Universidad
Nacional
de Córdoba

Facultad de Matemática, Astronomía,
Física y Computación

Trabajo especial de la Licenciatura en
Ciencias de la Computación

Algoritmos para decidir definibilidad de relaciones en fragmentos de primer orden

Autor:
Pablo Ventura

Director:
Dr. Miguel Campercholi

Febrero de 2016

Abstract

The definibility of a relation for a class of structures, for some first order fragment, holds a semantic notion associated with its bases in preservation properties of differents kinds of morphisms. In this work, we propose algorithms to decide definibility in an automatic way; also, algorithms to generate the algebra of definable relations for a class of structures are provided. Beside presenting the algorithms, we'll introduce the package Definibility, used to implement the algorithms. Documentations of its use will be provided with some examples. Lastly, a theorem about the characterization of binary definable relations by positive existentials on distributive lattices will be proved. The theorem was inspired with the use of the package.

Resumen

La definibilidad de una relación en una clase de estructuras para algún fragmento de primer orden tiene una noción semántica asociada fundamentada en propiedades de preservación de diferentes tipos de morfismos. Este trabajo propone algoritmos para decidir definibilidad automáticamente como así también algoritmos para generar el álgebra de relaciones definibles para una clase de estructuras. Además de presentar los algoritmos, se presenta *Definability*, un paquete que implementa los algoritmos y se documenta su uso con algunos ejemplos. Luego se prueba un teorema sobre la caracterización de las relaciones binarias definibles por existenciales positivas en reticulados distributivos, que fue inspirado con la utilización de nuestro paquete.

Agradecimientos

A Inés, por ayudarme tanto y soportar los fines de semana de estudio,
a mi familia, por el apoyo y el incentivo,
a mis amigos, dentro y fuera de la facultad,
a Camper, por el entusiasmo y la confianza,
y a todo FaMAF, por la emoción de cada clase.

Índice general

Capítulo 1

Introducción

En nuestra experiencia, los buenos algoritmos son descubiertos al hacer un análisis formal y/o matemático del problema a resolver. En este caso nos proponemos el desarrollo de algoritmos para resolver un problema básico de la teoría de modelos finitos: decidir definibilidad de relaciones en el lenguaje de primer orden. La motivación inicial es la explotación de resultados teóricos del artículo [Campercholi y Vaggione 2015] que caracterizan la definibilidad en términos semánticos.

1.1. El problema

Sean $\mathcal{L}$ un lenguaje de primer orden y $\mathcal{K}$ una clase de $\mathcal{L}$-estructuras. Dado un símbolo de relación n-ario $R \in \mathcal{L}$ diremos que la fórmula $\varphi(x_1, \ldots, x_n)$ *define* a R en $\mathcal{K}$ si

$$\mathcal{K} \vDash \forall \overline{x}\ R(\overline{x}) \leftrightarrow \varphi(\overline{x}).$$

El problema computacional estudiado en esta monografía es el siguiente.

Problema 1. Dados:

- $\mathcal{L}$ un lenguaje de primer orden finito,
- $\mathcal{K}$ un conjunto finito de estructuras finitas de $\mathcal{L}$,
- $R \in \mathcal{L}$ un símbolo de relación, y
- Σ un conjunto recursivo de fórmulas de $\mathcal{L} - \{R\}$,

decidir si hay una fórmula de Σ que define a R en $\mathcal{K}$.

La tipología de este problema sugiere que su complejidad computacional en el caso general será muy elevada. Una solución por «fuerza bruta» requiere la construcción de todas las relaciones Σ-definibles en $\mathcal{K}$ para poder dar una respuesta negativa. Nuestro enfoque es evitar este costo explotando propiedades de preservación que pueda tener el conjunto Σ.

En el trabajo [Campercholi y Vaggione 2015] se presentan condiciones semánticas equivalentes a la existencia de una $\mathcal{L}$-fórmula $\varphi(\bar{x})$ que define a R en $\mathcal{K}$, donde φ pertenece a un fragmento de primer orden especifico, como las fórmulas abiertas, las abiertas positivas, las existenciales, etc.

Por ejemplo, en el caso de la definibilidad por una fórmula abierta (i.e. fórmulas sin cuantificadores) el trabajo establece que (cf. 14) dado $\mathcal{L}$ un lenguaje de primer orden, $R \in \mathcal{L}$ un símbolo de relación n-ario y $\mathbf{A}$ una $\mathcal{L}$-estructura finita, los siguientes son equivalentes:

1. Hay una fórmula abierta que define R en $\mathbf{A}$.

2. Todo isomorfismo $\sigma : \boldsymbol{A}_0 \to \boldsymbol{B}_0$ donde $\mathbf{A}_0$ y $\mathbf{B}_0$ son subestructuras en $\mathcal{L} - \{R\}$ de $\mathbf{A}$, preserva R (i.e. si $(a_1, \ldots, a_n) \in R$, entonces $(\sigma(a_1), \ldots, \sigma(a_n)) \in R$).

Al utilizar este resultado teórico, decidir definibilidad se convierte en un problema de búsqueda y chequeo de morfismos entre estructuras, potencialmente más tratable en su costo computacional.

1.1.1. Ejemplo

Consideremos en el reticulado $\mathbf{2} \times \mathbf{2}$ la relación unaria P dada por el siguiente diagrama. ¿Es P definible por una fórmula abierta en $\mathbf{2} \times \mathbf{2}$?

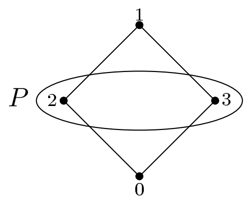


Figura 1.1.1: Reticulado $\mathbf{2} \times \mathbf{2}$ con una relación unaria P

Notar que entre el subreticulado con universo $\{1\}$ y el subreticulado con universo $\{2\}$ hay un isomorfismo σ que no preserva R, ya que $1 \in R$ pero $\sigma(1) = 2 \notin R$. A la luz del resultado mencionado anteriormente esto nos permite asegurar que no existe una fórmula abierta (del lenguaje de los reticulados) que defina la relación P en $\mathbf{2} \times \mathbf{2}$.

1.2. La solución desarrollada

En vista de lo expuesto anteriormente nuestro enfoque consiste en cambiar el costo computacional de construir las relaciones definibles sobre una clase de estructuras por el costo de computar una familia adecuada de morfismos y verificar que estos preservan la relación a definir. Sin dudas el mayor peso computacional de este enfoque está en encontrar la familia de morfismos. Por esta razón nuestra investigación se centró en estudiar el problema de cómo reducir el número de morfismos a construir. Los resultados obtenidos en esta dirección (REFERENCIAS CRUZADAS) se basan en encontrar una subconjunto *generador* de la familia de morfismos a calcular.

