat{\sigma}(\zeta)$ es una variable.*

Diremos que $\hat{\sigma}$ es el unificador más general.

A continuación se definen casos particulares del formato *ntμfθ/ntμxθ*, entre ellos el formato *pntree*.

Definición 6.2.2. *Sea un PTSS $P = \langle \Sigma, A, R \rangle$ en formato ntμfθ.*

1. *Una regla $r \in R$ está en* formato *nxμfθ si la fuente de sus premisas positivas son variables.*

2. *Una regla $r \in R$ está en* formato pntree *si está en* formato *nxμfθ, es bien fundada y no tiene variables libres.*

Un PTSS P está en formato *nxμfθ* (resp. formato pntree) *si todas las reglas en R están en formato nxμfθ (resp. formato pntree).*

El Teorema 6.2.1, el cual enuncia que todo PTSS en formato *ntμfθ/ntμxθ* puede reducirse a uno equivalente en formato pntree, se demuestra de forma incremental.

1. Utilizando el Corolario 5.3.1, toda especificación en formato *ntμfθ/ntμxθ* puede reducirse a una especificación equivalente en formato *ntμfθ*.

2. Luego se demuestra que para todo PTSS P en formato *ntμfθ* existe un PTSS P' en formato *nxμfθ* tal que $P \vdash \frac{H}{c}$ sii $P' \vdash \frac{H}{c}$ para toda regla $\frac{H}{c}$ en formato *nxμfθ* (Lema 6.2.2). Esto implica $P \vdash \frac{H'}{c}$ sii $P' \vdash \frac{H'}{c}$ para todo regla $\frac{H'}{c}$ con H' un conjunto de premisas negativas. Por el Lema 4.3.2, P y P' son equivalentes.

3. Finalmente se demuestra que para todo PTSS P en formato *nxμfθ* existe un PTSS P' en formato pntree, tal que para toda regla cerrada $\frac{H}{c}$ con H conteniendo sólo premisas negativas, $P \vdash \frac{H}{c}$ sii $P' \vdash \frac{H}{c}$ (Lemma 6.2.3). Una vez más, por Lema 4.3.2, P y P' son equivalentes.

Esta serie de lemas converge en el Teorema principal el cual establece que todo PTSS en formato *ntμfθ/ntμxθ* puede reducirse a uno equivalente en formato pntree. Notemos que las reglas en formato pntree son siempre bien fundadas. Como consecuencia de esto, se puede concluir que un PTSS en formato *ntμfθ/ntμxθ* no debe ser bien fundado para satisfacer que la bisimulación es una congruencia para todo operador definido por éste.

La idea para probar el Lema 6.2.2 es la siguiente: dado un PTSS con un conjunto de reglas R, se define R' como el conjunto de todas las reglas demostrables desde R tales que satisfacen el formato *nxμfθ*. Por el Lema 6.1.2 es directo que toda regla demostrable en R' es demostrable en R. La vuelta es más compleja. Dada una prueba en R, se recorre la prueba desde abajo hacia arriba, empezando desde la regla que se utilizo para derivar la raíz. Una premisa incluye una nueva regla (se sube un nivel) si la misma no satisface las condiciones del formato; si la satisface, se frena. De esta forma se crea un árbol de derivación que deriva una regla que satisface el formato, por tanto la regla pertenece a R'. Notemos que las premisas que necesita la nueva regla se pueden derivar utilizando una estructura de prueba de menor tamaño que la original, luego, por hipótesis inductiva, las premisas son derivables en R'.

Lema 6.2.2. *Sea $P = (\Sigma, A, R)$ un PTSS en formato ntμfθ. Entonces existe un PTSS $P' = (\Sigma, A, R')$ en formato nxμfθ tal que $P \vdash \frac{H}{c}$ sii $P' \vdash \frac{H}{c}$ para toda regla $\frac{H}{c}$ en formato nxμfθ.*

Demostración. Definamos $P' = (\Sigma, A, R')$ de forma tal que $r \in R'$ sii r es una regla en formato *nxμfθ* demostrable en P . La implicación de derecha a izquierda es directa por el Lema 6.1.2.

Para la otro implicación procedemos por inducción en el orden parcial sobre la estructura de prueba. Supongamos $P \vdash \frac{H}{\psi}$, donde $\frac{H}{\psi}$ es una regla en formato *nxμfθ* y sea (B, r, ϕ) una estructura de prueba para $\frac{H}{\psi}$ en P. Por Def. 6.1.2 existe una substitución σ que coincide con (B, r, ϕ) y

- $\sigma(\text{conc}(r)) = \psi$.
- Para todo $\chi \in \sigma(\text{top}(B, r, \phi))$, $\chi \in H$ o χ es una premisa cuantitativa válida.

Utilizando (B, r, ϕ) construiremos de forma recursiva una sub-estructura de prueba (B', r, ϕ') la cual es una estructura de prueba rara una regla $r' \in R'$. i.e. r' está en formato $nx\mu f\theta$, su conclusión es tal que $\sigma(\text{conc}(r')) = \psi$ y para cada premisa ψ' de $\sigma(\text{prem}(r'))$ vale algunas de las siguientes condiciones: ψ' es un literal cuantitativo cerrado válido o la regla $\frac{H}{\psi'}$ es demostrable R'. Luego, por Lema 6.1.1, $\frac{H}{\psi}$ es demostrable en P'. Además, construiremos en etapas una substitución ρ la cual es unificada por σ, i.e. si $\rho(x)$ está definido entonces $\sigma(\rho(x)) = \sigma(x)$. En esta construcción definimos ρ^0 como la función identidad.

Procedemos con la definición de las reglas en B' y la definición de la substitución ρ:

(I) $r \in B'$.

(II) Si $b \in B \setminus \{r\}$ y $\phi(b)$ es una premisa $t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m$ de una regla en B' tal que para algún $k \geq 0$,

   a) $\rho^i(t_m(\vec{z}))$ está definido para todo $i = 0, \ldots, k$

   b) $\rho^i(t_m(\vec{z}))$ es una variable para todo $i = 0, \ldots, k-1$

   c) $\rho^k(t_m(\vec{z}))$ tiene la forma $f(t_1, \ldots, t_{\mathsf{r}(f)})$ con $t_i \in \mathbb{T}(\Sigma)$

   entonces $b \in B'$. Notemos que las condiciones pueden ser satisfechas sólo si $\rho^i(t_m(\vec{z}))$ es una variable para $i = 0, \ldots, k-1$. Esto implica que $\rho^0(t_m(\vec{z})) = t_m(\vec{z})$ es una variable. Por como fue definida la notación, la variable pertenece a $\vec{z}$.

(III) Dado que σ coincide con (B, r, ϕ), $\sigma(\text{conc}(b)) = \sigma(t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m)$. Dado que el formato de regla restringe la forma de la conclusión conc(b), podemos reescribir la última igualdad como: $\sigma(f(x_1, \ldots, x_{\mathsf{r}(f)}) \xrightarrow{a} \theta) = \sigma(t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m)$ Además, σ unifica la substitución parcial ρ, entonces si $\rho^{k-1}(t_m(\vec{z}))$ es una variable, $\sigma(t_m(\vec{z})) = \sigma\rho^k(t_m(\vec{z})) = \sigma(f(t_1, \ldots, t_n))$.

   Dado que la conclusión conc(b) tiene la forma $f(x_1, \ldots, x_{\mathsf{r}(f)}) \xrightarrow{a} \theta$, $\sigma(x_j) = \sigma(t_j)$ para $j = 1, \ldots, \mathsf{r}(f)$ y $\sigma(\theta) = \sigma(\mu^{\vec{z}}_m)$. Definamos $\rho(x_j) = t_j$ para $j = 1, \ldots, \mathsf{r}(f)$ (aquí definimos el lado izquierdo de una conclusión de una regla en $B' \setminus \{r\}$). Además, definamos $\rho(\mu^{\vec{z}}_m) = \theta$. Notemos que esta extensión de ρ es unificada por σ y, por Def. 6.1.2, las variables x_j y $\mu^{\vec{z}}_m$ aparecen sólo en esta regla, entonces la substitución ρ queda bien definida.

(IV) Definamos $\rho(\zeta) = \zeta$ para toda variable ζ si ζ no está definida para ρ. La substitución σ también unifica esta extensión de ρ.

(V) Por último, ϕ' es la restricción de ϕ a $B' \setminus \{r\}$. (Notemos que la substitución ρ es definida para el lado derecho de una premisa positiva en la imagen de ϕ' en el item (III).)

La substitución σ unifica a la substitución ρ, por el Lema 6.2.1, existe una substitución ρ' la cual unifica a ρ y:

(ρ'I) $\sigma\rho' = \sigma$.

(ρ'II) Si $\rho(\zeta) = \zeta$ entonces $\rho'(\zeta) = \zeta$, con ζ una variable de término o distribución.

(ρ'III) Si $\rho^k(\zeta)$ es una variable para $k \geq 0$ entonces $\rho'(\zeta)$ es una variable.

La estructura de prueba (B', r, ϕ') y la substitución ρ' están completamente definidas, entonces podemos demostrar que ρ' coincide con (B', r, ϕ'). Sea b una regla usada para construir B' y consideremos la substitución ρ. Recordemos que la conclusión de b tiene la forma $f(x_1, \ldots, x_{\mathrm{r}(f)}) \xrightarrow{a} \theta$ y $\phi'(b) = t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}}$ es tal que

$$\rho^k(t_m(\vec{z})) = f(t_1, \ldots, t_{\mathrm{r}(f)}) = \rho(f(x_1, \ldots, x_{\mathrm{r}(f)}))$$

por los items (II) y (III). Dado que ρ' unifica a ρ tenemos que:

$$\begin{array}{rcccccl} \rho'(\phi'(b)) & = & \rho'(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}}) & = & \rho'(\rho^k(t_m(\vec{z})) \xrightarrow{a_m} \rho(\mu_m^{\vec{z}})) & & \\ & = & \rho'(\rho(f(x_1, \ldots, x_{\mathrm{r}(f)}) \xrightarrow{a_m} \theta) & = & \rho'(f(x_1, \ldots, x_{\mathrm{r}(f)}) \xrightarrow{a_m} \theta) & = & \rho'(\mathrm{conc}(b)) \end{array}$$

Entonces la substitución ρ' coincide con la estructura de prueba (B', r, ϕ'). Además, por construcción de ρ, sabemos que si x es tal que $\rho(x) \neq x$ entonces x satisface una de las siguientes condiciones:

1. x aparece en el lado izquierdo de una conclusión de una regla en $B' \setminus \{r\}$,
2. x aparece en el lado derecho de una premisa positiva en la imagen de ϕ'.

Sea $(\theta(Y) \trianglerighteq p) \in \mathrm{qtop}(B', r, \phi')$. Si $\zeta \in Var(\theta)$ es una variable que aparece en el lado derecho de una premisa positiva en la imagen de ϕ', i.e. ζ es una variable de distribución, tenemos que $\rho'(\zeta) \in \mathbb{DT}(\Sigma)$; en cambio si $\zeta \in Var(\theta)$ es una variable de estado entonces $\rho'(\zeta) \in \mathbb{T}(\Sigma)$. Para ambos casos $\rho'(\theta) \in \mathbb{DT}(\Sigma)$. Además, $\rho(y) = y$ para $y \in Y$ debido a que estas variables no aparecen del lado izquierdo de la conclusión, por lo tanto $\rho'(y) = y$. Luego, $\rho'(\theta(Y) \trianglerighteq p) \in \mathbb{DT}(\Sigma)$, es decir, la premisa no es cerrada y por lo tanto no debemos verificar si es válida o no. Luego, la regla $s = \rho'\left(\frac{\{h : h \in \mathrm{top}(B', r, \phi')\}}{\mathrm{conc}(r)}\right)$ es demostrable (Def. 6.1.2).

Notemos que $\rho'(\theta(Y) \trianglerighteq p)$ satisface las restricciones sintácticas del formato $nx\mu f\theta$. Demostremos que lo mismo ocurre con la conclusión. Sea $g(x_1, \ldots, x_m) \xrightarrow{b} \theta$ la conclusión de r. Luego $\rho(x_j) = x_j$ para $j = 1, \ldots, m$ y, por lo tanto $\rho'(x_j) = x_j$ por (ρ'II). Por otro lado, para θ se puede repetir el razonamiento utilizado para las premisas cuantitativas y de esta forma concluir $\rho'(\theta) \in \mathbb{DT}(\Sigma)$. Por lo tanto la conclusión $\rho'(g(x_1, \ldots, x_m) \xrightarrow{b} \theta) = g(x_1, \ldots, x_m) \xrightarrow{a} \rho'(\theta)$ de s tiene la forma que el formato $nx\mu f\theta$ exige.

Verifiquemos las premisas de s. Sea $\rho'(t \xrightarrow{a} \mu)$ una premisa positiva en $\rho'(\mathrm{top}(B', r, \phi'))$. Luego $t \xrightarrow{a} \mu$ es una premisa positiva de una regla en B' la cual no pertenece a la imagen de ϕ'. Entonces μ es tal que $\rho(\mu) = \mu$ y esto implica $\rho'(\mu) = \mu$. Para demostrar que $\rho'(t)$ es una variable existen dos casos para analizar:

- $t \xrightarrow{a} \mu \in \mathrm{top}(B, r, \phi)$. Luego $\sigma(t \xrightarrow{a} \mu) \in H$ y dado que $\frac{H}{\psi}$ está en format $nx\mu f\theta$, $\sigma(t)$ es una variable. Por lo tanto $\sigma\rho'(t) = \sigma(t)$ y entonces $\rho'(t)$ debe ser una variable.

- $t \xrightarrow{a} \mu \notin \text{top}(B, r, \phi)$. Luego hay una regla $b \in B$ tal que $\phi(b) = t \xrightarrow{a} \mu$. Dado que $t \xrightarrow{a} \mu$ no pertenece a la imagen de ϕ' tenemos que $b \notin B'$. Por la construcción de B' y ρ, $\rho^k(t)$ es una variable para todo $k \geq 0$ (ver (II)). Entonces la condición (ρ'III) asegura que $\rho'(t)$ es una variable.

Esto demuestra que las premisas positivas satisfacen los requerimientos del formato $nx\mu f\theta$.

Las restricciones sintácticas para las premisas positivas, las premisas cuantitativas y la conclusión de satisfacen. Además, no existen restricciones para las premisas negativas, por lo tanto s está en formato $nx\mu f\theta$ y $s \in R'$.