Otra característica importante de las algoritmos desarrollados, en este caso en relación a su implementación, es que la construcción de morfismos se lleva a cabo como una instancia del *Constraint Satisfaction Problem* (CSP) (CITA). Esto nos permite usufructuar la eficiencia alcanzada por los resolvedores de CSP, y hace muy promisoria la especialización del Problema 1 a clases de estructuras en las cuales el CSP para cómputo de morfismos está bien condicionado (véase, por ejemplo, [Idziak y col. 2010]).

1.3. Estructura del trabajo

El trabajo se encuentra organizado de la siguiente forma:

En el Capítulo 2 se prueban algunos lemas fundamentales para luego probar los teoremas de definibilidad de relaciones por fórmulas de primer orden, abiertas, abiertas positivas, existenciales, existenciales positivas, conjunción de atómicas y fórmulas primitivas positivas.

En el Capítulo 3, se encuentran los algoritmos para chequeo de definibilidad y los resultados teóricos que fundamentan sobre el subconjunto de morfismos que genera por composición a todo el conjunto de morfismos a chequear. Luego de los algoritmos mostramos cómo modelizar la búsqueda de homomorfismos mediante un CSP, y finalizamos el capítulo con la generación de subestructuras.

En el Capítulo 4, presentamos los algoritmos para generar el álgebra de relaciones definibles de una estructura. Luego mostramos algunos ejemplos mediante el uso de nuestro paquete *Definability* sobre reticulados. Finalizamos el capítulo con un teorema que caracteriza las relaciones binarias definibles por existenciales positivas en los reticulados distributivos, inspirado en los ejemplos probados sobre el paquete.

En el Capítulo 5, presentamos nuestro paquete *Definability* para el software matemático SageMath, documentando su instalación y su uso.

En el Capítulo 6, se exponen nuestras conclusiones y las posibles continuaciones a este trabajo.

Capítulo 2

Teoremas de definibilidad

En esta sección presentamos resultados que caracterizan la definibilidad, en diferentes fragmentos de primer orden, de relaciones sobre familias de estructuras. Estas caracterizaciones reducen la pregunta de si una relación es definible a la pregunta de si dicha relación es preservada por una colección de morfismos que depende del fragmento de primer orden considerado. Por ejemplo, (cf. 14) dados $\mathcal{L}$ un lenguaje de primer orden, $R \in \mathcal{L}$ un símbolo de relación n-ario y $\mathbf{A}$ una $\mathcal{L}$-estructura, entonces hay una fórmula abierta que define R en $\mathbf{A}$, sii todo isomorfismo $\sigma : \boldsymbol{A}_0 \to \boldsymbol{B}_0$ donde $\mathbf{A}_0$ y $\mathbf{B}_0$ son subestructuras en $\mathcal{L} - \{R\}$ de $\mathbf{A}$, preserva R (i.e. si $(a_1, \ldots, a_n) \in R$, entonces $(\sigma(a_1), \ldots, \sigma(a_n)) \in R$).

Estos resultados, expuestos originalmente en [Campercholi y Vaggione 2015], constituyen el punto de partida para nuestro estudio del chequeo computacional de la definibilidad. Los algoritmos desarrollados pueden encontrarse en los capítulos siguientes.

2.1. Preliminares

Si bien la intención ha sido la de escribir un trabajo autocontenido, suponemos que el lector está familiarizado con los conceptos básicos de la lógica de primer orden y su teoria de modelos elemental. Textos clásicos de referencia en estos temas son por ejemplo (CITAS).

Una fórmula es *abierta* si no tiene ocurrencias de $\forall$, $\exists$. Es *positiva* si no tiene ocurrencias de $\neg$, $\to$, $\leftrightarrow$. Una fórmula es *existencial* si es de la forma $\exists \bar{x}\, \varphi(\bar{x}, \bar{y})$, con $\varphi(\bar{x}, \bar{y})$ una fórmula abierta, mientras que es *existencial positiva* si $\varphi(\bar{x}, \bar{y})$ es abierta y positiva. Una fórmula es *conjunción de atómicas* si es de la forma $\varphi_1(\bar{x}) \wedge \cdots \wedge \varphi_n(\bar{x})$, donde cada $\varphi_i \in \mathrm{At}(\mathcal{L})$. Una fórmula es *primitiva positiva* si es de la forma $\exists \bar{x}\, \varphi(\bar{x}, \bar{y})$, con $\varphi(\bar{x}, \bar{y})$ conjunción de atómicas.

Introducimos ahora la notación que utilizaremos para los fragmentos de primer orden que serán estudiados en este trabajo.

$$\mathrm{Fo}(\mathcal{L}) = \{\varphi : \varphi \text{ es una } \mathcal{L}\text{-fórmula}\},$$

$$\mathrm{Op}(\mathcal{L}) = \{\varphi : \varphi \text{ es una } \mathcal{L}\text{-fórmula abierta}\},$$

$$\mathrm{Op}^{+}(\mathcal{L}) = \{\varphi : \varphi \text{ es una } \mathcal{L}\text{-fórmula abierta positiva}\},$$

$$\mathrm{E}(\mathcal{L}) = \{\varphi : \varphi \text{ es una } \mathcal{L}\text{-fórmula existencial}\},$$

$$\mathrm{E}^{+}(\mathcal{L}) = \{\varphi : \varphi \text{ es una } \mathcal{L}\text{-fórmula existencial positiva}\},$$
$$\pm\mathrm{At}(\mathcal{L}) = \mathrm{At}(\mathcal{L}) \cup \{\neg\alpha : \alpha \in \mathrm{At}(\mathcal{L})\},$$
$$\mathrm{At}(\mathcal{L}) = \{\mathcal{L}\text{-fórmulas atómicas}\}.$$

A continuación definimos dos nociones centrales del presente trabajo.

Definición 2. Sean $\mathcal{L}$ un lenguaje de primer orden y $\mathcal{K}$ una clase de $\mathcal{L}$-estructuras. Si $R \in \mathcal{L}$ es un símbolo de relación n-ario, diremos que la fórmula $\varphi(x_1, \ldots, x_n)$ *define a* R *en* $\mathcal{K}$ si para todo $\mathbf{A} \in \mathcal{K}$ y todo $a_1, \ldots, a_n \in A$ vale que

$$(a_1, \ldots, a_n) \in R^{\mathbf{A}} \iff \mathbf{A} \vDash \varphi(a_1, \ldots, a_n).$$

Definición 3. Sean $\mathcal{L}$ un lenguaje de primer orden, $R \in \mathcal{L}$ un símbolo de relación n-ario y $\mathbf{A}$,$\mathbf{B}$ dos $\mathcal{L}$-estructuras. Diremos que una función $f : A \to B$ *preserva* a R si para toda tupla $(a_1, \ldots, a_n) \in A$ tenemos que si $(a_1, \ldots, a_n) \in R^{\mathbf{A}}$ implica que $(f(a_1), \ldots, f(a_n)) \in R^{\mathbf{B}}$. Es decir, f es un homomorfismo de $\langle A, R^{\mathbf{A}} \rangle$ en $\langle B, R^{\mathbf{B}} \rangle$.

Dado un conjunto de fórmulas Δ, escribiremos $\Delta(\bar{x})$ para anunciar que cada una de las fórmulas en Δ tiene sus variables libres contenidas en la tupla $\bar{x}$. Si $\mathbf{A}$ es una estructura y $\bar{a}$ es una tupla de elementos de A, escribiremos $\mathbf{A} \vDash \Delta[\bar{a}]$ cuando $\mathbf{A} \vDash \delta[\bar{a}]$ para cada $\delta \in \Delta(\bar{x})$.

Dados $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden y $\mathbf{A}$ una $\mathcal{L}'$-estructura, usaremos $\mathbf{A}_{\mathcal{L}}$ para indicar el reducto de $\mathbf{A}$ al lenguaje $\mathcal{L}$. Si $\mathbf{A}, \mathbf{B}$ son $\mathcal{L}$-estructuras, escribimos $\mathbf{A} \leq \mathbf{B}$ para expresar que $\mathbf{A}$ es subestructura de $\mathbf{B}$. Dada una estructura $\mathbf{A}$ y $\bar{a} = a_1, \ldots, a_n \in A$ utilizaremos $\langle \bar{a} \rangle^{\mathbf{A}}$ para denotar la subestructura de $\mathbf{A}$ generada por $\bar{a}$.

Definición 4. Dos fórmulas $\alpha(\bar{x})$ y $\beta(\bar{x})$ se dicen *equivalentes* sobre una familia de estructuras $\mathcal{K}$ si para cada $\mathbf{A} \in \mathcal{K}$ y cada $\bar{a}$ de A vale que

$$\mathbf{A} \vDash \alpha[\bar{a}] \iff \mathbf{A} \vDash \beta[\bar{a}].$$

El siguiente teorema será utilizado para demostrar todos las caracterizaciones de definibilidad; es la herramienta que nos permitirá transformar un diagrama infinitario en una fórmula.

Teorema 5. *Sea $\mathcal{K}$ un conjunto finito de estructuras finitas. Hay un conjunto finito de fórmulas $\Sigma(x_1, \ldots, x_n)$ tal que para toda fórmula $\varphi(\overline{x})$ hay $\sigma(\overline{x}) \in \Sigma(\overline{x})$ tal que $\varphi(\bar{x})$ y $\sigma(\overline{x})$ son equivalentes sobre $\mathcal{K}$.*

Demostración. Supongamos $\mathcal{K} = \{\mathbf{A}_1, \ldots, \mathbf{A}_m\}$. Para un fórmula $\varphi(\bar{x})$ sea

$$[\varphi(\bar{x})]^{\mathcal{K}} = \{\bar{a} \in A_1^n \mid \mathbf{A}_1 \vDash \varphi[\bar{a}]\} \times \cdots \times \{\bar{a} \in A_m^n \mid \mathbf{A}_m \vDash \varphi[\bar{a}]\}.$$

Notar que $\varphi(\bar{x})$ es equivalente $\psi(\bar{x})$ en $\mathcal{K}$ sii $[\varphi(\bar{x})]^{\mathcal{K}} = [\psi(\bar{x})]^{\mathcal{K}}$. Luego el número de clases de equivalencia es finito ya que está acotado por $|\mathcal{P}(A_1^n \times \cdots \times A_m^n)|$. □

El siguiente lema establece que los embeddings preservan las formulas abiertas.

Lema 6 (Los embeddings preservan fórmulas abiertas)**.** *Sean* $\mathbf{A}$*,* $\mathbf{B}$ *estructuras y* $\gamma : \mathbf{A} \to \mathbf{B}$ *una función. Son equivalentes:*

1. *γ es un embedding de $\mathbf{A}$ en $\mathbf{B}$.*

2. *Para toda fórmula abierta $\varphi(\bar{x})$ y para cada $\bar{a}$ de A vale que:*

$$\mathbf{A} \vDash \varphi[\bar{a}] \Leftrightarrow \mathbf{B} \vDash \varphi[\gamma(\bar{a})].$$

Demostración. 1⇒2) Sea γ un embedding de $\mathbf{A}$ en $\mathbf{B}$, sea $\varphi(x_1, ..., x_n)$ una fórmula abierta y $\bar{a} \in A^n$. El caso base es directo ya que los homomorfismos preservan términos y relaciones. Veamos los casos inductivos:

Sea $\varphi(\bar{x}) = \neg\varphi_1(\bar{x})$:

$$\mathbf{A} \vDash \neg\varphi_1[\bar{a}] \text{ sii } \mathbf{A} \nvDash \varphi_1[\bar{a}] \text{ sii } \mathbf{B} \nvDash \varphi_1[\gamma(\bar{a})] \text{ sii } \mathbf{B} \vDash \neg\varphi_1[\gamma(\bar{a})].$$

Sea $\varphi(\bar{x}) = (\varphi_1 \eta \varphi_2)(\bar{x})$:

$$\begin{aligned} \mathbf{A} \vDash (\varphi_1\eta\varphi_2)[\bar{a}] &\text{ sii } \mathbf{A} \vDash \varphi_1[\bar{a}]\text{"}\eta\text{"}\mathbf{A} \vDash \varphi_2[\bar{a}] \\ \text{sii } \mathbf{A} \vDash \varphi_1[\gamma(\bar{a})]\text{"}\eta\text{"}\mathbf{B} \vDash \varphi_2[\gamma(\bar{a})] &\text{ sii } \mathbf{B} \vDash (\varphi_1\eta\varphi_2)[\gamma(\bar{a})]. \end{aligned}$$

2⇒1) Supongamos que para toda fórmula abierta $\varphi = \varphi(\bar{x})$ y cada $\bar{a} \in A^m$ vale que:

$$\mathbf{A} \vDash \varphi[\bar{a}] \Leftrightarrow \mathbf{B} \vDash \varphi[\gamma(\bar{a})].$$

Es rutina ver que γ es homomorfismo.

- Veamos que γ es inyectiva:

$$\begin{aligned} &\gamma(a) = \gamma(a') \\ &\quad \mathbf{B} \vDash (x_1 \equiv x_2)[\gamma(a), \gamma(a')] \\ &\quad \mathbf{A} \vDash (x_1 \equiv x_2)[a, a'] \\ &\quad a = a'. \end{aligned}$$

- Veamos que γ es embedding:

  Sea $r \in \mathcal{R}_n$

$$\begin{aligned} &(\gamma(a_1), \ldots, \gamma(a_n)) \in r^{\mathbf{B}} \\ &\qquad \mathbf{B} \vDash r(x_1, \ldots, x_n)[\gamma(a_1), \ldots, \gamma(a_n)] \\ &\qquad \mathbf{A} \vDash r(x_1, \ldots, x_n)[a_1, \ldots, a_n] \\ &(a_1, \ldots, a_n) \in r^{\mathbf{A}}. \end{aligned}$$

□

El siguiente lema determina que las subestructuras preservan formulas abiertas.