Para toda premisa positiva $\psi' \in \sigma(\text{top}(B', r, \phi'))$ la regla $\frac{H}{\psi'}$ está en formato $nx\mu f\theta$ y es demostrable en R por una estructura de prueba menor que (B, r, ϕ). Aplicando hipótesis inductiva estas reglas son demostrables en R'. Aplicando el Lema 6.1.1 en estas reglas y usando s se puede demostrar que $\frac{H}{\psi}$ es demostrable en R'. □

El Lema 6.2.3, el cual establece que una especificación en formato $nx\mu f\theta$ se puede reducir a una especificación en formato pntree, se demuestra de forma similar.

Lema 6.2.3. *Sea $P = (\Sigma, A, R)$ un PTSS en formato $nx\mu f\theta$. Entonces existe un PTSS $P' = (\Sigma, A, R')$ en formato pntree tal que para toda regla cerrada $\frac{H}{\psi}$ con sólo premisas negativas y ψ un literal positivo, $P \vdash \frac{H}{\psi}$ sii $P' \vdash \frac{H}{\psi}$*

Demostración. Sea $P' = (\Sigma, A, R')$ tal que R' es el conjunto de reglas demostrables de P en formato pntree. Por el Lema 6.1.2, la implicación de derecha a izquierda es válida.

Para la demostración de izquierda a derecha procedemos por inducción. Sea $\frac{H}{\psi}$ una regla cerrada demostrable en P donde H sólo contiene literales negativos. Luego $\psi \in H$ por Lema 6.1.1 existe una regla r y una substitución ρ tal que $\rho(\text{conc}(r)) = \psi$ y $P \vdash \frac{H}{\psi'}$ para toda premisa $\psi' \in \rho(\text{pprem}(r))$. Por lo tanto, $P' \vdash \frac{H}{\psi'}$ de forma trivial o por inducción.

Como r está en formato $nx\mu f\theta$, r tiene la siguiente forma

$$\frac{\bigcup_{m\in M}\{w^{\vec{z}}_m \xrightarrow{a_m} \mu^{\vec{z}}_m : \vec{z} \in \mathcal{Z}\} \cup \bigcup_{n\in N}\{t_n(\vec{z}) \xrightarrow{b_n} \not\rightarrow : \vec{z} \in \mathcal{Z}\} \cup \{\theta_l(Y_l) \trianglerighteq_{l,k} p_{l,k} : l \in L, k \in K_l\}}{f(x_1, \ldots, x_{\text{r}(f)}) \xrightarrow{a} \theta}$$

donde cada $w^{\vec{z}}_m$ es una variable en la tupla $\vec{z}$.

Sea G el grafo de dependencia asociado a $\text{pprem}(r) \cup \text{qprem}(r)$. A partir de r, construimos una regla $r' \in S$ de la siguiente forma. Sea $\mu^{\vec{z}}_m$ el destino de una premisa positiva tal que no existe un camino para atrás en G desde un vértice $\mu^{\vec{z}}_m$ a algún vértice x_i, con $i \in \{1, \ldots, x_{\text{r}(f)}\}$. Notemos que, por el requisito de simetría en la Def. 5.2.3, esto sucede para toda variable $\mu^{\vec{z}}_m$ con $\vec{z} \in \mathcal{Z}$. Antes de obtener r', obtengamos una regla r'' de la siguiente forma: (I) se reemplazan las variables $w^{\vec{z}}_m$ y $\mu^{\vec{z}}_m$ por $\rho(w^{\vec{z}}_m)$ y $\rho(\mu^{\vec{z}}_m)$, respectivamente, y (II) se reemplazan las variables libres ζ en θ_l y θ por $\rho(\zeta)$. La regla r'' no tiene variables libres y es una instancia de substitución de r, entonces r'' es demostrable en P. Para obtener r', reemplazamos cada premisa positiva cerrada $\rho(w^{\vec{z}}_m \xrightarrow{a} \mu^{\vec{z}}_m)$ por H. Dado que $w^{\vec{z}}_m \xrightarrow{a} \mu^{\vec{z}}_m$ es una premisa positiva de r, por inducción, $P \vdash \frac{H}{\rho(w^{\vec{z}}_m \xrightarrow{a} \mu^{\vec{z}}_m)}$. Por lo tanto r' también demostrable en P.

Notemos que la regla resultante r' está en formato $nx\mu f\theta$ y no tiene variables libres. Más aún, r' es bien fundada pues en su grafo de dependencia cada cadena para atrás alcanza un vértice x_i. Por lo tanto r' es una regla en formato pntree, luego $r' \in R'$.

Sea $p \in \text{prem}(r')$. Entonces $p \in H$ (y p es cerrado) o $p \in \text{prem}(r)$. Para ambos casos, $P' \vdash \frac{H}{\rho(p)}$ (si $p \in \text{prem}(r)$, vale por inducción). Por lo tanto $\frac{H}{\rho(\text{conc}(r'))} \in R'^{\vdash}$. Dado que $\rho(\text{conc}(r')) = \rho(\text{conc}(r)) = c$, $P' \vdash \frac{H}{c}$. □

Teorema 6.2.1. *Sea $P = (\Sigma, A, R)$ un PTSS en formato $nt\mu f\theta/nt\mu x\theta$. Entonces existe un PTSS $P' = (\Sigma, A, R')$ en formato pntree equivalente a P.*

La prueba del Teorema 6.2.1 es directa aplicando el Corolario 5.3.1 y los Lemas 6.2.2, 6.2.3, y 4.3.2, en ese orden.

Corolario 6.2.1. *Si P es un PTSS en formato $nt\mu f\theta/nt\mu x\theta$, entonces $\sim$ es una congruencia para todo operador en P.*

Siguiendo la notación utilizada en [34], decidimos que una regla está en *formato pntree simple* si la regla está en formato pntree y todas sus premisas negativas son de la forma $x \overset{a}{\not\rightarrow}$. Uno podría preguntarse si una PTSS en formato pntree puede reducirse a un PTSS equivalente en formato pntree simple. La respuesta es negativa. En otras palabras, se puede demostrar que el formato pntree (y por lo tanto el formato $nt\mu f\theta/nt\mu x\theta$) es estrictamente más expresivo que el formato pntree simple. Esto se puede demostrar adaptando el ejemplo introducido en [34] para demostrar el mismo resultado en el contexto no probabilístico. No desarrollaremos el ejemplo por ser éste directo.

6.3. Observaciones finales

En esta sección hemos demostrado que para toda especificación en formato $nt\mu f\theta/nt\mu x\theta$ existe una especificación equivalente en formato pntree (Teorema 6.2.1). Para demostrar este resultado se necesitó introducir las estructuras de pruebas, las cuales permitieron definir una variante prueba para una regla. El corolario de este resultado es que no es necesario que una especificación en formato $nt\mu f\theta/nt\mu x\theta$ sea bien fundada para que la bisimulación sea una congruencia para toda función de la signatura. Por lo tanto, se generaliza el Teorema 5.3.2 a todo PTSS en formato $nt\mu f\theta/nt\mu x\theta$.

Si bien este capítulo es de carácter técnico, el mismo nos permite ver cómo se pueden trasladar los resultados para el formato *ntyft/ntyxt* al formato $nt\mu f\theta/nt\mu x\theta$.

El procedimiento de reducción que se presentó sigue la misma estructura que el trabajo [34], en el cual se demuestra que una especificación en formato *ntyft/ntyxt* puede reducirse a una especificación equivalente en formato *ntree* (formato *ntyft/ntyxt* con reglas puras y sólo variables como fuente de las premisas positivas). Es decir que se realiza la misma reducción en un contexto no probabilístico. La diferencia con respecto a éste es que en el caso probabilístico se debe tener en cuenta las premisas cuantitativas. Sin embargo esto no presentó mayores dificultades debido a que los términos de distribución brindaron el marco para realizar esto de forma directa. Esto no fue casualidad: la versión previa del formato $nt\mu f\theta/nt\mu x\theta$ fue el formato $nt\mu fv/nt\mu xv$ [22]. En éste,

cada premisa cuantitativa sólo podía utilizar una variable de distribución. Con esta restricción es imposible derivar el resultado, esta fue la razón por la que se introducen los términos de distribuciones en [59] y se define el formato actual. Luego, la adaptación de resultados desde el contexto no probabilístico al probabilístico puede generar mejoras en el formato.

Capítulo 7

La mayor congruencia inducida por el formato $nt\mu f\theta/nt\mu x\theta$

En el Capítulo 5 se mencionó que cuando se define un formato, las restricciones impuestas sobre éste tienen que permitir definir operadores con suficiente poder de expresividad, de lo contrario, el formato sería inútil. En este Capítulo se demostrará que el formato $nt\mu f\theta/nt\mu x\theta$ permite definir operadores sumamente poderosos. Tanto es así, que es posible definir operadores que distinguen cuando dos procesos son bisimilares aún usando la semántica más débil para los sistemas de transiciones probabilísticos. En otras palabras se demostrará que la congruencia inducida por contextos del formato $nt\mu f\theta/nt\mu x\theta$, con respecto a la equivalencia de trazas, es exactamente la bisimulación.

Sea $P = \langle \Sigma, A, R \rangle$ un PTSS y sea $\rightarrow_P$ su sistema de transición asociado. Para demostrar el resultado se debe extender la especificación P de forma tal que esta extensión permita verificar si dos términos $t, t' \in T(\Sigma)$ son bisimilares con respecto a $\rightarrow_P$. Notar que esta extensión debe preservar los comportamientos de t y t', de lo contrario ya no se estaría verificando el PTS $\rightarrow_P$. Para asegurar que los comportamientos se preservan se introducen resultados que garantizan la *extensión conservativa* de especificaciones, i.e. los resultados garantizan que una especificación puede extenderse combinándola con otra sin modificar el comportamiento de los términos que la primera define.

La Sección 7.1 presenta los resultados que garantizan la *extensión conservativa* de especificaciones en formato $nt\mu f\theta/nt\mu x\theta$. La Sección 7.2 presenta los resultados asociados la congruencia inducida por contextos del formato $nt\mu f\theta/nt\mu x\theta$.

7.1. Especificaciones modulares

A veces, uno desea extender un lenguaje con nuevas operación y comportamientos. Una forma de hacer esto es agregar nuevos operadores y reglas a la especificación original. En otros palabras, dados dos PTSS P^0 y P^1, uno desea combinarlos para obtener una nueva especificación

$P^0 \oplus P^1$. En general asumiremos que P^0 es la especificación original y que P^1 es la extensión. La única restricción que se impone para combinar especificaciones es que si un nombre de función es común a ambas especificaciones entonces la aridad de la función debe ser la misma. Utilizaremos el término *suma* para referirnos a la combinación de dos especificaciones. Para definir la *suma* de dos PTSS se necesita definir la *suma* de dos signaturas.

Definición 7.1.1. *Sea* $\Sigma^0 = (F^0, \mathsf{r}^0)$ *y* $\Sigma^1 = (F^1, \mathsf{r}^1)$ *dos signaturas tales que* $f \in F^0 \cap F^1 \Rightarrow \mathsf{r}^0(f) = \mathsf{r}^1(f)$. *La* suma *de* Σ^0 *y* Σ^1*, notación* $\Sigma^0 \oplus \Sigma^1$*, es una nueva signatura* $(F^0 \cup F^1, \mathsf{r})$ *donde*

$$\mathsf{r}(f) = \begin{cases} \mathsf{r}^0(f) & \text{si } f \in F^0 \\ \mathsf{r}^1(f) & \text{caso contrario} \end{cases}$$

Definición 7.1.2. *Dado dos PTSS* $P^0 = (\Sigma^0, A^0, R^0)$ *y* $P^1 = (\Sigma^1, A^1, R^1)$ *tales que* $\Sigma = \Sigma^0 \oplus \Sigma^1$ *está definido, la* suma *de* P^0 *y* P^1*, notación* $P^0 \oplus P^1$*, es el PTSS* $P^0 \oplus P^1 = (\Sigma^0 \oplus \Sigma^1, A^0 \cup A^1, R^0 \cup R^1)$.

Una propiedad deseada es que la extensión no altere el comportamiento de los términos de la especificación original. Es decir que uno espera que todo término $t \in T(\Sigma^0)$ definido en la especificación original preserve el mismo comportamiento en $P^0 \oplus P^1$. En este contexto "preservar el mismo comportamiento" se interpreta como realizar el mismo conjunto de transiciones. Si se satisface esta propiedad diremos entonces que $P^0 \oplus P^1$ es una *extensión conservativa* de P^0. Notemos que en el caso de los sistemas de transiciones basados en pruebas bien soportadas puede darse el siguiente caso: $t \overset{a}{\not\rightarrow} \notin \rightarrow_{P^0}$ y $t \overset{a}{\not\rightarrow} \in \rightarrow_{P^0 \oplus P^1}$ para $t \in T(\Sigma^0)$ y $a \in A^1 - A^0$. No interpretaremos esto como una extensión que modifica el comportamiento del término t. La definición formal de extensión conservativa se basa en la presentada en [2].

Definición 7.1.3. *Diremos que* $P^0 \oplus P^1$ *es una* extensión conservativa *de* P^0*, o que* P^1 puede ser sumado conservativamente *a* P^0 *si para todo* $t \in T(\Sigma^0)$*,* $a \in A^0 \cup A^1$*,* $\pi \in \Delta(T(\Sigma^0 \cup \Sigma^1))$ *y un conjunto N de premisas negativas cerradas*

$$P^0 \oplus P^1 \vdash \frac{N}{t \overset{a}{\longrightarrow} \pi} \Rightarrow P^0 \vdash \frac{N}{t \overset{a}{\longrightarrow} \pi}$$

Notemos que la definición de extensión conservativa no utiliza el concepto de prueba bien soportada, pues los literales que se pueden derivar dependen directamente de las reglas $\frac{N}{\psi}$ que se pueden derivar utilizando las pruebas básicas. En algunas definiciones de extensión conservativa se reemplaza el $\Rightarrow$ por $\Leftrightarrow$, notemos que la implicación de derecha a izquierda es directa, luego esta definición es análoga pero más simple.

El Teorema 7.1.1 da condiciones suficientes para asegurar cuando un PTSS puede ser sumado conservativamente otro. El enunciado del mismo está basado en el Teorema 4.8 de [24].