Lema 7. *Si $\mathbf{A}$ es una subestructura de $\mathbf{B}$ y $\varphi(\bar{x})$ es una fórmula abierta, entonces para cada $\bar{a} \in A^n$ vale que*

$$\mathbf{A} \vDash \varphi[\bar{a}] \Leftrightarrow \mathbf{B} \vDash \varphi[\bar{a}].$$

Demostración. Notar que la inclusión $\iota : A \to B$, definida por $\iota(a) = a$, es un embedding de $\mathbf{A}$ en $\mathbf{B}$. Luego una aplicación directa del Lema 6, produce el resultado deseado. $\square$

Definimos a continuación el diagrama abierto, que será utilizado en la construcción de formulas abiertas que definen relaciones.

Definición 8. Sea $\mathbf{A}$ una estructura y sean $a_1, \ldots, a_n \in A$. Definimos el *diagrama abierto* para $a_1, \ldots, a_n$ en $\mathbf{A}$ como:

$$\Delta_{\mathbf{A},\bar{a}}(x_1, \ldots, x_n) := \{\alpha(\bar{x}) \mid \alpha \in \mathrm{Op}(\mathcal{L}) \text{ y } \mathbf{A} \vDash \alpha[\bar{a}]\}.$$

El siguiente lema establece una relación muy interesante entre elementos de dos estructuras al mostrar que si dos secuencias de elementos tienen las mismas propiedades abiertas (i.e., el mismo diagrama abierto) significa que generan subestructuras isomorfas.

Lema 9. *Sea $\mathbf{A}$ una estructura y $b_1, \ldots, b_n \in B$, son equivalentes:*

1. $\mathbf{B} \vDash \Delta_{\mathbf{A},\bar{a}}\left[\bar{b}\right]$,
2. *Hay un isomorfismo γ de $\langle \bar{a} \rangle^{\mathbf{A}}$ en $\left\langle \bar{b} \right\rangle^{\mathbf{B}}$ tal que $\gamma(\bar{a}) = \bar{b}$.*

Demostración. Veamos que 1⇒2. Supongamos que $\mathbf{B} \vDash \Delta_{\mathbf{A},\bar{a}}\left[\bar{b}\right]$. Es decir que si $\alpha(\bar{x})$ es una fórmula abierta y $\mathbf{A} \vDash \alpha[\bar{a}]$, luego $\mathbf{B} \vDash \alpha\left[\bar{b}\right]$. Notar que

$$\begin{aligned} &\gamma : \langle \bar{a} \rangle^{\mathbf{A}} \to \left\langle \bar{b} \right\rangle^{\mathbf{A}} \\ &\gamma(t^{\mathbf{A}}[a_1, \ldots, a_n]) = t^{\mathbf{B}}[b_1, \ldots, b_n] \end{aligned}$$

es un función bien definida que cumple $\gamma(\bar{a}) = \bar{b}$. Además, por hipótesis preserva fórmulas abiertas. Así, por el Lema 6, sabemos que γ es un embedding. Por último, es evidente de la definición queγ es sobreyectiva.
La implicación 2⇒1 es consecuencia directa del Lema 6. $\square$

En el lema siguiente se determina que los homomorfismos preservan formulas atómicas, y también formulas abiertas positivas.

Lema 10. *Sean $\mathbf{A}, \mathbf{B}$ estructuras y $h : A \to B$ una función, son equivalentes:*

1. *h es un homomorfismo de $\mathbf{A}$ en $\mathbf{B}$.*
2. *Para toda fórmula atómica $\varphi(\bar{x})$ y para cada $\bar{a}$ de A vale que*
$$\mathbf{A} \vDash \varphi[\bar{a}] \Longrightarrow \mathbf{B} \vDash \varphi[h(\bar{a})].$$
3. *Para toda fórmula abierta positiva $\varphi(\bar{x})$ y para cada $\bar{a}$ de A vale que*
$$\mathbf{A} \vDash \varphi[\bar{a}] \Longrightarrow \mathbf{B} \vDash \varphi[h(\bar{a})].$$

Demostración. Rutina. $\square$

Ahora definimos el diagrama atómico positivo. Éste se utilizará para la construcción de fórmulas positivas.

Definición 11. Sea $\mathbf{A}$ una estructura y sean $a_1, \ldots, a_n \in A$. Definimos el diagrama atómico positivo de $\bar{a}$ en $\mathbf{A}$ como

$$\Delta^+_{\mathbf{A},\bar{a}}(x_1, \ldots, x_n) := \{\alpha \mid \alpha \in \mathrm{At}(\mathcal{L}) \text{ y } \mathbf{A} \vDash \alpha[\bar{a}]\}.$$

De manera análoga al Lema 9, si dos secuencias de elementos en dos estructuras satisfacen las mismas propiedades atómicas positivas, hay un homomorfismo entre las subestructuras generadas.

Teorema 12. *Sean $\mathbf{B}$ una estructura y $b_1, \ldots, b_n \in B$, son equivalentes:*

1. $\mathbf{B} \vDash \Delta^+_{\mathbf{A},\bar{a}}[\bar{b}]$.

2. *Hay un homomorfismo h de $\langle \bar{a} \rangle^{\mathbf{A}}$ en $\langle \bar{b} \rangle^{\mathbf{B}}$ tal que $h(\bar{a}) = \bar{b}$.*

Demostración. Esta prueba es muy similar a la del Lema 9. □

2.2. Definibilidad por fórmulas abiertas

Ahora estableceremos la noción semántica para la definibilidad abierta.

En el siguiente lema, construimos una fórmula equivalente al diagrama abierto para una secuencia de elementos en una estructura, que utilizaremos en el teorema de definibilidad abierta.

Lema 13. *Sean $\mathbf{A}, \mathbf{B}$ estructuras finitas y $a_1, \ldots, a_n \in A$, entonces existe una fórmula abierta $\varphi(\bar{x})$ tal que para todo $\bar{b} \in B^n$ son equivalentes:*

1. $\mathbf{B} \vDash \varphi[\bar{b}]$

2. *Hay un isomorfismo γ de $\langle \bar{a} \rangle^{\mathbf{A}}$ en $\langle \bar{b} \rangle^{\mathbf{B}}$ tal que $\gamma(\bar{a}) = \bar{b}$*

Demostración. Tomamos

$$\varphi = \bigwedge_{\alpha \in \Delta_{\mathbf{A}\bar{a}}} \alpha(\bar{x})$$

que es una fórmula, por el Teorema 5.

Es claro que $\mathbf{B} \vDash \bigwedge_{\alpha \in \Delta_{\mathbf{A}\bar{a}}} \alpha(\bar{x})[\bar{b}] \Leftrightarrow \mathbf{B} \vDash \Delta_{\mathbf{A}\bar{a}}[\bar{b}]$

Luego por Lema 9, hay isomorfismo $\gamma : \langle \bar{a} \rangle^{\mathbf{A}} \to \langle \bar{b} \rangle^{\mathbf{B}}$ tal que $\gamma(\bar{a}) = \bar{b}$ □

Como determina el siguiente teorema, la definibilidad abierta depende de la preservación en los isomorfismos entre subestructuras de las estructuras de $\mathcal{K}$.

Teorema 14. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden, sea $R \in \mathcal{L}' - \mathcal{L}$ un símbolo de relación n-ario. Para una clase finita $\mathcal{K}$ de $\mathcal{L}'$-estructuras, los siguientes son equivalentes:*

1. Hay una fórmula en $\mathrm{Op}(\mathcal{L})$ que define R en $\mathcal{K}$.

2. Para todas $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{K}$, todas $\boldsymbol{A}_0 \leq \boldsymbol{A}_{\mathcal{L}}$,$\boldsymbol{B}_0 \leq \boldsymbol{B}_{\mathcal{L}}$, todo isomorfismo $\sigma : \boldsymbol{A}_0 \to \boldsymbol{B}_0$ preserva R.

Demostración. 1⇒2) Sea $\varphi(\bar{x})$ la fórmula que define R en $\mathcal{K}$, sean $\mathbf{A}, \mathbf{B} \in \mathcal{K}$ y sean $\mathbf{A}_0 \leq \mathbf{A}_{\mathcal{L}}$ y $\mathbf{B}_0 \leq \mathbf{B}_{\mathcal{L}}$ tales que $\sigma : \mathbf{A}_0 \to \mathbf{B}_0$ sea un isomorfismo. Sea $\bar{a} \in R^{\mathbf{A}_0} \cap A_0$ veamos que $\sigma(\bar{a}) \in R^{\mathbf{B}_0}$. Como $\bar{a} \in R^{\mathbf{A}_0} \Rightarrow \bar{a} \in R^{\mathbf{A}} \Leftrightarrow \mathbf{A} \vDash \varphi[\bar{a}]$, por Lema 7, $A_0 \vDash \varphi[\bar{a}]$ y como $\mathbf{A}_0$ y $\mathbf{B}_0$ son isomorfos por σ, $\mathbf{B}_0 \vDash \varphi[\sigma(\bar{a})]$. Nuevamente por Lema 7 $\mathbf{B} \vDash \varphi[\sigma(\bar{a})]$ y como por hipótesis φ define a R en $\mathcal{K}$, $\sigma(\bar{a}) \in R^{\mathbf{B}}$.

2⇒1) Sea

$$\varphi = \bigvee_{\mathbf{A}\in\mathcal{K}} \left(\bigvee_{\bar{a}\in R^{\mathbf{A}}} \left(\bigwedge_{\alpha\in\Delta_{\mathbf{A}\bar{a}}} \alpha(\bar{x}) \right) \right),$$

la cual es fórmula por el Teorema 5. Veamos que para cada $\mathbf{B} \in \mathcal{K}$, $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$ sii $\bar{b} \in R^{\mathbf{B}}$. Sea $\mathbf{B} \in \mathcal{K}$:

⇒) Supongamos $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$, entonces en particular para algún $\mathbf{A} \in \mathcal{K}$ y $\bar{a} \in R^{\mathbf{A}}$,

$$\mathbf{B} \vDash \bigwedge_{\alpha\in\Delta_{\mathbf{A}\bar{a}}} \alpha(\bar{x})\left[\bar{b}\right].$$

Por Lema 13, hay un isomorfismo $\gamma : \langle\bar{a}\rangle^{\mathbf{A}} \to \left\langle\bar{b}\right\rangle^{\mathbf{B}}$ tal que $\gamma(\bar{a}) = \bar{b}$. Entonces por hipótesis γ preserva R. Finalmente como $\bar{a} \in R^{\mathbf{A}}$, luego $\gamma(\bar{a}) \in R^{\mathbf{B}}$ o sea que $\bar{b} \in R^{\mathbf{B}}$.

⇐) Supongamos $\bar{b} \in R^{\mathbf{B}}$. Entonces

$$\varphi = \bigvee_{\mathbf{A}\in\mathcal{K}} \left(\bigvee_{\bar{a}\in R^{\mathbf{A}}} \left(\bigwedge_{\alpha\in\Delta_{\mathbf{A}\bar{a}}} \alpha(\bar{x}) \right) \right) = \cdots \vee \cdots \vee \left(\bigwedge_{\alpha\in\Delta_{\mathbf{B}\bar{b}}} \alpha(\bar{x}) \right),$$

evidentemente $\mathbf{B} \vDash \Delta_{\mathbf{B}\bar{b}}\left[\bar{b}\right]$, luego $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$. □

2.3. Definibilidad por fórmulas abiertas positivas

En esta sección, determinaremos la noción semántica para la definibilidad abierta positiva.

En el siguiente lema, construimos una fórmula equivalente al diagrama atómico positivo para un conjunto de elementos en una estructura, que utilizaremos en el teorema de definibilidad abierta positiva.

Lema 15. *Sean $\mathbf{A}, \mathbf{B}$ estructuras finitas y $a_1, \ldots, a_n an \in A$, entonces existe una fórmula abierta $\varphi(\bar{x})$ tal que para todo $\bar{b} \in B^n$ son equivalentes:*

1. $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$

2. *Hay un homomorfismo h de $\langle\bar{a}\rangle^{\mathbf{A}}$ en $\left\langle\bar{b}\right\rangle^{\mathbf{B}}$ tal que $h(\bar{a}) = \bar{b}$*

Demostración. Tomamos $\varphi = \bigwedge_{\alpha\in\Delta^{+}_{\mathbf{A}\bar{a}}} \alpha(\bar{x})$ que es una fórmula, por el Teorema 5.

Es claro que $\mathbf{B} \vDash \varphi\left[\bar{b}\right] \Leftrightarrow \mathbf{B} \vDash \Delta^{+}_{\mathbf{A}\bar{a}}\left[\bar{b}\right]$

Entonces por Lema 12, hay un homomorfismo h de $\langle\bar{a}\rangle^{\mathbf{A}}$ en $\left\langle\bar{b}\right\rangle^{\mathbf{B}}$ tal que $h(\bar{a}) = \bar{b}$ □

El siguiente teorema establece que la definibilidad abierta positiva depende de la preservación en los homomorfismos entre subestructuras de las estructuras de $\mathcal{K}$. Notar que esta condición incluye a los isomorfismos entre subestructuras de estructuras de $\mathcal{K}$, chequeados para el Teorema 14.