Teorema 7.1.1. *Sean* $P^0 = (\Sigma^0, A^0, R^0)$ *y* $P^1 = (\Sigma^1, A^1, R^1)$ *dos PTSS en formato ntμfθ/ntμxθ tales que* $conc(r) = t \overset{a}{\longrightarrow} \theta$ *y* $t \notin \mathbb{T}(\Sigma_0)$ *para todo* $r \in R^1$ *. Entonces* $P^0 \oplus P^1$ *es una extensión conservativa de* P_0.

Demostración. Dado los resultados del Capítulo 6 podemos suponer que la especificación P_0 está en formato pntree. Esto implica que la especificación es bien fundada, por lo cual es posible realizar inducción en el grafo de dependencia de las premisas de las reglas de P_0.

Sean H y ψ tales que $src(\psi) \in T(\Sigma_0)$ y $P^0 \oplus P^1 \vdash \frac{H}{\psi}$ con prueba $\mathfrak{p}(\frac{H}{\psi})$. Demostremos por inducción completa en $n = h(\mathfrak{p}(\frac{H}{\psi}))$ que $P^0 \vdash \frac{H}{\psi}$. Supongamos que la hipótesis vale para $n' < n$ y demostremos el caso n. La prueba $\mathfrak{p}(\frac{H}{\psi})$ implica que existe una regla r y una substitución σ tal que $\sigma(\text{conc}(r)) = \psi$ y para todo $\psi' \in \sigma(\text{pprem}(r))$, $P^0 \oplus P^1 \vdash \frac{H}{\psi}'$ con una prueba $\mathfrak{p}(\frac{H}{\psi}')$. Dado que $src(\psi) \in T(\Sigma_0)$ y que $src(\text{conc}(r')) \notin \mathbb{T}(\Sigma_0)$ para todo $r' \in R_1$ se tiene que $r \in R^0$. Demostremos que todas las variables que aparecen en las premisas positivas son instanciadas por términos en $T(\Sigma_0) \cup \Delta(T(\Sigma_0))$, esto permitirá usar la hipótesis inductiva. Procedamos por inducción en m para $m = n_{\text{VDG}}(x)$ donde n_{VDG} está definido con respecto al grafo de dependencia de las premisas no negativas de r. Si $n_{\text{VDG}}(x) = 1$, entonces una de las posibilidades es que x sea una de las variables en la fuente de la conclusión, i.e. $x \in \{x_1, \ldots, x_{\mathrm{r}(f)}\}$. Dado que $src(\psi) \in T(\Sigma_0)$ esto implica que $\sigma(x) \in T(\Sigma_0)$. Por otro lado podría darse el caso $x \in Y$ donde $\pi(Y) \trianglerighteq q$ es una premisa de la regla con $\pi \in \Delta(T(\Sigma_0))$. Dado que σ es una substitución genuina, $\sigma(y) \in \mathsf{support}(\pi)$ para todo $y \in Y$, por lo tanto $\sigma(y) \in T(\Sigma_0)$. Supongamos que para todo ζ tal que $n_{\text{VDG}}(\zeta) < m$, $\sigma(\zeta) \in T(\Sigma_0) \cup T(\Sigma_0)$. Sea $t \xrightarrow{a} \mu$ una premisa positiva de r tal que $n_{\text{VDG}}(\mu) = m$. Por definición de n_{VDG} para todo $x \in Var(t)$, $n_{\text{VDG}}(x) < m$ y por inducción $\sigma(t) \in T(\Sigma_0)$. Por la primera hipótesis inductiva, $P^0 \vdash H/\sigma(t \xrightarrow{a} \mu)$ y por consiguiente $\sigma(\mu) \in \Delta(T(\Sigma_0))$. El caso $\zeta \in Y$ con $\theta(Y) \trianglerighteq q$ es similar al caso base dado que $\sigma(\theta) \in \Delta(T(\Sigma_0))$. Luego todas las premisas positivas son demostrables; las premisas cuantitativas válidas, por ser las mismas que en la prueba original; las negativas, por definición, se encuentran en H. Por lo tanto $P^0 \vdash \frac{H}{\psi}$. □

7.2. Test de bisimulación

El formato $nt\mu f\theta/nt\mu x\theta$ permite una gran variedad de tests en las hipótesis de las reglas para la construcción estructural de las transiciones. Estos tests incluyen la imposibilidad de la realización de una transición (premisas negativas), verificación de valores de probabilidad que permiten un alto grado de distinción de las distribuciones (premisas cuantitativas), identificación de ejecuciones futuras (lookahed), aparte de las usuales premisas positivas. Estos tipos de test en conjunto permiten una identificación bastante precisa de la semántica de los distintos términos, y por lo tanto, es razonable que el formato permita distinguir unos términos de otros simplemente inspeccionando qué transiciones pueden o no hacer éstos. De hecho, en esto se basa la construcción estructural de la semántica y tiene un impacto directo en la interpretación de un contexto $C[\_]$ según se instancie con distintos términos. Por ejemplo, el contexto $\mathsf{U}(\_)$ sólo podrá realizar una transición etiquetada con a si es instanciado con un término que posea una ejecución probabilista que *no* pueda realizar transiciones etiquetadas con a con probabilidad 1 (según la semántica en el Ejemplo 4.2.1). De esta manera, el contexto $\mathsf{U}(\_)$ puede "distinguir" entre términos cuyas ejecuciones siempre puedan realizar una acción a con cierta probabilidad y términos que posean una ejecución que fallen en ejecutar una acción a con probabilidad 1. La palabra "distinguir" se ha utilizado aquí con el siguiente sentido: un contexto $C[\_]$ puede distinguir entre dos términos t y t', si $C[t]$ puede realizar algo *evidentemente* distinto de $C[t']$. Dentro de semántica de procesos, lo *más* evidente es el comportamiento de trazas.

En este sentido surge la pregunta de cuál es la "precisión" alcanzable en la distinción de términos a través de un contexto cualquiera y mediante la simple observación de trazas, cuando

la semántica se deriva de un PTSS en formato *ntμfθ/ntμxθ*.

En un sentido más técnico, dado un PTSS $P_0 = \langle \Sigma_0, A_0, R_0 \rangle$ en formato *ntμfθ/ntμxθ*, queremos saber cuál es la relación de equivalencia entre términos $t, t' \in T(\Sigma_0)$ cualesquiera tal que $C[t]$ y $C[t']$ son indistinguibles a través de las trazas (i.e., sus semánticas definen exactamente el mismo conjunto de trazas) para todo contexto $C[\_]$ de $\Sigma_0 \oplus \Sigma_1$ en cualquier extensión conservativa $P_0 \oplus P_1$ posible con $P_1 = \langle \Sigma_1, A_1, R_1 \rangle$ también en formato *ntμfθ/ntμxθ*.

Es claro que el formato no puede distinguir más allá de la bisimulación dado que el Teorema 5.3.2 garantiza que si $t \sim t'$ luego $C[t] \sim C[t']$ (y en consecuencia coinciden en sus conjunto de trazas) bajo la semántica inducida por cualquier PTSS en formato *ntμfθ/ntμxθ*. En el resto de esta sección demostraremos que la equivalencia inducida por el formato *ntμfθ/ntμxθ* es precisamente la bisimulación. Es decir, si dos términos t y t' no son bisimilares en un PTSS en formato *ntμfθ/ntμxθ*, existe una extensión conservativa de éste, también en formato *ntμfθ/ntμxθ*, y un contexto $C[\_]$ en dicha extensión tal que los conjuntos de trazas de $C[t]$ y $C[t']$ son distintos.

Definición 7.2.1. *Sea $P = (\Sigma, A, R)$ y sea $\rightarrow_P$ su sistema de transición asociado. Sea $t \in T(\Sigma)$, una secuencia $a_1 \dots a_n \in A^*$ es una traza de t sii existen términos $t_0, \dots, t_n \in T(\Sigma)$ y distribuciones $\pi_1, \dots, \pi_n$ tales que $t_0 = t$, $t_i \xrightarrow{a_{i+1}} \pi_{i+1}$ y $\pi_{i+1}(t_{i+1}) > 0$ para $0 \leq i < n$. Sea $\mathsf{Tr}(t)$ un conjunto de de todas las trazas t. Dos términos $t, t' \in T(\Sigma)$ son* equivalentes en trazas respecto a *P, notación $t \equiv_P^T t'$, sii $\mathsf{Tr}(t) = \mathsf{Tr}(t')$.*

Definición 7.2.2. *Sea $P = \langle \Sigma, A, R \rangle$ un PTSS completo en formato ntμfθ/ntμxθ. Dos términos $t, t' \in T(\Sigma)$ son* congruentes en trazas con respecto a *ntμfθ/ntμxθ, notación $t \equiv^T_{ntμfθ/ntμxθ} t'$, sii para todo PTSS $P' = (\Sigma', A', R')$ en formato ntμfθ/ntμxθ el cual puede sumarse conservativamente a P, y para todo contexto $C[x]$ de $\Sigma \oplus \Sigma'$, $C[t] \equiv^T_{P \oplus P'} C[t']$.*

Concretamente, demostraremos que la congruencia de trazas respecto a *ntμfθ/ntμxθ* coincide con la bisimulación bajo la hipótesis de que el sistema de transiciones probabilistas inducido por el PTSS P es de *imagen finita* ($\rightarrow_P$ tiene imagen finita sii para todo $t \in T(\Sigma)$ y $a \in A$, el conjunto $\{\mu \mid t \xrightarrow{a}_P \mu\}$ es finito). Es decir, $t \equiv^T_{ntμfθ/ntμxθ} t'$ si y sólo si $t \sim t'$ para todo par de término t y t' siempre que sus semánticas sean de imagen finita.

La demostración utiliza una caracterización inductiva de la bisimulación [8]. Tal caracterización (Def. 7.2.3), que denominaremos *bisimulación finita*, coincide con la bisimulación para los sistemas de transiciones probabilistas de imagen finita. El resto de la prueba se basa en la existencia de un *tester de bisimulación* (Def. 7.2.4). Esto es, un PTSS P_T que puede sumarse conservativamente a otro PTSS P y que se demuestra que contiene el conjunto de operadores y reglas suficientes para definir contextos que permitan distinguir dos términos que no son (finitamente) bisimilares (Lema 7.2.1).

Definición 7.2.3. *Sea $P = (\Sigma, A, R)$ y $\rightarrow_P$ su sistema de transición asociado. La relación $\simeq_P^n \subseteq T(\Sigma) \times T(\Sigma)$ para $n \in \mathbb{N}$ está definida inductivamente por:*

$$\begin{aligned} \simeq_P^0 &= T(\Sigma) \times T(\Sigma) \\ \simeq_P^{n+1} &= \{(t, t') \mid (t \xrightarrow{a} \pi \Rightarrow \exists \pi' : t' \xrightarrow{a} \pi' \wedge \pi \simeq_P^n \pi') \wedge (t' \xrightarrow{a} \pi' \Rightarrow \exists \pi : t \xrightarrow{a} \pi \wedge \pi \simeq_P^n \pi')\} \end{aligned}$$

donde $\pi \simeq_P^n \pi'$ si y sólo si $\forall Q \subseteq T(\Sigma) : \simeq_P^n$-cerrado$(Q) \Rightarrow \pi(Q) = \pi'(Q)$. Dado dos términos $t, t' \in T(\Sigma)$, t y t' son n-bisimilares *si $t \simeq_P^n t'$. Decimos que t y t' son* finitamente bisimilares*, notación $t \simeq_P t'$, si $t \simeq_P^n t'$ para todo $n \in \mathbb{N}$.*

La bisimulación finita y la bisimulación coinciden para procesos con imagen finita [8, Lemma 3.4.8]. Es decir, si $\rightarrow_P$ tiene imagen finita entonces $\sim = \simeq_P$.

El tester de bisimulación P_T introduce dos familias de funciones: las funciones binarias B_n y las funciones Pr_n^k con aridad $(k+1)$ $(n, k \in \mathbb{N})$. Además se introduce una constante $\perp$. Las funciones $\mathsf{B}_n(t,u)$ son utilizadas para verificar si dos términos t y u son n-bisimilares. Si este es el caso, existirá una transición $\mathsf{B}_n(t,u) \xrightarrow{yes} \delta_\perp$; en el caso contrario, $\mathsf{B}_n(t,u) \xrightarrow{no} \delta_\perp$. De esta forma dos términos t y u que no son bisimilares serán distinguidos por el contexto $\mathsf{B}_n(t, \_)$ para algún n. Por otro lado, Pr_n^k es utilizado como un operador auxiliar para evaluar la probabilidad de alcanzar k (no necesariamente diferentes) clases de equivalencias de la $(n-1)$-bisimulación. Más precisamente, $\mathsf{Pr}_n^k(t, u_1, \cdots, u_k) \xrightarrow{(a,q_1,\ldots,q_k)} \delta_\perp$ si existe una transición $t \xrightarrow{a} \pi$ tal que $\pi([u_1]_{\simeq^{n-1}}) \geq q_1, \ldots, \pi([u_k]_{\simeq^{n-1}}) \geq q_k$, donde $q_1, \ldots, q_k$ son números racionales.

En la definición del tester de bisimulación, $\mathsf{B}_{n-1}(z, Z) \xrightarrow{yes} \mu$ con $Z \subseteq \mathcal{V}$ denota el conjunto de premisas $\{\mathsf{B}_{n-1}(z, z') \xrightarrow{yes} \mu \mid z' \in Z\}$.

Definición 7.2.4. *Sea $P = (\Sigma, A, R)$ un PTSS. El* bisimulation tester de *P es un PTSS $P_T = (\Sigma_T, A_T, R_T)$ donde Σ_T contiene las funciones binarias B_n y las funciones Pr_n^k con aridad $k+1$, $n \in \mathbb{N}$ y una constante $\perp$, $A_T = A \cup (\bigcup_{i>0}(A \times \mathbb{Q}^i)) \cup \{yes, no\}$, y R contiene las siguientes reglas (para todo $n, k > 0$, $a \in A$ y $q_i \in \mathbb{Q}$ con $i \in \mathbb{N}$):*

$$(1)\ \mathsf{B}_0(x,y) \xrightarrow{yes} \delta_\perp \qquad \frac{\mathsf{Pr}_n^k(x, z_1, \ldots, z_k) \xrightarrow{(a,q_1,\ldots,q_k)} \mu \quad \mathsf{Pr}_n^k(y, z_1, \ldots, z_k) \xrightarrow{(a,q_1,\ldots,q_k)} \not\rightarrow}{\mathsf{B}_n(x,y) \xrightarrow{no} \delta_\perp} \quad (3)$$

$$(2)\ \frac{x \xrightarrow{a} \mu \quad \{\mathsf{B}_{n-1}(z_i, Z_i) \xrightarrow{yes} \mu_i,\ \mu(Z_i) \geq q_i\}_{i=1}^k}{\mathsf{Pr}_n^k(x, z_1, \ldots, z_k) \xrightarrow{(a,q_1,\ldots,q_k)} \delta_\perp} \qquad \frac{\mathsf{B}_n(x,y) \xrightarrow{no} \not\rightarrow \quad \mathsf{B}_n(y,x) \xrightarrow{no} \not\rightarrow}{\mathsf{B}_n(x,y) \xrightarrow{yes} \delta_\perp} \quad (4)$$

La idea detrás la función Pr_n^k explicada con anterioridad puede verse reflejada en la regla (2). Notemos además que la distinción entre dos términos que no son n-bisimilares se realiza en la regla (3). Aquí la premisa negativa indica que no es posible encontrar una acción para y la cual alcance las clases de equivalencia $[z_i]_{\simeq^{n-1}}$ (en realidad las clases de equivalencia del término en que se instancia z_i) con una probabilidad mayor que q_i para cada caso. Mientras que la positiva premisa indica que esto si es posible para x.

Observemos que P_T está en formato $nt\mu f\theta$ pero *no* es puro. Si bien esto no es necesario, resulta conveniente: la regla (3), la cual no es pura debido a la presencia de las variables $z_1, \ldots z_k$, permite la instanciación de términos arbitrarios lo cual incluye los términos que pertenecen a las clases de equivalencia de la bisimulación finita $(n-1)$. Esto es esencial para definir la bisimulación finita n (dado que dos términos están relacionados por una bisimulación n si alcanzan con la misma probabilidad las clases de equivalencia de la bisimulación $(n-1)$). De cualquier forma, el hecho de que P_T no sea puro no es un problema dado que la extensión conservativa está garantizada por el Teorema 7.1.1.

El siguiente lema es central para la demostración del Teorema 7.2.1. El mismo establece que el tester de bisimulación es correcto. Es decir que $\mathsf{B}_n(t,t') \xrightarrow{yes} \delta_\perp \in \rightarrow_{P \oplus P_T} \Leftrightarrow t \simeq_P^n t'$, para todo $t, t' \in T(\Sigma)$.

Lema 7.2.1. *Sea $P = (\Sigma, A, R)$ un PTSS completo en formato ntμfθ/ntμxθ. Además, $\rightarrow_P$ tiene imagen finita,yes, no $\notin A$ y Σ no contiene los nombres de función B_n y Pr^k_n para todo $n, k \in \mathbb{N}$. Entonces $\mathsf{B}_n(t,t') \xrightarrow{yes} \delta_\perp \in \rightarrow_{P\oplus P_\mathcal{T}} \Leftrightarrow t \simeq^n_P t'$, para todo $t, t' \in T(\Sigma)$.*

Demostración. Notemos primero que la operación B_n es conmutativa, lo cual asegura simetría. Además, la relación $\simeq^n_P$ también es simétrica por construcción. Procedemos por inducción en n.

Sea $n = 0$, entonces $P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{B}_0(t,t') \xrightarrow{yes} \delta_\perp$ y $t \simeq^0_P t'$ para todo $t, t' \in T(\Sigma)$. Para el caso inductivo supongamos $\mathsf{B}_n(t,t') \xrightarrow{yes} \delta_\perp \in \rightarrow_{P\oplus P_\mathcal{T}} \Leftrightarrow t \simeq^n_P t'$ para todo $t, t' \in T(\Sigma)$ y demostremos $\mathsf{B}_{n+1}(t,t') \xrightarrow{yes} \delta_\perp \in \rightarrow_{P\oplus P_\mathcal{T}} \Leftrightarrow t \simeq^{n+1}_P t'$. Primero la implicación de izquierda a derecha. Debemos demostrar que si $P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{B}_{n+1}(t,t') \xrightarrow{yes} \delta_\perp$ y $P \vdash_{ws} t \xrightarrow{a} \pi$, entonces existe π' tal que $P \vdash_{ws} t' \xrightarrow{a} \pi'$ y $\pi \simeq^n_P \pi'$. Dado que $P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{B}_{n+1}(t,t') \xrightarrow{yes} \delta_\perp$ entonces por la regla (4) del tester de bisimulación

$$P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{B}_{n+1}(t,t') \xrightarrow{no} \qquad\qquad P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{B}_{n+1}(t',t) \xrightarrow{no}$$

Lo que implica que para todo substitución ρ tal que $\rho(x) = t$ y $\rho(y) = t'$ no existe regla del tester de bisimulación de la forma (3) tal que las premisas se satisfagan. Esto implica, junto a las reglas de la forma (2), que $t' \xrightarrow{a} \pi'$ para algún π'.

Procedamos por el absurdo. Supongamos que para π' tal que $t' \xrightarrow{a} \pi'$ no se satisface $\pi \simeq^{n+1}_P \pi'$. Dado que el sistema de transición asociado P tiene imagen finita, podemos suponer t' puede realizar exactamente $K \in \mathbb{N}$ transiciones con la etiquetada a. Entonces para cada transición $P \vdash_{ws} t' \xrightarrow{a} \pi_i$ con $i \in \{1, \ldots, K\}$, sea $Q_i \in A(\simeq^n_P)$, i.e. Q_i es un átomo del álgebra de Bool $\langle \{X : \simeq^n_P\text{-cerrado}(X)\}, \subseteq \rangle$, tal que $\pi(Q_i) > \pi_i(Q_i)$. Notemos que Q_i debe existir, sino $\pi \simeq^{n+1}_P \pi'$ Además, sea $t_i \in Q_i$ y $q_i \in \mathbb{Q}$ tal que $\pi(Q_i) \geq q_i > \pi_i(Q_i)$ para $i \in \{1, \ldots, K\}$ (la existencia de q_i está garantizada porque $\mathbb{Q}$ es denso para $\mathbb{R}$).

Como $P \vdash_{ws} t \xrightarrow{a} \pi$, por extensión conservativa, $P \oplus P_\mathcal{T} \vdash_{ws} t \xrightarrow{a} \pi$. Por hipótesis inductiva, si $t'_i \in Q_i$ entonces $P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{B}_n(t_i, t'_i) \xrightarrow{yes} \delta_\perp$. Además, sea ρ una substitución tal que $\rho(x) = t$, $\rho(\mu) = \pi, \rho(z_i) = t_i$ y $\rho(Z_i) = Q_i$ para $i \in \{1, \ldots, K\}$. Entonces, con las pruebas de $P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{B}_n(t_i, t'_i) \xrightarrow{yes} \delta_\perp$, $P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{B}_n(t_i, t'_i) \xrightarrow{yes} \delta_\perp$, la sustitución ρ y la siguiente regla

$$r = \frac{x \xrightarrow{a} \mu \quad \{\mathsf{B}_n(z_i, Z_i) \xrightarrow{yes} \mu_i,\ \mu(Z_i) \geq q_i\}_{i=1}^K}{\mathsf{Pr}^K_{n+1}(x, z_1, \ldots, z_K) \xrightarrow{(a,q_1,\ldots,q_K)} \delta_\perp}$$

se puede construir una prueba de $P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{Pr}^k_{n+1}(t, t_1, \ldots, t_K) \xrightarrow{(a,q_1,\ldots,q_K)} \delta_\perp$.

Esto implica que $P \oplus P_\mathcal{T} \vdash_{ws} \mathsf{Pr}^k_{n+1}(t', t_1, \ldots, t_K) \xrightarrow{(a,q_1,\ldots,q_K)} \delta_\perp$, caso contrario una regla del tipo (3) se hubiese podido aplicar. Además, por extensión conservativa, $P \oplus P_\mathcal{T} \vdash_{ws} t' \xrightarrow{a} \pi_i$ con $i \in \{1, \ldots, K\}$ son las únicas transiciones que puede ejecutar t' con la acción a. Luego debe existir ρ' tal que $\rho'(x) = t'$, $\rho(\mu) \in \{\pi_k \mid 1 \leq k \leq K\}, \rho'(z_i) = t_i, \rho'(Z_i) \subseteq Q_i$ y $\rho'(\mu_i) = \delta_\perp$, para todo $1 \leq i \leq K$, tal que las premisas de la regla prem$(\rho'(r))$ son válidas. Por lo tanto, $\rho'(\mu(Z_i)) \geq q_i$ también tiene que ser válida para todo $1 \leq i \leq K$. Pero para todo $1 \leq i \leq K$ tal

que $\rho'(\mu) = \pi_i$, $q_i > \pi_i(Q_i) \geq \rho'(\mu(Z_i)) \geq q_i$ lo que es una contradicción. Por lo tanto debe existir $\pi' \in \Delta(\Sigma)$ tal que $t' \xrightarrow{a} \pi'$ y $\pi \simeq_P^{n+1} \pi'$.

Demostremos la implicación de derecha a izquierda. Supongamos la hipótesis para n y demostremos el caso $n+1$. Si $t \simeq_P^{n+1} t'$, debemos demostrar que $P \oplus P_T \vdash_{ws} \mathsf{B}_{n+1}(t,t') \xrightarrow{yes} \delta_\perp$. Por la regla 4 del tester debemos probar que los literales $\mathsf{B}_n(t,t') \xrightarrow{no}\!\!\!\!\!\!/\;\;$ y $\mathsf{B}_n(t',t) \xrightarrow{no}\!\!\!\!\!\!/\;\;$ son demostrables mediante una prueba bien soportada. Demostremos sólo $P \oplus P_T \vdash_{ws} \mathsf{B}_n(t,t') \xrightarrow{no}\!\!\!\!\!\!/\;\;$, el otro caso es análogo.

Se debe demostrar que para todo prueba de $P \oplus P_T \vdash \frac{H}{\mathsf{B}_n(t,t') \xrightarrow{no} \pi_n}$ con $\pi_n \in \Delta(\Sigma)$ existe $h \in H$ tal que $P \oplus P_T \vdash_{ws} \neg h$. Las reglas en P_T implican que la prueba debe tener la siguiente forma:

$$\dfrac{\dfrac{\dfrac{H_0}{\vdots}}{t \xrightarrow{a} \pi} \qquad \dfrac{\dfrac{H_i}{\vdots}}{\{\mathsf{B}_n(t_i,T_i) \xrightarrow{yes} \pi_i\}_{i=1}^k} \quad \{\pi(T_i) \geq q_i\}_{i=1}^k}{\mathsf{Pr}_{n+1}^k(t,t_1,\ldots,t_k) \xrightarrow{(a,q_1,\ldots,q_k)} \delta_\perp} \qquad \mathsf{Pr}_{n+1}^k(t',t_1,\ldots,t_k) \xrightarrow{(a,q_1,\ldots,q_k)}\!\!\!\!\!\!/\;\;}{\mathsf{B}_{n+1}(t,t') \xrightarrow{no} \delta_\perp}$$

P es completo, luego si $P \vdash_{ws} t \xrightarrow{a}\!\!\!\!\!\!/\;\;$ entonces $P \vdash_{ws} \neg h_0$ con $h_0 \in H_0$ y por lo tanto $P \oplus P_T \vdash_{ws} t \xrightarrow{a}\!\!\!\!\!\!/\;\;$. Luego, supongamos $P \vdash_{ws} t \xrightarrow{a} \pi$ y H_0 tal que no existe $h_0 \in H_0$ tal que $P \vdash_{ws} \neg h_0$. Supongamos además que para todo H_i con $i \in \{1,\ldots,k\}$ no existe $h_i \in H_i$ tal que $P \oplus P_T \vdash_{ws} \neg h_i$. Demostremos entonces que se puede derivar $P \oplus P_T \vdash_{ws} \mathsf{Pr}_n^k(t',t_1,\ldots,t_k) \xrightarrow{(a,q_1,\ldots,q_k)} \delta_\perp$.

Dado que $t \simeq_P^{n+1} t'$, existe π' tal que $P \vdash_{ws} t' \xrightarrow{a} \pi'$ y $\pi \simeq_P^n \pi'$. Entonces $P \oplus P_T \vdash_{ws} t' \xrightarrow{a} \pi'$ y por lo tanto existe H_0' tal que $P \oplus P_T \vdash H_0'/t' \xrightarrow{a} \pi'$ y no existe $h_0' \in H_0'$ tal que $P \vdash_{ws} \neg h_0'$. Notemos ahora que para todo $t_i' \in T_i$, por hipótesis inductiva, $t_i \simeq_P^n t_i'$ y dado que $\pi \simeq_P^n \pi'$, esto garantiza que los literales $\{\pi'(T_i) \geq q_i\}_{i=1}^k$ son válidos. Todo esto implica que existe una prueba de

$$P \oplus P_T \vdash \frac{H_0' \cup H_1 \cup \ldots \cup H_k}{\mathsf{Pr}_n^k(t,t_1,\ldots,t_k) \xrightarrow{(a,q_1,\ldots,q_k)} \delta_\perp}$$

lo cual concluye la prueba. □

Finalmente el resultado que se desea probar es directo del Teorema 5.3.2, Lema 7.2.1 y [8, Lema 3.4.8].

Teorema 7.2.1. *Sea $P = (\Sigma, A, R)$ un PTSS puro en formato ntμfθ/ntμxθ tal que la signatura Σ contiene al menos una constante. Además, $\rightarrow_P$ tiene imagen finita, yes, no $\notin A$ y Σ no contiene los nombres de funciones B_n y Pr_n^k para todo $n,k \in \mathbb{N}$. Entonces para todo $t,t' \in T(\Sigma)$*

$$t \equiv^T_{ntμfv/ntμxv} t' \Leftrightarrow t \simeq_P t' \Leftrightarrow t \sim_P t'$$

Demostración. Supongamos que $t \sim_P t'$. Sea $P' = \langle \Sigma', A', R' \rangle$ un PTSS en formato *ntμfθ/ntμxθ* tal que $P \oplus P'$ es una extensión conservativa de P. Entonces $t \sim_{P \oplus P'} t'$. Por el Teorema 5.3.2,

para todo contexto $C[x]$ de $P \oplus P'$, $C[t] \sim_{P\oplus P'} C[t']$. Por lo tanto $t \equiv^{T}_{ntμfθ/ntμxθ} t'$. De esta forma demostramos que $t \sim_P t' \Rightarrow t \equiv^{T}_{ntμfθ/ntμxθ} t'$.