Teorema 16. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden, sea $R \in \mathcal{L}' - \mathcal{L}$ un símbolo de relación n-ario. Para una clase finita $\mathcal{K}$ de $\mathcal{L}'$-estructuras, los siguientes son equivalentes:*

1. Hay una fórmula en $\mathrm{Op}^+(\mathcal{L})$ que define R en $\mathcal{K}$.

2. Para todas $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{K}$, todas $\boldsymbol{A}_0 \leq \boldsymbol{A}_{\mathcal{L}}$,$\boldsymbol{B}_0 \leq \boldsymbol{B}_{\mathcal{L}}$, todo homomorfismo $h : \boldsymbol{A}_0 \to \boldsymbol{B}_0$ preserva R.

Demostración. Veamos 1⇒2:

Sea $\varphi(\bar{x})$ la fórmula abierta positiva que define R en $\mathcal{K}$, sean $\mathbf{A}, \mathbf{B} \in \mathcal{K}$ y sean $\mathbf{A_0} \leq \mathbf{A}_{\mathcal{L}}$ y $\mathbf{B}_0 \leq \mathbf{B}_{\mathcal{L}}$ tales que $h : \mathbf{A}_0 \to \mathbf{B}_0$ sea un homomorfismo. Sea $\bar{a} \in R^{\mathbf{A}_0} \cap A_0$ veamos que $h(\bar{a}) \in R^{\mathbf{B}_0}$:

$\bar{a} \in R^{\mathbf{A}_0} \Rightarrow \bar{a} \in R^{\mathbf{A}} \Leftrightarrow \mathbf{A} \vDash \varphi[\bar{a}]$, por Lema 7, $\mathbf{A}_0 \vDash \varphi[\bar{a}]$ y como h es un homomorfismo de $\mathbf{A}_0$ en $\mathbf{B}_0$ y φ es abierta positiva por Lema 10, $\mathbf{B}_0 \vDash \varphi[h(\bar{a})]$. Nuevamente por Lema 7 $\mathbf{B} \vDash \varphi[h(\bar{a})]$ y como por hipótesis φ define a R en $\mathcal{K}$, $h(\bar{a}) \in R^{\mathbf{B}}$.

Veamos 2⇒1:

Sea

$$\varphi = \bigvee_{\mathbf{A} \in \mathcal{K}} \left(\bigvee_{\bar{a} \in R^{\mathbf{A}}} \left(\bigwedge_{\alpha \in \Delta^+_{\mathbf{A}\bar{a}}} \alpha(\bar{x}) \right) \right),$$

la cual es fórmula por el Teorema 5. Sea $\mathbf{B} \in \mathcal{K}$:

⇒) Supongamos $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$, entonces en particular para algún $\mathbf{A} \in \mathcal{K}$ y $\bar{a} \in R^{\mathbf{A}}$, $\mathbf{B} \vDash \Delta^+_{\mathbf{A}\bar{a}}\left[\bar{b}\right]$. Por Lema 15, hay un homomorfismo $h : \langle \bar{a} \rangle^{\mathbf{A}} \to \langle \bar{b} \rangle^{\mathbf{B}}$ tal que $h(\bar{a}) = \bar{b}$. Entonces por hipótesis h preserva R. Finalmente como $\bar{a} \in R^{\mathbf{A}}$, luego $h(\bar{a}) \in R^{\mathbf{B}}$ o sea que $\bar{b} \in R^{\mathbf{B}}$.

⇐) Supongamos $\bar{b} \in R^{\mathbf{B}}$. Entonces $\varphi = \cdots \vee \cdots \vee \left(\bigwedge_{\alpha \in \Delta^+_{\mathbf{B}\bar{b}}} \alpha(\bar{x}) \right)$, evidentemente $\mathbf{B} \vDash \Delta^+_{\mathbf{B}\bar{b}}\left[\bar{b}\right]$, entonces $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$. □

2.4. Definibilidad de primer orden

Éste es el tipo de definibilidad más débil. Los morfismos en cuestión son los isomorfismos entre estructuras de $\mathcal{K}$, por lo que son un subconjunto de los revisados en el Teorema 14.

Teorema 17. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden, sea $R \in \mathcal{L}' - \mathcal{L}$ un símbolo de relación n-ario. Para una clase finita $\mathcal{K}$ de $\mathcal{L}'$-estructuras, los siguientes son equivalentes:*

1. *Hay una fórmula en* $\mathrm{Fo}(\mathcal{L})$ *que define* R *en* $\mathcal{K}$.

2. *Para todas* $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{K}$, *todo isomorfismo* $\gamma : \boldsymbol{A}_{\mathcal{L}} \to \boldsymbol{B}_{\mathcal{L}}$ *preserva* R.

Demostración. 1⇒2) Sea φ una $\mathcal{L}$-fórmula que define a R en $\mathcal{K}$, y sea $\gamma : \mathbf{A} \to \mathbf{B}$ un isomorfismo. Si $\mathbf{A} \vDash \varphi[\bar{a}]$, entonces $\mathbf{B} \vDash \varphi[\gamma(\bar{a})]$, y como φ define a R en $\mathcal{K}$, tenemos que $\gamma(\bar{a}) \in R^{\mathbf{B}}$.

2⇒1) Para cada $\mathbf{A} \in \mathcal{K}$, cada $\bar{a} \in R^{\mathbf{A}}$, fijamos $\bar{a'}$ tal que $\langle \bar{a}, \bar{a'} \rangle^{\mathbf{A}} = \mathbf{A}$. Ahora sea

$$\delta_{\mathbf{A},(\bar{a},\bar{a'})}(\bar{x},\bar{y}) = \bigwedge_{\alpha \in \Delta_{(\bar{a},\bar{a'}),\mathbf{A}}} \alpha(\bar{x},\bar{y}),$$

que es fórmula por el Teorema 5. Definimos

$$\varphi(\bar{x}) = \bigvee_{\mathbf{A}\in\mathcal{K}} \left(\left(\bigvee_{\bar{a}\in R^{\mathbf{A}}} \left(\exists \bar{y}\, \delta_{\mathbf{A},(\bar{a},\bar{a'})}(\bar{x},\bar{y})\right) \right) \wedge \left(\forall x_1,\ldots,x_{|\mathbf{A}|+1} \bigvee_{i\neq j} x_i = x_j \right) \right).$$

Veremos que $\varphi(\bar{x})$ define a R en $\mathcal{K}$. Fijamos $\mathbf{B}$ en $\mathcal{K}$ y $b_1,\ldots,b_n \in B$. Supongamos que $(b_1,\ldots,b_n) \in R^{\mathbf{B}}$. Entonces, como

$$\varphi = \cdots \vee \left(\left(\exists \bar{y}\, \delta_{\mathbf{B},(\bar{b},\bar{b'})}(\bar{x},\bar{y}) \vee \ldots \right) \wedge \left(\forall x_1,\ldots,x_{|\mathbf{B}|+1} \bigvee_{i\neq j} x_i = x_j \right) \right) \vee \ldots,$$

y como $\mathbf{B} \vDash \exists \bar{y}\, \delta_{\mathbf{B},(\bar{b},\bar{b'})}(\bar{x},\bar{y}) \left[\bar{b},\bar{b'}\right]$ y $\mathbf{B} \vDash \forall x_1,\ldots,x_{|\mathbf{B}|+1} \bigvee_{i\neq j} x_i = x_j \left[\bar{b},\bar{b'}\right]$, tenemos que entonces $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$.