Supongamos ahora que $t \not\sim_P t'$. Esto implica que existe un $n \in \mathbb{N}$ tal que $t \not\sim^{n}_{P} t'$. Entonces, por el Lema 7.2.1, $P \oplus P_{\mathcal{T}} \vdash_{ws} B^n(t,t) \xrightarrow{yes} \delta_{\perp}$ mientras que $P \oplus P_{\mathcal{T}} \vdash_{ws} B^n(t,t') \not\xrightarrow{yes}$. Por lo tanto el contexto $B^n(t, \_)$ permite diferenciar t de t' y en consecuencia $t \not\equiv^{T}_{ntμfθ/ntμxθ} t'$. Por contrarecíproca queda demostrado que $t \equiv^{T}_{ntμfθ/ntμxθ} t' \Rightarrow t \simeq_P t'$.

El hecho de que $t \simeq_P t' \Rightarrow t \sim t'$ está demostrado en [8, Lema 3.4.8]. □

7.3. Observaciones finales

En este capítulo hemos demostrado que la congruencia inducida por contextos del formato *ntμfθ/ntμxθ*, con respecto a la equivalencia de trazas, es exactamente la bisimulación. De modo auxiliar a este resultado definimos el concepto de extensión conservativa y presentamos algunos resultados relacionados a éste. De todas maneras, los resultados de extensión conservativa tienen valor por sí mismos dado que brindan condiciones suficientes para extender un lenguaje sin alterar la semántica del sub-lenguaje definido originalmente.

Dado los tipos de premisas que permite utilizar el formato *ntμfθ/ntμxθ*, el hecho de que la bisimulación es la menor congruencia para este formato no es un resultado totalmente inesperado. Para entender mejor esta afirmación podemos realizar una analogía entre estos tipos y los tipos de observabilidad presentados en el Capítulo 2. De esta forma, las premisas positivas podrían interpretarse como los tipo $\{A^O, \rightleftarrows\}$, los cuales permiten registrar que acciones puede ejecutar un proceso. Por otro lado, las premisas negativas cumplen una función similar al tipo $\neg$, que es indicar que ejecuciones no puede realizar un proceso. Notar que en este caso, las premisas negativas sólo permiten codificar qué acciones no puede ejecutar un término, lo cual es mucho menos que una ejecución completa. Aquí es fundamental toda la maquinaria provista por las reglas del tester de bisimulación. Éste permite codificar la existencia de una ejecución no común entre los términos que se comparan. Además, para ambos casos, las premisas cuantitativas junto al lookahead son necesarias para medir las distintas probabilidades.

Un resultado similar es imposible para los formatos RTSS [53], PGSOS [10], y Segala-GSOS [10] dado que estos no poseen forma de cuantificar las probabilidades. Es más, sólo una variante del formato RTSS posee lookahead.

Si comparamos el resultado con el resultado análogo para el formato *ntyft/ntyxt* [43] podemos ver que la estrategia de prueba se mantiene aunque la parte técnica no se preserva debido a las probabilidades y al uso de las pruebas bien soportadas para derivar los sistemas de transiciones.

Capítulo 8

Conclusión

A continuación se realiza un resumen de los resultados presentados en esta tesis y las posibles continuaciones para éstos.

8.1. Logros

Las contribuciones de esta tesis pueden enmarcarse en tres distintas categorías dentro del estudio de los sistemas concurrentes:

1. *Semánticas para modelos interactivos.* Utilizando autómatas de interfaces para modelar sistemas interactivos se realizaron los siguientes aportes:
   a) Se establecieron las suposiciones que rigen al modelo.
   b) Se definió un espectro de semánticas lineales y semánticas ramificadas (*branching semantics*) siguiendo un enfoque basado en nociones de observabilidad.
2. *No interferencia en sistemas interactivos.* Se definieron las *interfaces para seguridad* y la propiedades V-SNNI/V-NNI, donde V es una noción de observabilidad. En este contexto se realizaron los siguientes aportes:
   a) Se estudió cómo se relacionan las variantes V-SNNI y V-NNI.
   b) Se demostró que las propiedades V-SNNI y V-NNI no son preservadas por la composición para toda noción de observabilidad V.
   c) Se brindaron condiciones suficientes para garantizar que las propiedades BSNNI, BNNI, RSNNI y RNNI son preservadas por la composición
   d) Se presentó un algoritmo para decidir si un sistema satisface la propiedad RSNNI o RNNI. Para el caso en que la respuesta sea negativa, se presento un segundo algoritmo el cual, si es posible, refina un sistema que sí satisface la propiedad a costa de restringir las acciones de entrada que puede éste recibir.

3. *Especificación de sistemas probabilísticos.* Se definió el formato *ntμfθ/ntμxθ* para especificar sistemas de transiciones probabilistas. En este contexto se realizaron los siguientes aportes:

   a) Se demostró que la bisimulación para sistemas probabilistas es una congruencia para todo operador cuya semántica se define utilizando el formato *ntμfθ/ntμxθ*.

   b) Se estudió además la construcción modular de especificaciones y se presenta un resultado que garantiza la extensión conservativa de especificaciones.

   c) Se demostró que la congruencia trazas para procesos con imagen finita inducida por el formato *ntμfθ/ntμxθ* es precisamente la bisimulación sobre este tipo particular de sistemas de transiciones.

8.2. Trabajos futuros

En la sección anterior podemos ver que esta tesis realiza aportes en diferentes contextos. Cada uno de estos contextos ofrece distintas posibles continuaciones.

Para el caso de las semánticas para sistemas interactivos es necesario realizar un estudio más profundo de cómo las mismas pueden aplicarse en otros contextos donde existan acciones de entrada y de salida (o lo que es equivalente, controlables y no controlables). Esto ya se mencionó en la Sección 2.4.2. Con respecto a este punto, actualmente nos encontramos estudiando el siguiente problema: en [64] se estudia cómo verificar si una implementación satisface la relación de conformidad **ioco** en un contexto donde la comunicación se realiza a través de canales asíncronos. El problema que se presenta bajo esta configuración es que un tester pueda declarar que una implementación no satisface la especificación porque confunde una acción de salida correcta, que llega tarde, con un error, i.e. una acción de salida no especificada. Nuestra actual conjetura es que tal vez se pueda caracterizar un conjunto de tester que se comporta correctamente, ya sea con comunicación síncrona o asíncrona, utilizando alguna de las nociones de observabilidad presentadas y alguna transformación sobre el sistema de transición. Esto está también relacionado con la propiedad de no interferencia pues la propiedad deseada podría expresarse como: "una acción de salida que llega tarde no interfiere con el juicio del tester". Ésta es una reciente linea de investigación, por lo tanto aún no se cuenta con suficientes indicios que permitan dar una sentencia sobre la veracidad de la conjetura.

Con respecto a la propiedades de no interferencia y las interfaces de autómatas, esta tesis no estudió cómo se comporta la propiedad de seguridad con respecto a la relación alternating simulation [25] (recordar que esta relación es como la relación SIR, Def. 2.1.11, excepto que el punto (b) no es requerido). Esta relación modela la noción de refinamiento en el contexto de autómatas de interfaces. Intuitivamente, si una IA S' refina a otra IA S, entonces S' acepta al menos las entradas que S acepta y produce a lo sumo las acciones de salida que puede producir S'. Para la misma se satisface lo siguiente: sean tres IA S, T y S' tales que $S \parallel T$ está bien definido. Si S' refina a S, entonces $S' \parallel T$ está bien definido y refina a $S \parallel T$. Algunos interrogantes que surgen para esta relación son los siguientes: como fortalecer la relación de refinamiento para garantizar que si S' refina a S y S es una interfaz segura entonces S' es también segura. Quizás S y T no son seguros, pero $S \parallel T$ si lo es, luego qué condiciones debe satisfacer S' para que $S' \parallel T$ sea

seguro. Por supuesto, estas respuestas dependerán de qué variante de V-SNNI/V-NNI se utilice para modelar la propiedad de seguridad.

Otro punto importante en esta línea de investigación es con respecto a la practicidad de lo desarrollado. En este caso, los servicios web brindan el marco propicio para realizar este estudio. Esto se debe a que ya se han realizado distintos trabajos en ese contexto que facilitarían la tarea. Por ejemplo, en [46], un autómata de interfaz es construido a partir de una especificación que describe el comportamiento de un servicio web. Esta descripción está en lenguaje OWL-S [17]. El AI construido es una especificación que se utiliza para buscar servicios webs tales que al componerse satisfacen la misma. Sumado a esto, en [29] se estudia cómo adaptar el comportamiento de una interfaz de un servicio web para llevar a cabo una comunicación exitosa. Usando este tipo de técnicas, se pueden restringir las entradas que una interfaz recibe y así obtener comportamientos que no revelen la información confidencial. Finalmente en [52] se propone un modelo para especificar las propiedades de seguridad que un servicio web satisface. Además, se define una metodología para crear contratos entre servicios, basados en las propiedades que cada uno satisface, que garantizan que la composición entre ellos tiene como resultado un nuevo servicio seguro. En este marco, las propiedades V-SNNI y V-NNI podrían ser utilizadas como nuevos atributos de seguridad. Sumado a esto, los resultados presentados en el Teorema 3.3.1, el Corolario 3.3.1 y Teorema 3.3.2 podrían ser útiles para la creación de los contratos antes mencionados.

Pero sin duda alguna, la línea dedicada a la especificación de sistemas de transiciones probabilistas es la que más oportunidades ofrece. Esto se debe principalmente a dos razones. La primera, el formato *ntμfθ/ntμxθ* guarda una estrecha similitud con el formato *ntyft/ntyxt*, el cual ha sido ampliamente estudiado. La segunda, en comparación a los otros formatos para contextos probabilísticos, el formato *ntμfθ/ntμxθ* es más elegante y mucho más expresivo. En los próximos párrafos describimos algunas excitantes líneas de investigación que pensamos continuar.

En [36] se presenta un método para descomponer formulas de una variante probabilista de la lógica Hennessy–Milner (HML) para procesos probabilísticos. Este método permite verificar la validez de una fórmula con respecto a un proceso, evaluando si los subtérminos del proceso satisfacen las fórmulas obtenidas a través de una descomposición de la fórmula original. Esta descomposición es definida en función del formato utilizado para definir los operadores. Particularmente, en [36] se utiliza el formato RTSS [53]. Por ser nuestro formato más expresivo, extender estos resultados remplazando el formato RTSS por el formato *ntμfθ/ntμxθ* podría presentar mejoras significativas con respecto a la técnica actual. Es más, la reducción del formato *ntμfθ/ntμxθ* al formato pntree (Capítulo 6) presenta parte de los resultados necesarios para realizar esta tarea.

Otra linea interesante está relacionada con la relajación de condiciones que se imponen para considerar que dos sistemas probabilísticos son equivalentes. En general, éstas requieren que los dos sistemas tengan exactamente el mismo comportamiento probabilístico. En ciertos casos esto es muy restrictivo, por esta razón se presentan variantes de estas nociones donde se relaja este requisito. En [73] se estudia este problema. Para esto se revisan distintas nociones de distancia entre procesos y se trabaja con el concepto de *no–expansión*: un operador f con n argumentos es *no–expansivo*, con respecto a una noción de distancia, si para los procesos $s_1, t_1, \ldots, s_n, t_n$, tales que la distancia entre s_i y t_i es ε_i, se satisface que la distancia entre $f(s_1, \ldots, s_n)$ y $f(t_1, \ldots, t_n)$ está acotada por $\varepsilon_1 + \ldots + \varepsilon_n$. Basados en el formato RTSS, se presentan formatos de reglas

para las distintas variantes de no-expansión. Sería una contribución interesante extender estos resultados al formato $nt\mu f\theta/nt\mu x\theta$.

La última posibilidad que mencionaremos es una en la cual nos encontramos trabajando. Dado que se utilizan términos algebraicos para representar procesos, sería útil contar con axiomas, por ejemplo $x + \mathbf{0} = x$, que permitan realizar una manipulación efectivamente algebraica de los mismos. En [1] se presenta un algoritmo que, dada una especificación en formato GSOS, crea un sistema completo de axiomas. Es decir, dos términos representan sistemas de transiciones equivalentes (para alguna semántica, puntualmente en [1] es la bisimulación) sí y sólo sí se puede demostrar la igualdad de términos utilizando los axiomas derivados. Si bien sólo recientemente hemos empezado a trabajar en un resultado análogo en el contexto probabilista, ha surgido un interrogante interesante a nivel conceptual:

¿Por qué utilizar términos en $T(\Sigma)$ para representar sistemas de transiciones probabilistas si los mismos no manipulan probabilidades?

Según nuestro estudio preliminar, representar procesos mediante un objeto que manipule probabilidades facilitaría la extensión del algoritmo al contexto probabilístico. Esta afirmación puede relacionarse con los resultados presentados en [20]. En este trabajo se estudia una variante de sistemas de transiciones etiquetadas, denominada *DLTS*, los cuales están compuestos por transiciones de la forma $\Delta(Q) \times A \times \Delta(Q)$ donde $\Delta(Q)$ es el espacio de distribuciones sobre un conjunto de estados Q y A es un conjunto de acciones. Para este modelo se demuestra que el espectro semántico para sistemas de transiciones etiquetadas, así también como la caracterización lógica de éstas en términos de fórmulas HML, escala directamente al contexto de transiciones probabilistas cuando estos son representados mediante DLTS.

Llevar estas ideas a nuestro contexto implicaría reemplazar las transiciones de la forma $T(\Sigma) \times A \times \Delta(T(\Sigma))$ por transiciones de la forma $\Delta(T(\Sigma)) \times A \times \Delta(T(\Sigma))$, adaptar las ideas de [20] y los resultados presentados en esta tesis. Esto no debería presentar mayores inconvenientes, dado que los términos guardan una estrecha relación con los términos de distribuciones.

Apéndice A

Pruebas

A.1. Demostración del Lema 3.2.1

Lema. *Sean S y T dos IA y sea $V = \{a^O, \top, \tau, \varepsilon, \vee, \rightleftarrows, RT, \bigwedge, \eta, b\}$ una noción de observabilidad. Entonces vale $S \geqslant T$ sii $O_V(S) \supseteq O_V(T)$.*

Demostración. Supongamos que vale $p \geqslant q$ y sea $\phi \in O(q)$. Demostremos por inducción en la estructura de la fórmula ϕ que si $\phi \in O(q)$ entonces $\phi \in O(p)$, i.e.$O(p) \supseteq O(q)$. Supongamos que la hipótesis vale para las fórmulas ψ y ϕ. Dada la definición de V, una observación puede tener distintas formas y ser válida por distintas reglas. A continuación la prueba para cada una de las posibilidades.