Supongamos ahora que . Entonces hay algún $\mathbf{A} \in \mathcal{K}$, y algún $\bar{a} \in R^{\mathbf{A}}$ tales que

$$\mathbf{B} \vDash \left(\exists \bar{y}\, \delta_{\mathbf{A},(\bar{a},\bar{a'})}(\bar{x},\bar{y})\right) \wedge \left(\forall x_1,\ldots,x_{|\mathbf{A}|+1} \bigvee_{i\neq j} x_i = x_j \right) \left[\bar{b}\right].$$

Se sigue que hay un $\bar{z}$ en $\mathbf{B}$ tal que

$$\mathbf{B} \vDash \delta_{\mathbf{A},(\bar{a},\bar{a'})}(\bar{x},\bar{y}) \left[\bar{b},\bar{z}\right].$$

Como $\mathbf{B} \vDash \Delta_{(\bar{a},\bar{a'}),\mathbf{A}} \left[\bar{b},\bar{z}\right]$, por Lema 9, hay $\gamma : \langle \bar{a},\bar{a'} \rangle^{\mathbf{A}} \to \langle \bar{b},\bar{b'} \rangle^{\mathbf{B}}$ isomorfismo tal que $\left(\gamma(\bar{a}), \gamma\left(\bar{a'}\right)\right) = \left(\bar{b},\bar{z}\right)$ con $\langle \bar{a},\bar{a'} \rangle^{\mathbf{A}} = \mathbf{A}$ y $\langle \bar{b},\bar{b'} \rangle^{\mathbf{B}} \leq \mathbf{B}$, lo que implica que $|\mathbf{A}| \leq |\mathbf{B}|$. Pero como

$$\mathbf{B} \vDash \forall x_1,\ldots,x_{|\mathbf{A}|+1} \bigvee_{i\neq j} x_i = x_j \left[b_1,\ldots,b_n\right]$$

tenemos que $|\mathbf{B}| \leq |\mathbf{A}|$, y así $|\mathbf{B}| = |\mathbf{A}|$. Luego $\gamma : \mathbf{A} \to \mathbf{B}$ es un isomorfismo que cumple $\left(\gamma(\bar{a}), \gamma\left(\bar{a'}\right)\right) = \left(\bar{b},\bar{z}\right)$, y sabemos que los isomorfismos entre estructuras de $\mathcal{K}$ preservan R. Entonces, ya que$\bar{a} \in R^{\mathbf{A}}$, tenemos que $\gamma(\bar{a}) = \bar{b} \in R^{\mathbf{B}}$. □

2.5. Definibilidad por fórmulas existenciales

A continuación, estableceremos la noción semántica para la definibilidad existencial.

En el siguiente lema, construimos una fórmula equivalente al diagrama abierto para un conjunto de elementos tal que sea capaz de generar la estructura completa.

Lema 18. *Sea* $\mathbf{A}$ *una estructura finita y* $a_1, \ldots, a_n \in A$*. Entonces hay una fórmula existencial* $\varphi(x_1, \ldots, x_n)$ *tal que para toda estructura* $\mathbf{B}$ *y para todo* $\bar{b} \in B^n$ *los siguientes son equivalentes:*

1. $\mathbf{B} \vDash \varphi[\bar{b}]$.

2. *Hay un embedding* $\gamma : \mathbf{A} \to \mathbf{B}$ *tal que* $\gamma(\bar{a}) = \bar{b}$

Demostración. Sean $a'_1, \ldots, a'_m$ tales que $\langle a_1, \ldots, a_n, a'_1, \ldots, a'_m \rangle^{\mathbf{A}} = \mathbf{A}$ y sea

$$\varphi = \exists y_1, \ldots, y_m \left(\bigwedge_{\alpha \in \Delta_{(\bar{a}, \bar{a'}), \mathbf{A}}} \alpha(\bar{x}, \bar{y}) \right)$$

que es fórmula por el Teorema 5.

Supongamos $\mathbf{B} \vDash \varphi[\bar{b}]$, luego existen $b'_1, \ldots, b'_m \in B$ tales que

$$\mathbf{B} \vDash \bigwedge_{\alpha \in \Delta_{(\bar{a}, \bar{a'}), \mathbf{A}}} \alpha(\bar{x}, \bar{y}) [\bar{b}, \bar{b'}]$$

por Lema 9 hay isomorfismo $\gamma : \langle \bar{a}, \bar{a'} \rangle^{\mathbf{A}} \to \langle \bar{b}, \bar{b'} \rangle^{\mathbf{B}}$ tal que $\gamma(\bar{a}, \bar{a'}) = (\bar{b}, \bar{b'})$, y como $\langle \bar{a}, \bar{a'} \rangle^{\mathbf{A}} = \mathbf{A}$ y $\langle \bar{b}, \bar{b'} \rangle^{\mathbf{B}} \leq \mathbf{B}$, γ es un embedding de $\mathbf{A}$ en $\mathbf{B}$ tal que $\gamma(\bar{a}) = \bar{b}$.

Supongamos que hay un embedding $\gamma : \mathbf{A} \to \mathbf{B}$ tal que $\gamma(\bar{a}) = \bar{b}$, por lo tanto $\gamma : \mathbf{A} \to \mathbf{S} \leq \mathbf{B}$ es un isomorfismo. Como $\mathbf{A} \vDash \varphi[\bar{a}]$, por ser γ un isomorfismo $\mathbf{S} \vDash \varphi[\gamma(\bar{a})]$ y entonces $\mathbf{S} \vDash \varphi[\bar{b}]$. Por lo tanto existe $\bar{b'}$en $S \subseteq B$ tal que $\mathbf{S} \vDash \bigwedge_{\alpha \in \Delta_{(\bar{a}, \bar{a'}), \mathbf{A}}} \alpha(\bar{x}, \bar{y}) [\bar{b}, \bar{b'}]$. Como $\bigwedge_{\alpha \in \Delta_{(\bar{a}, \bar{a'}), \mathbf{A}}} \alpha(\bar{x}, \bar{y})$ es una fórmula abierta, por Lema 7, $\mathbf{B} \vDash \bigwedge_{\alpha \in \Delta_{(\bar{a}, \bar{a'}), \mathbf{A}}} \alpha(\bar{x}, \bar{y}) [\bar{b}, \bar{b'}]$, luego $\mathbf{B} \vDash \varphi[\bar{b}]$ □

En el teorema enunciado a continuación, se determina que la definibilidad existencial depende de la preservación en los embeddings entre las estructuras de $\mathcal{K}$. Notar que esta condición incluye a los isomorfismos de estructuras de $\mathcal{K}$, chequeados para el Teorema 17, pero no a todos los isomorfismos entre subestructuras de estructuras de $\mathcal{K}$, chequeados para el Teorema 14.