- Sea $a\psi \in O_V(q)$ con $a \in A^I \cup A^O$. Luego $q \Rightarrow \xrightarrow{a} q'$ y $q' \models \psi$. Por (b), (c) y (d) de la Def. 2.1.11 existe p' tal que $p \Rightarrow \xrightarrow{a} p'$ y $p' \geqslant q'$. Por hipótesis inductiva $p' \models \psi$. Aplicando la regla τ y alguna de las reglas (a^O) o ($\rightleftarrows$) podemos concluir $\phi = a\psi \in O_V(p)$.

- Sea $X\psi \in O_V(q)$ con $I(q) = X$: por (a) y (b) de la Def. 2.1.11 podemos concluir que $I(p) = X$. Por hipótesis inductiva $p \models \psi$. Aplicando la regla (RT) podemos concluir $X\psi \in O_V(p)$

- Sea $\psi(a\phi) \in O_V(q)$ con $a \in A^I$. Entonces $q \Rightarrow q' \xrightarrow{a} q''$, $q' \models \psi$ y $q'' \models \phi$: por (b) y (d) de la Def. 2.1.11 existe p' y p'' tal que $p \Rightarrow p' \xrightarrow{a} p''$, $p' \geqslant q'$ y $p'' \geqslant q''$. Por hipótesis inductiva $p'' \models \phi$. Además $\psi \in O(q')$, luego $\psi \in O(p')$ y por lo tanto $p' \models \psi$. Aplicando la regla ($\eta_{\rightleftarrows}$) podemos concluir $\psi(a\phi) \in O_V(p')$. Aplicando la regla (τ) podemos concluir $\psi(a\phi) \in O_V(p)$.

- Los casos $\varepsilon\psi$ y $\psi\tau\phi$ son similares a los anteriores.

- Los casos $\psi = \bigwedge_{i \in I} \phi_i, \psi = \bigvee_{i \in I} \phi_i$ son directos.

Para demostrar la vuelta, definamos para todo proceso p, q la relación $p \succcurlyeq q$ si $O(p) \supseteq O(q)$. Sean p y q tales que $O(p) \supseteq O(q)$ y demostremos que las condiciones de la Def. 2.1.11 se satisfacen.

- Sea $p \xrightarrow{a} p'$ con $a \in A^I$. Luego $a \in X = I(p)$ y existe observación $X\psi_0\tau\psi_1 \in O_V(p)$. Supongamos $I(q) = X' \neq X$, entonces existe observación $X'\psi'_0\tau\psi'_1 \in O_V(q)$ tal que $X'\psi'_0\tau\psi'_1 \notin O_V(p)$, absurdo. Luego $X' = X$ y por lo tanto $q \xrightarrow{a} q'$. Dado que $O(p) \supseteq O(q)$ es directo demostrar que $O(p') \supseteq O(q')$.
- Sea $q \xrightarrow{a} q'$ con $a \in A^I$. La prueba es análoga al caso anterior.
- Sea $q \xrightarrow{a} q'$ con $a \in A^O$. Este caso es igual al mismo caso en la prueba del Teorema 2.3.1.
- Sea $q \xrightarrow{a} q'$ con $a \in A^H$. Entonces debemos demostrar que existe p' tal que $p \Rightarrow p'$ y $O_V(p') \supseteq O_V(q')$. Si $O_V(p) \supseteq O_V(q')$ no hay nada que demostrar, entonces supongamos lo opuesto, sea entonces $\psi_p \in O_V(q')\backslash O_V(p)$. Sea S el conjunto $\{p' \mid p \rightarrow\!\Rightarrow p'\}$. Si para todo $s \in S$ vale $O_V(s) \not\supseteq O_V(q')$ entonces debe existir una formula $\psi_s \in O_V(q')\backslash O_V(s)$ para cada s. Por lo tanto la formula $\bigwedge_{s\in S} \psi_s$ no es válida para todo p' tal que $p \rightarrow\!\Rightarrow p'$ (siempre falla al menos un ψ_s). Luego $q \models \top\tau(\psi_p \wedge (\bigwedge_{s\in S} \psi_s))$ pero $p \not\models \top\tau(\psi_p \wedge \bigwedge_{s\in S} \psi_s))$. Absurdo pues $O(p) \supseteq O(q)$.

□

A.2. Desmostración del Lema 5.3.5

Lema. *Sea P un PTSS puro en formato nt$\mu f\theta$/nt$\mu x\theta$. Sea $H \subseteq T(\Sigma)$ un conjunto de literales negativos cerrados, $t, t' \in T(\Sigma)$, $a \in A$, $\pi \in \mathsf{DT}(\Sigma)$ tales que $P \vdash (\frac{H}{t \xrightarrow{a} \pi})$ con prueba $\mathfrak{p}(\frac{H}{t \xrightarrow{a} \pi})$ y $t \mathrel{R_P} t'$, entonces existen $H' \subseteq T(\Sigma)$, un conjunto de literales negativos, y $\pi' \in \mathsf{DT}(\Sigma)$ tales que $P' \vdash (\frac{H'}{t' \xrightarrow{a} \pi'})$ con prueba $\mathfrak{p}(\frac{H'}{t' \xrightarrow{a} \pi'})$, $\pi \mathrel{R_P} \pi'$ y para todo $h' \in H'$*

1. *existe $h \in H$ tal que $h \mathrel{R_P} h'$ o*
2. *existen literales positivos $\eta \in \mathfrak{p}(\frac{H}{t \xrightarrow{a} \pi})$, $\eta' \in \mathfrak{p}(\frac{H'}{t' \xrightarrow{a} \pi'})$ y conjuntos $H_\eta \subseteq H$, $H_{\eta'} \subseteq H'$ tales que $src(\eta) \sim src(\eta')$, $h' \in H_{\eta'}$, $P \vdash \frac{H_\eta}{\eta}$ y $P \vdash \frac{H_{\eta'}}{\eta'}$.*

Demostración. Por los corolarios 5.3.1 y 5.3.2 podemos asumir que P está en formato *nt$\mu f\theta$* y que es puro. Procedemos por inducción completa en $\gamma = h(\mathfrak{p}(H/t \xrightarrow{a} \pi))$; supongamos que la hipótesis, que es directa del enunciado del lema, vale para $\gamma' < \gamma$.

Si $t \mathrel{R_P} t'$ porque se satisface la condición (I) de la Definición 5.3.4, $t \sim t'$ y $t \xrightarrow{a} \pi$ implica que existe π' tal que $t' \xrightarrow{a} \pi'$ y $\pi \sim \pi'$. Sea $H' \subseteq T(\Sigma)$, un conjunto de literales negativos, y sea $\pi'\mathsf{DT}(\Sigma)$ tal que $P \vdash \frac{H'}{t \xrightarrow{a} \pi'}$ con prueba $\mathfrak{p}(\frac{H'}{t \xrightarrow{a} \pi'})$. Para probar que (2) vale para todo $h' \in H'$ sólo se necesita definir $\eta = t \xrightarrow{a} \pi$, $\eta' = t' \xrightarrow{a} \pi'$, $H_\eta = H$ y $H_{\eta'} = H'$.

Procedamos entonces con el caso en el que $t \mathrel{R_P} t'$ vale porque se satisface la condición (II) de la Definición 5.3.4: $t = f(u_1, \ldots, u_{r(f)})$, $t' = f(v_1, \cdots, v_{r(f)})$ y $u_i \mathrel{R_P} v_i$ con $u_i, v_i \in T(\Sigma)$ para

$i \in \{1, \ldots, \mathsf{r}(f)\}$. Por Def. 4.3.1, la prueba $\mathfrak{p}(H/f(u_1, \ldots, u_{\mathsf{r}(f)}) \xrightarrow{a} \pi)$ tiene su raíz etiquetada con $f(u_1, \ldots, u_{\mathsf{r}(f)}) \xrightarrow{a} \pi$ y si K es el conjunto de etiquetas que se encuentran por encima de ésta, entonces existe $r \in R$ y una substitución cerrada ρ tal que $\rho(r) = \frac{K}{f(u_1,\ldots,u_{\mathsf{r}(f)}) \xrightarrow{a} \pi}$. La regla r tiene la siguiente forma:

$$\frac{\bigcup_{m\in M}\{t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m : \vec{z} \in \mathcal{Z}\} \cup \bigcup_{n\in N}\{t_n(\vec{z}) \xrightarrow{b_n} \not\rightarrow : \vec{z} \in \mathcal{Z}\}\{\theta_l(Y_l) \trianglerighteq_{l,k} p_{l,k} : l \in L, k \in K_l\}}{f(x_1, \ldots, x_{\mathsf{r}(f)}) \xrightarrow{a} \theta}$$

y ρ es tal que para todo $1 \leq k \leq \mathsf{r}(f)$, $\vec{z} \in \mathcal{Z}$, $m \in M$, $n \in N$, $l \in L$ y $k \in K_l$ las siguientes condiciones son válidas

(ρ1) $\rho(x_k) = u_k$,

(ρ2) $\pi = \rho(\theta)$,

(ρ3) $P \vdash \frac{H}{\rho(t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m)}$ con prueba $\mathfrak{p}(H/\rho(t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m))$, subprueba de $\mathfrak{p}(H/f(u_1, \ldots, u_{\mathsf{r}(f)}) \xrightarrow{a} \pi)$, y por lo tanto $h(\mathfrak{p}(H/\rho(t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m))) < \gamma$,

(ρ4) $\rho(t_n(\vec{z}) \xrightarrow{b_n} \not\rightarrow) \in H$.

(ρ5) $\rho(\theta_l(Y_l)) \trianglerighteq_{l,k} p_{l,k}$.

Ahora necesitamos definir un substitución genuina ρ' que en conjunto con la regla r y la hipótesis inductiva permita concluir hipótesis.

Notar que ρ' debe ser tal que las premisas cuantitativas sean satisfechas[1], para esto, definamos $\rho(Y_l)/\mathsf{R}_P = \{\rho(Y_l) \cap [t]_{\mathsf{R}_P} \mid t \in T(\Sigma)\}$. Es decir, obtenemos una partición de $\rho(Y_l)$ inducida por los átomos del álgebra de Bool $\langle\{X : \mathsf{R}_P\text{-closed}(X)\}, \subseteq\rangle$. Para cada conjunto $d \in \rho(Y_l)/\mathsf{R}_P$ definamos $d^{\uparrow}$ como el único átomo del álgebra de Bool $d^{\uparrow} \in A(\mathsf{R}_P)$ tal que $d \subseteq d^{\uparrow}$. Ahora definamos Ξ, una partición $\mathcal{Z}$, tal que para todo $l \in L$ y $d_l \in \rho(Y_l)/\mathsf{R}_P$ hay un elemento de la partición $Z_{\prod_{l\in L} d_l} \in \Xi$ tal que

(Ξ1) $|Z_{\prod_{l\in L} d_l}| \geq \omega$ con $\omega = |\mathbb{N}|$.

(Ξ2) $\exists \vec{z} \in Z_{\prod_{l\in L} d_l} : \rho(\vec{z}) \in \prod_{l\in L} d_l \times \prod_{k=1}^{\mathsf{r}(f)}\{u_k\}$

(Ξ3) si $\mu^{\vec{z}}_m \in Var(\theta)$ o $\mu^{\vec{z}}_m \in Var(\theta_l)$ para algún $l \in L$, $\vec{z} \in \mathcal{Z}$, $m \in M$ entonces si $\rho(\vec{z}) \in \prod_{l\in L} d_l \times \prod_{k=1}^{\mathsf{r}(f)}\{u_k\}$ entonces $\vec{z} \in Z_{\prod_{l\in L} d_l}$.

(Ξ4) si $\vec{z}(i) \in Var(\theta)$ para $\vec{z} \in \mathcal{Z}$, $i \in \{1, \ldots, |z|\}$ entonces $\vec{z} \in Z_{\prod_{l\in L} d_l}$.

Notar que la cardinalidades de los conjuntos Y_l's, L (restricción 1 en la Def. 5.2.3) y M junto a que la cantidad de elementos que pueden satisfacer las condiciones (Ξ3) y (Ξ4) son finitos, garantizan que tal partición puede ser construida. La condición (Ξ1) asegurará que existen suficientes variables para definir la substitución ρ' de forma tal que las premisas cuantitativas sean

[1] Esta parte de la demostración es exactamente igual en la demostración del Teorema 5.3.2

satisfechas. La condición (Ξ2) será fundamental para relacionar los términos cerrados obtenidos mediante la substitución ρ' con los términos cerrados obtenidos mediante ρ a través de la relación R_P. Notemos que para cada partición de términos cerrados $\prod_{l\in L} d_l \times \prod_{k=1}^{\mathsf{r}(f)}\{u_k\}$ existe una partición de variables $Z_{\prod_{l\in L} d_l}$ (notar el subíndice de $Z_{\prod_{l\in L} d_l}$) la cual contiene al menos un vector de variables que es instanciado por ρ a un valor en la primera partición, i.e. $\exists \vec{z} \in Z_{\prod_{l\in L} d_l}$. Luego ρ' instanciará en $Z_{\prod_{l\in L} d_l}$ los términos cerrados que se relacionan con los elementos de $\prod_{l\in L} d_l \times \prod_{k=1}^{\mathsf{r}(f)}\{u_k\}$. Teniendo en cuenta esto, la condición (Ξ3), una vez definida ρ', implica $\rho(\vec{z})\ \mathsf{R}_P\ \rho'(\vec{z})$ para los $\vec{z}$ tales que $\mu^{\vec{z}}$ se usa en algún término de distribución. De forma similar, la condición (Ξ4) implicará, si $\vec{z}(i) \in Var(\theta)$, $\rho(\vec{z}(i)) = \rho'(\vec{z}(i))$. Ambos hechos serán importantes para demostrar los puntos relacionados a las premisas cuantitativas. Definamos ahora la substitución genuina ρ' tal que para toda variable $\zeta \in Var(r)$ se satisfacen las siguientes propiedades:

Definamos ahora la substitución genuina ρ' de forma tal que para toda variable $\zeta \in Var(r)$ se satisfacen las siguientes propiedades:

($\rho'1$) si $\zeta = x_k$ para algún $1 \le k \le \mathsf{r}(f)$ entonces $\rho'(\zeta) = v_k$ y por lo tanto $\rho(\zeta)\ \mathsf{R}_P\ \rho'(\zeta)$;

($\rho'2$) si $\zeta = \mu_m^{\vec{z}}$ con $m \in M$ y $\vec{z} \in \mathcal{Z}$, entonces existe $\vec{z'} \in \mathcal{Z}$ tal que $\rho(t_m(\vec{z'}) \xrightarrow{a_m} \mu_m^{\vec{z'}})\ \mathsf{R}_P\ \rho'(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}})$. Más aún, si $\mu_m^{\vec{z}} \in Var(\theta)$ o $\mu_m^{\vec{z}}$ aparece en una premisa cuantitativa, entonces $\rho'(\mu_m^{\vec{z}})\ \mathsf{R}_P\ \rho(\mu_m^{\vec{z}})$. Además existe $H_m^{\vec{z}}$ tal que $P \vdash H_m^{\vec{z}}/\rho'(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}})$ con una prueba $\mathfrak{p}(H_m^{\vec{z}}/\rho'(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}}))$ y para todo $h^{\vec{z}} \in H_m^{\vec{z}}$

- *a*) existe etiqueta $h \in H$ tal que $h\ \mathsf{R}_P\ h^{\vec{z}}$ o
- *b*) existen literales positivos $\eta \in \mathfrak{p}(H/\rho(t_m(\vec{z'}) \xrightarrow{a_m} \mu_m^{\vec{z'}}))$, $\eta^{\vec{z}} \in \mathfrak{p}(H_m^{\vec{z}}/\rho'(t_m(\vec{z}) \xrightarrow{a_m} \mu_m^{\vec{z}}))$ y conjuntos $H_\eta \subseteq H$, $H_{\eta^{\vec{z}}} \subseteq H_m^{\vec{z}}$ tales que $src(\eta) \sim src(\eta^{\vec{z}})$, $h^{\vec{z}} \in H_{\eta^{\vec{z}}}$, $P \vdash H_\eta/\eta$ y $P \vdash H_{\eta^{\vec{z}}}/\eta^{\vec{z}}$.

($\rho'3$) si $\zeta = y \in Y_l$, entonces Y_l es tal que [2]:

- *a*) $\rho'(\pi_l(Z_{\prod_{l\in L} d_l})) = d_l^{\uparrow} \cap \mathsf{support}(\rho'(\theta_l))$ (aquí, π_l indica la proyección de la l-ésima componente),
- *b*) para todo $t_m(\vec{z})$ con $\vec{z} = \langle z_0, \ldots, z_{|\vec{z}|}\rangle$, existe $\vec{z'} = \langle z'_0, \ldots, z'_{|\vec{z}|}\rangle$ tal que para todo $i \in \{0, \ldots, |\vec{z}|\}$ si $n_{\mathrm{VDG}}(z_i) \le n_{\mathrm{VDG}}(y)$ entonces $\rho'(z_i)\ \mathsf{R}_P\ \rho(z'_i)$.
- *c*) si $\mu_m^{\vec{z}} \in Var(\theta)$ o $\mu_m^{\vec{z}}$ aparece en una premisa cuantitativa, entonces $\rho'(z_i)\ \mathsf{R}_P\ \rho(z_i)$.
- *d*) si $y \in Var(\theta)$ entonces $\rho(y)\ \mathsf{R}_P\ \rho'(y)$

Definimos $\rho'(x)$ utilizando inducción en $n = n_{\mathrm{VDG}}(x)$ con respecto a grafo de dependencia del conjunto $\mathrm{pprem}(r) \cup \mathrm{qprem}(r)$. La definición será tal que las propiedades enunciadas son directas. Supongamos que ρ' está definida para las variables ζ' tales que $n_{\mathrm{VDG}}(\zeta') < n$, definamos ρ' para ζ tal que $n_{\mathrm{VDG}}(\zeta) = n$. Luego hay 3 casos que analizar $\zeta = x_i$, $\zeta = \mu_m^{\vec{z}}$ y $\zeta \in Y_{l'}$.

Si $\zeta = x_i$ para $i \in \{1, \ldots, \mathsf{r}(f)\}$. Entonces definamos $\rho'(\zeta) = v_i$. De esta forma se satisface la condición ($\rho'1$).

[2] Notar que $\forall y' \in Y_l : n_{\mathrm{VDG}}(y) = n_{\mathrm{VDG}}(y')$, esto permite definir la substitución y la propiedad para todo el conjunto Y_l al mismo tiempo

Si $\zeta = \mu^{\vec{z}}_m$, dado que $n_{\text{VDG}}(\mu^{\vec{z}}_m) = n$, tenemos que para todo $x' \in Var(t_m(\vec{z}))$ vale $n_{\text{VDG}}(x') < n$. Por la condición $(\rho' 3b)$, podemos asegurar que existe un $\vec{z'}$ tal que $\rho'(t_m(\vec{z}))\ \mathsf{R}_P\ \rho(t_m(\vec{z'}))$. Más aun, si las condiciones enunciadas en $(3c)$ se satisfacen, podemos asegurar $\rho'(t_m(\vec{z}))\ \mathsf{R}_P\ \rho(t_m(\vec{z}))$. Por $(\rho 3)$, dado que $t_m(\vec{z'}) \xrightarrow{a_m} \mu^{\vec{z}}_m$ es también una premisa positiva de r, $P \vdash \frac{H}{\rho(t_m(\vec{z'})) \xrightarrow{a_m} \rho(\mu^{\vec{z}}_m)}$ con una prueba $\mathfrak{p}(H/\rho(t_m(\vec{z'})) \xrightarrow{a_m} \rho(\mu^{\vec{z}}_m))$ tal que $h(\mathfrak{p}(H/\rho(t_m(\vec{z'})) \xrightarrow{a_m} \rho(\mu^{\vec{z}}_m))) < \gamma$. Ahora, por definición de R_P, dos casos se presentan:

(c1) $\rho(t_m(\vec{z'})) \sim \rho'(t_m(\vec{z}))$.

(c2) Existe un nombre de función $g \in F$ y términos $u_k, u'_k \in T(\Sigma)$, $1 \leq k \leq \mathsf{r}(g)$, tales que

- $\rho(t_m(\vec{z'})) = g(w_1, \ldots, w_{\mathsf{r}(g)})$,
- $\rho'(t_m(\vec{z})) = g(w'_1, \ldots, w'_{\mathsf{r}(g)})$, y
- $u_k\ \mathsf{R}_P\ u'_k$ para todo $1 \leq k \leq \mathsf{r}(g)$.

Para (c1), repitiendo el análisis que se hizo para el caso $t\ \mathsf{R}_P\ t'$ porque $t \sim t'$, y para (c2), usando la primera hipótesis inductiva (recordar $h(H/\mathfrak{p}(\rho(t_m(\vec{z'})) \xrightarrow{a_m} \rho(\mu^{\vec{z}}_m))) < \gamma$), se concluye que existen $\pi \in \mathsf{DT}(\Sigma)$, $H^{\vec{z}}_m$ tales que $P \vdash \frac{H^{\vec{z}}_m}{\rho'(t_m(\vec{z})) \xrightarrow{a_m} \pi}$, $\rho(\mu^{\vec{z}}_m)\ \mathsf{R}_P\ \pi$ y para todo $h^{\vec{z}} \in H^{\vec{z}}_m$

- existe $h \in H$ tal que $h\ \mathsf{R}_P\ h^{\vec{z}}$ o
- existen literales positivos $\eta \in \mathfrak{p}(H/\rho(t_m(\vec{z'}) \xrightarrow{a_m} \mu^{\vec{z}}_m))$, $\eta^{\vec{z}} \in \mathfrak{p}(H^{\vec{z}}_m/\rho'(t_m(\vec{z}) \xrightarrow{a_m} \mu^{\vec{z}}_m))$ y conjuntos $H_\eta \subseteq H$, $H_{\eta^{\vec{z}}} \subseteq H^{\vec{z}}_m$ tales que $src(\eta) \sim src(\eta^{\vec{z}})$, $h^{\vec{z}} \in H_{\eta^{\vec{z}}}$, $P \vdash H_\eta/\eta$ y $P \vdash H_{\eta^{\vec{z}}}/\eta^{\vec{z}}$.

Por lo tanto, para ambos casos podemos definir $\rho'(\mu^{\vec{z}}_m) = \pi$ y satisfacer las condiciones de $(\rho' 2)$.

Sea ahora $\zeta \in Y_{l'}$ para algún $l' \in L$. En este caso definimos la substitución para todo el conjunto $Y_{l'}$ al mismo tiempo. Para toda variable de término en $Y_{l'}$, definamos ρ' tal que para cada $Z_{\prod_{l \in L} d_l} \in \Xi$ vale

$$\rho'(\pi_{l'}(Z_{\prod_{l \in L} d_l})) = d^{\uparrow}_{l'} \cap \mathsf{support}(\rho'(\theta_{l'}))$$

Esta condición se puede garantizar por $(\Xi 1)$. Además las condiciones $(\Xi 2)$, $(\Xi 3)$ y $(\Xi 4)$ aseguran las distintas propiedades de $(\rho' 3)$.

La substitución ρ' está totalmente definida, entonces sea:

$$H' = \bigcup_{\vec{z} \in \mathcal{Z}, m \in M} H^{\vec{z}}_m \cup \bigcup_{n \in N} \{\rho'(t_n(\vec{z})) \xrightarrow{b_n} \not\rightarrow : \vec{z} \in \mathcal{Z}\}$$

Por $(\rho' 2)$ y $\rho'(\text{nprem}(r)) \subseteq H$, para construir una prueba de $P \vdash \frac{H}{f(v_1, \ldots, v_{\mathsf{r}f}) \xrightarrow{a} \pi'}$ con $\pi' = \rho'(\theta)$ sólo necesitamos demostrar que toda premisa en $\rho'(\text{qprem}(r))$ vale. Sea $\theta_l(Y_l) \trianglerighteq_{l,k} p_{l,k}$ una premisa cuantitativa de r. Si $\zeta \in Var(\theta_l)$, por la condición 7 de la Def. 5.2.3 y dado que la regla no tiene variables, ζ es una variable de distribución. Más aún ζ aparece en el destino de una premisa positiva; entonces $\zeta = \mu^{\vec{z}}_m$ con $m \in M$ y $\vec{z} \in \mathcal{Z}$. Por la condición $(\rho' 2)$, $\rho'(\mu^{\vec{z}}_m)\ \mathsf{R}_P\ \rho(\mu^{\vec{z}}_m)$.

Por el Lema 5.3.4 tenemos que $\rho'(\theta_l)\ \mathsf{R}_P\ \rho(\theta_l)$. Además, por ($\rho$5), vale $\rho(\theta_l(Y_l)) \trianglerighteq_{l,k}\ p_{l,k}$. El siguiente cálculo es suficiente para probar que las premisas cuantitativas de r valen bajo ρ':

$$\begin{aligned}
\rho'(\theta_l)(\rho'(Y_l)) &= \sum_{d' \in \rho'(Y_l)/\mathsf{R}_P} \rho'(\theta_l)(d') \\
&= \sum_{d \in \rho(Y_l)/\mathsf{R}_P} \rho'(\theta_l)(d^{\uparrow}) && \text{(por } (\rho'3c)) \\
&= \sum_{d \in \rho(Y_l)/\mathsf{R}_P} \rho(\theta_l)(d^{\uparrow}) && \text{(por Lema 5.3.4)} \\
&\geq \sum_{d \in \rho(Y_l)/\mathsf{R}_P} \rho(\theta_l)(d) && (d \subseteq d^{\uparrow}) \\
&= \rho(\theta_l)(\rho(Y_l)) \\
&\trianglerighteq_{l,k}\ p_{l,k} && \text{(dado que } \rho(\theta_l(Y_l)) \trianglerighteq_{l,k}\ p_{l,k})
\end{aligned}$$

Por lo tanto, para todo $l \in L$, $k \in K_l$ y $\vec{z} \in \mathcal{Z}$, vale $\rho'(\theta_l(Y_l)) \trianglerighteq_{l,k}\ p_{l,k}$. Dado que todas las premisas de $\rho'(r)$ valen, podemos concluir que $P \vdash H'/\rho'(\mathrm{conc}(r))$.

Solamente resta demostrar que $\rho(\theta)\ \mathsf{R}_P\ \rho'(\theta)$, pero por Lema 5.3.4 esto es directo. Recordemos que la regla no tiene variables libres, entonces si $\zeta \in \mathit{Var}(\theta)$ y ζ es una variable de distribución, tenemos que $\zeta = \mu_m^{\vec{z}}$ para algún $\vec{z} \in \mathcal{Z}$ y $m \in M$. Por condición ($\rho'2$) vale $\rho(\mu_m^{\vec{z}})\ \mathsf{R}_P\ \rho'(\mu_m^{\vec{z}})$. Por otro lado, si ζ es una variable de término, por las condiciones ($\rho'1$) y ($\rho'3d$) vale que $\rho(\zeta)\ \mathsf{R}_P\ \rho'(\zeta)$. Es decir que las condiciones del Lema 5.3.4 son satisfechas.

Sólo resta demostrar que para todo $h' \in H'$ vale (1) o (2), lo cual es directo por construcción (ver $\rho'2$)).

□

Bibliografía

[1] Luca Aceto, Bard Bloom, and Frits W. Vaandrager. Turning SOS rules into equations. *Inf. Comput.*, 111(1):1–52, 1994.

[2] Luca Aceto, Wan Fokkink, and Chris Verhoef. Conservative extension in structural operational semantics. In *Current Trends in Theoretical Computer Science*, pages 504–524. 2001.

[3] Luca Aceto, Wan Fokkink, and Chris Verhoef. Structural operational semantics. In *Handbook of Process Algebra*, pages 197–292. Elsevier, 2001.

[4] Rajeev Alur, Costas Courcoubetis, and David L. Dill. Model-checking for real-time systems. In *LICS*, pages 414–425. IEEE Computer Society, 1990.

[5] Rajeev Alur, Costas Courcoubetis, and Thomas A. Henzinger. The observational power of clocks. In Jonsson and Parrow [50], pages 162–177.

[6] Jos C. M. Baeten, Twan Basten, and Michel A. Reniers. *Process Algebra: Equational Theories of Communicating Processes*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[7] Jos C. M. Baeten and Chris Verhoef. A congruence theorem for structured operational semantics with predicates. In Best [14], pages 477–492.

[8] Christel Baier. *On Algorithmics Verification Methods for Probabilistic Systems*. Habilitation thesis, University of Mannheim, 1999.

[9] J.W. Bakker, J.N. Kok, J.-J.Ch. Meyer, E.-R. Olderog, and J.I. Zucker. Contrasting themes in the semantics of imperative concurrency. In J.W. Bakker, W.P. Roever, and G. Rozenberg, editors, *Current Trends in Concurrency*, volume 224 of *Lecture Notes in Computer Science*, pages 51–121. Springer Berlin Heidelberg, 1986.

[10] Falk Bartels. *On Generalised Coinduction and Probabilistic Specification Formats*. PhD thesis, Vrije Universiteit, 2004.

[11] Falk Bartels, Ana Sokolova, and Erik P. de Vink. A hierarchy of probabilistic system types. *Theor. Comput. Sci.*, 327(1-2):3–22, 2004.

[12] Gilles Benattar, Franck Cassez, Didier Lime, and OlivierH. Roux. Synthesis of non-interferent timed systems. In Joël Ouaknine and FritsW. Vaandrager, editors, *Formal Modeling and Analysis of Timed Systems*, volume 5813 of *Lecture Notes in Computer Science*, pages 28–42. Springer Berlin Heidelberg, 2009.

[13] Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In Jörg Desel, Wolfgang Reisig, and Grzegorz Rozenberg, editors, *Lectures on Concurrency and Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 87–124. Springer, 2003.

[14] Eike Best, editor. *CONCUR '93, 4th International Conference on Concurrency Theory, Hildesheim, Germany, August 23-26, 1993, Proceedings*, volume 715 of *Lecture Notes in Computer Science*. Springer, 1993.

[15] Bard Bloom, Sorin Istrail, and Albert R. Meyer. Bisimulation can't be traced. *J. ACM*, 42(1):232–268, 1995.

[16] Roland Bol and Jan Friso Groote. The meaning of negative premises in transition system specifications. *J. ACM*, 43(5):863–914, 1996.

[17] Mark Burstein, Jerry Hobbs, Ora Lassila, Drew Mcdermott, Sheila Mcilraith, Srini Narayanan, Massimo Paolucci, Bijan Parsia, Terry Payne, Evren Sirin, Naveen Srinivasan, and Katia Sycara. OWL-S: Semantic Markup for Web Services, November 2004.

[18] Franck Cassez, John Mullins, and Olivier H. Roux. Synthesis of non-interferent systems. In *4th Int. Conf. on Mathematical Methods, Models and Architectures for Computer Network Security (MMM-ACNS'07)*, volume 1 of *Communications in Computer and Inform. Science*, pages 307–321. Springer, 2007.

[19] Keith L. Clark. Negation as failure. In *Logic and Data Bases*, pages 293–322, 1977.

[20] Silvia Crafa and Francesco Ranzato. A spectrum of behavioral relations over LTSs on probability distributions. In Joost-Pieter Katoen and Barbara König, editors, *CONCUR 2011 – Concurrency Theory*, volume 6901 of *Lecture Notes in Computer Science*, pages 124–139. Springer Berlin Heidelberg, 2011.

[21] Pedro R. D'Argenio, Holger Hermanns, and Joost-Pieter Katoen. On generative parallel composition. *Electr. Notes Theor. Comput. Sci.*, 22, 1999.

[22] Pedro R. D'Argenio and Matias David Lee. Probabilistic transition system specification: Congruence and full abstraction of bisimulation. In Lars Birkedal, editor, *Foundations of Software Science and Computational Structures*, volume 7213 of *Lecture Notes in Computer Science*, pages 452–466. Springer Berlin Heidelberg, 2012.

[23] Pedro R. D'Argenio, Pedro Sánchez Terraf, and Nicolás Wolovick. Bisimulations for nondeterministic labelled Markov processes. *Mathematical Structures in Computer Science*, 22(1):43–68, 2012.

[24] Pedro R. D'Argenio and Chris Verhoef. A general conservative extension theorem in process algebras with inequalities. *Theor. Comput. Sci.*, 177(2):351–380, 1997.

[25] Luca de Alfaro and Thomas Henzinger. Interface-based design. In Manfred Broy, Johannes Grünbauer, David Harel, and Tony Hoare, editors, *Engineering Theories of Software Intensive Systems*, volume 195 of *NATO Science Series*, pages 83–104. Springer, 2005.

[26] Luca de Alfaro and Thomas A. Henzinger. Interface automata. In *Proceedings of the 8th European software engineering conference held jointly with 9th ACM SIGSOFT international symposium on Foundations of software engineering*, ESEC/FSE-9, pages 109–120, New York, NY, USA, 2001. ACM.

[27] Robert de Simone. Higher-level synchronising devices in Meije-SCCS. *Theor. Comput. Sci.*, 37:245–267, 1985.

[28] Eric Destefanis. Semánticas de procesos para sistemas interactivos. Master's thesis, Facultad de Matemática, Astronomía y Física - UNC, Córdoba, Argentina, 2013.

[29] Marlon Dumas, Kenneth W.S. Wang, and Murray L. Spork. Adapt or perish: Algebra and visual notation for service interface adaptation. In Schahram Dustdar, Jose-Luis Fiadeiro, and Amit Sheth, editors, *4th International Conference on Business Process Management*, pages 65–80, Vienna, Austria, 2006. Springer.

[30] Jean-Claude Fernandez and Laurent Mounier. "On the fly" verification of behavioural equivalences and preorders. In *Proceedings of the 3rd International Workshop on Computer Aided Verification*, CAV '91, pages 181–191, London, UK, 1992. Springer.

[31] Riccardo Focardi and Roberto Gorrieri. A classification of security properties for process algebras. *Journal of Computer Security*, 3:5–33, 1994.

[32] Riccardo Focardi and Roberto Gorrieri. Classification of security properties (part I: Information flow). In *Revised versions of lectures given during the IFIP WG 1.7 International School on Foundations of Security Analysis and Design on Foundations of Security Analysis and Design: Tutorial Lectures*, FOSAD '00, pages 331–396, London, UK, 2001. Springer.

[33] Wan Fokkink. Unification for infinite sets of equations between finite terms. *Information Processing Letters*, 62(4):183 – 188, 1997.

[34] Wan Fokkink and Rob J. van Glabbeek. Ntyft/ntyxt rules reduce to ntree rules. *Inf. Comput.*, 126(1):1–10, 1996.

[35] Guillaume Gardey, John Mullins, and Olivier H. Roux. Non-interference control synthesis for security timed automata. *Electr. Notes Theor. Comput. Sci.*, 180(1):35–53, 2007.

[36] Daniel Gebler and Wan Fokkink. Compositionality of probabilistic hennessy-milner logic through structural operational semantics. In *Proceedings of the 23rd international conference on Concurrency Theory*, CONCUR'12, pages 395–409, Berlin, Heidelberg, 2012. Springer-Verlag.

[37] Rob J. van Glabbeek. The linear time - branching time spectrum II. In Best [14], pages 66–81.

[38] Rob J. van Glabbeek. The linear time-branching time spectrum I - the semantics of concrete, sequential processes. In *Handbook of Process Algebra, chapter 1*, pages 3–99. Elsevier, 2001.

[39] Rob J. van Glabbeek. The meaning of negative premises in transition system specifications II. *Journal of Logic and Algebraic Programming*, 60-61:229–258, 2004.

[40] Rob J. van Glabbeek, Scott A. Smolka, and Bernhard Steffen. Reactive, generative and stratified models of probabilistic processes. *Inf. Comput.*, 121(1):59–80, 1995.

[41] Joseph A. Goguen and José Meseguer. Security Policies and Security Models. In *Proceedings of the 1982 IEEE Symposium on Security and Privacy*, pages 11–20. IEEE Computer Society, April 1982.

[42] Jan Friso Groote. *Process Algebra and Structured Operational Semantics*. Centrum voor Wiksunde en Informatica, 1991.

[43] Jan Friso Groote. Transition system specifications with negative premises. *Theor. Comput. Sci.*, 118(2):263–299, 1993.

[44] Jan Friso Groote and Frits Vaandrager. Structured operational semantics and bisimulation as a congruence. *Inf. Comput.*, 100(2):202–260, 1992.

[45] Hans A. Hansson and Lars-ake Fredlund. *Time and Probability in Formal Design of Distributed Systems*. Real-time safety critical systems. Elsevier, 1994.

[46] Seyyed Vahid Hashemian and Farhad Mavaddat. A graph-based approach to web services composition. In *Proceedings of the The 2005 Symposium on Applications and the Internet*, SAINT '05, pages 183–189, Washington, DC, USA, 2005. IEEE Computer Society.

[47] C. A. R. Hoare. Communicating sequential processes. *Commun. ACM*, 21(8):666–677, 1978.

[48] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[49] Claude Jard and Thierry Jéron. TGV: theory, principles and algorithms: A tool for the automatic synthesis of conformance test cases for non-deterministic reactive systems. *Int. J. Softw. Tools Technol. Transf.*, 7(4):297–315, August 2005.

[50] Bengt Jonsson and Joachim Parrow, editors. *CONCUR '94, Concurrency Theory, 5th International Conference, Uppsala, Sweden, August 22-25, 1994, Proceedings*, volume 836 of *Lecture Notes in Computer Science*. Springer, 1994.

[51] Richard M. Karp and Vijaya Ramachandran. Handbook of theoretical computer science (vol.A). chapter Parallel algorithms for shared-memory machines, pages 869–941. MIT Press, Cambridge, MA, USA, 1990.

[52] K. Khan, Jun Han, and Yuliang Zheng. A framework for an active interface to characterise compositional security contracts of software components. In *Software Engineering Conference, 2001. Proceedings.*, pages 117 –126, 2001.

[53] Ruggero Lanotte and Simone Tini. Probabilistic bisimulation as a congruence. *ACM Trans. Comput. Logic*, 10(2):1–48, 2009.

[54] Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. In *POPL*, pages 344–352. ACM Press, 1989.

[55] Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. *Inf. Comput.*, 94(1):1–28, 1991.

[56] Matias David Lee and Pedro R. D'Argenio. Describing secure interfaces with interface automata. *Electr. Notes Theor. Comput. Sci.*, 264(1):107–123, 2010.

[57] Matias David Lee and Pedro R. D'Argenio. A refinement based notion of non-interference for interface automata: Compositionality, decidability and synthesis. In *SCCC 2010, Proceedings of the XXIX International Conference of the Chilean Computer Science Society*, pages 280–289, 2010.

[58] Matias David Lee and Pedro R. D'Argenio. Semantics for interactive sequential systems and non-interference properties. *CLEI Electron. J.*, 14(3), 2011.

[59] Matias David Lee, Daniel Gebler, and Pedro R. D'Argenio. Tree rules in probabilistic transition system specifications with negative and quantitative premises. In Bas Luttik and Michel A. Reniers, editors, *DCM*, volume 89 of *EPTCS*, pages 115–130, 2012.

[60] Daryl McCullough. A hookup theorem for multilevel security. *IEEE Trans. Softw. Eng.*, 16(6):563–568, June 1990.

[61] Ron van der Meyden and Chenyi Zhang. A comparison of semantic models for noninterference. *Theoretical Computer Science*, 411(47):4123 – 4147, 2010.

[62] Robin Milner. *Communication and concurrency*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.

[63] Mohammad Reza Mousavi, Michel A. Reniers, and Jan Friso Groote. SOS formats and meta-theory: 20 years after. *Theor. Comput. Sci.*, 373(3):238–272, 2007.

[64] Neda Noroozi, Ramtin Khosravi, Mohammad Reza Mousavi, and Tim A. C. Willemse. Synchronizing asynchronous conformance testing. In *Proceedings of the 9th international conference on Software engineering and formal methods*, SEFM'11, pages 334–349, Berlin, Heidelberg, 2011. Springer-Verlag.

[65] Gordon D. Plotkin. The origins of structural operational semantics. *J. Log. Algebr. Program.*, 60-61:3–15, 2004.

[66] Gordon D. Plotkin. A structural approach to operational semantics. *J. Log. Algebr. Program.*, 60-61:17–139, 2004.

[67] Amir Pnueli and Lenore D. Zuck. Probabilistic verification. *Inf. Comput.*, 103(1):1–29, 1993.

[68] A. William Roscoe. CSP and determinism in security modelling. *IEEE Symposium on Security and Privacy*, pages 114–127, 1995.

[69] Peter Y. A. Ryan. Mathematical models of computer security. In *Revised versions of lectures given during the IFIP WG 1.7 International School on Foundations of Security Analysis and Design on Foundations of Security Analysis and Design: Tutorial Lectures*, FOSAD '00, pages 1–62, London, UK, 2001. Springer.

[70] Davide Sangiorgi. *Introduction to Bisimulation and Coinduction*. Cambridge University Press, 2011.

[71] Roberto Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, MIT, 1995.

[72] Roberto Segala and Nancy A. Lynch. Probabilistic simulations for probabilistic processes. In Jonsson and Parrow [50], pages 481–496.

[73] Simone Tini. Non-expansive epsilon-bisimulations for probabilistic processes. *Theor. Comput. Sci.*, 411(22-24):2202–2222, 2010.

[74] Jan Tretmans. Conformance testing with labelled transition systems: Implementation relations and test generation. *Computer Networks and ISDN Systems*, 29(1):49–79, 1996.

[75] Jan Tretmans. Model based testing with labelled transition systems. In Robert M. Hierons, Jonathan P. Bowen, and Mark Harman, editors, *Formal Methods and Testing*, volume 4949 of *Lecture Notes in Computer Science*, pages 1–38. Springer, 2008.

[76] Chris Verhoef. A congruence theorem for structured operational semantics with predicates and negative premises. In Jonsson and Parrow [50], pages 433–448.

[77] D.J. Walker. Automated analysis of mutual exclusion algorithms using CCS. *Formal Aspects of Computing*, 1:273–292, 1989.

[78] Peter Wegner. Why interaction is more powerful than algorithms. *Commun. ACM*, 40(5):80–91, May 1997.

[79] Lingyun Yang and Luoshan Xu. Algebraic aspects of generalized approximation spaces. *Int. J. Approx. Reasoning*, 51(1):151–161, 2009.