Teorema 19. *Sean* $\mathcal{L} \subseteq \mathcal{L}'$ *lenguajes de primer orden, sea* $R \in \mathcal{L}' - \mathcal{L}$ *un símbolo de relación n-ario. Para una clase finita* $\mathcal{K}$ *de* $\mathcal{L}'$*-estructuras, los siguientes son equivalentes:*

1. *Hay una fórmula en* $\mathrm{E}(\mathcal{L})$ *que define* R *en* $\mathcal{K}$.

2. *Para todas* $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{K}$*, todo embedding* $\gamma : \boldsymbol{A}_{\mathcal{L}} \to \boldsymbol{B}_{\mathcal{L}}$ *preserva* R.

Demostración. 1⇒2) Sea φ existencial que define a R en $\mathcal{K}$, y sea $\gamma : \mathbf{A} \to \mathbf{B}$ un embedding. Entonces $\gamma : \mathbf{A} \to \mathbf{S} \leq \mathbf{B}$ es un isomorfismo. Supongamos $\mathbf{A} \vDash \varphi[\bar{a}]$ entonces $\mathbf{S} \vDash \varphi[\gamma(\bar{a})]$. Como φ es de la forma $\varphi = \exists \bar{y}\, \psi(\bar{x}, \bar{y})$ con ψ abierta, existe un $\bar{s}$ tal que $\mathbf{S} \vDash \psi[\gamma(\bar{a}), \bar{s}]$. Como ψ es abierta y $\mathbf{S} \leq \mathbf{B}$, por el Lema 7 $\mathbf{B} \vDash \psi[\gamma(\bar{a}), \bar{s}]$. Entonces $\mathbf{B} \vDash \varphi[\gamma(\bar{a})]$ y como φ define a R en $\mathcal{K}$, entonces $\gamma(\bar{a}) \in R^{\mathbf{B}}$.

2⇒1) Para cada $\mathbf{A} \in \mathcal{K}$, cada $\bar{a} \in R^{\mathbf{A}}$, tomamos $\bar{a'}$ tal que $\langle \bar{a}, \bar{a'} \rangle^{\mathbf{A}} = \mathbf{A}$. Ahora tomamos

$$\delta_{\mathbf{A},(\bar{a},\bar{a'})}(\bar{x},\bar{y}) = \bigwedge_{\alpha \in \Delta_{(\bar{a},\bar{a'}),\mathbf{A}}} \alpha(\bar{x},\bar{y}),$$

que es fórmula por Teorema 5. Ahora tomamos

$$\varphi = \bigvee_{\mathbf{A} \in \mathcal{K}} \left(\bigvee_{\bar{a} \in R^{\mathbf{A}}} \left(\exists \bar{y}\, \delta_{\mathbf{A},(\bar{a},\bar{a'})}(\bar{x},\bar{y}) \right) \right)$$

Sea $(b_1, \ldots, b_n) \in R^{\mathbf{B}}$. Entonces como

$$\varphi = \cdots \vee \exists \bar{y}\, \delta_{\mathbf{B},(\bar{b},\bar{b'})}(\bar{x},\bar{y}) \vee \ldots,$$

y como

$$\mathbf{B} \vDash \exists \bar{y}\, \delta_{\mathbf{B},(\bar{b},\bar{b'})}(\bar{x},\bar{y}) \left[\bar{b}, \bar{b'}\right],$$

luego $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$.

Sea $\mathbf{B} \vDash \varphi[b_1, \ldots, b_n]$. Entonces para algún $\mathbf{A} \in \mathcal{K}$, y algún $\bar{a} \in R^{\mathbf{A}}$ tales que $\mathbf{B} \vDash \exists \bar{y}\, \delta_{\mathbf{A},(\bar{a},\bar{a'})}(\bar{x},\bar{y}) \left[\bar{b}\right]$. Entonces por el Lema 18 hay $\gamma : \mathbf{A} \to \mathbf{B}$ embedding tal que $\gamma(\bar{a}) = \bar{b}$. Finalmente como $\bar{a} \in R^{\mathbf{A}}$, $\gamma(\bar{a}) \in R^{\mathbf{B}}$, $\bar{b} \in R^{\mathbf{B}}$.

Notar que como φ es disyunción de existenciales, por teorema puede ser convertida en una fórmula existencial. □

2.6. Definibilidad por fórmulas existenciales positivas

Aquí determinaremos la noción semántica para la definibilidad existencial positiva.

En el siguiente lema, construimos una fórmula equivalente al diagrama atómico positivo para un conjunto de elementos tal que sea capaz de generar la estructura completa análogamente al Lema 18.

Lema 20. *Sea $\mathbf{A}$ una estructura finita y $a_1, \ldots, a_n \in A$. Entonces hay una fórmula existencial positiva $\varphi(x_1, \ldots, x_n)$ tal que para toda estructura $\mathbf{B}$ y para todo $\bar{b} \in B^n$ los siguientes son equivalentes:*

1. $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$.
2. *Hay un homomorfismo $h : \mathbf{A} \to \mathbf{B}$ tal que $h(\bar{a}) = \bar{b}$*

Demostración. Sean $a'_1, \ldots, a'_m$ tales que $\langle a_1, \ldots, a_n, a'_1, \ldots, a'_m \rangle^{\mathbf{A}} = \mathbf{A}$ y sea

$$\varphi = \exists y_1, \ldots, y_m \left(\bigwedge_{\alpha \in \Delta^+_{(\bar{a},\bar{a'}),\mathbf{A}}} \alpha(\bar{x},\bar{y}) \right)$$

que es fórmula por el Teorema 5.

Supongamos $\mathbf{B} \vDash \varphi\left[\bar{b}\right]$, luego existen $b'_1, \ldots, b'_m \in B$ tales que

$$\mathbf{B} \vDash \bigwedge_{\alpha \in \Delta^+_{(\bar{a},\bar{a'}),\mathbf{A}}} \alpha(\bar{x},\bar{y}) \left[\bar{b}, \bar{b'}\right]$$

por Lema 12 hay homomorfismo

$$h : \langle \bar{a}, \bar{a'} \rangle^{\mathbf{A}} \to \langle \bar{b}, \bar{b'} \rangle^{\mathbf{B}}$$

tal que $h(\bar{a}, \bar{a'}) = (\bar{b}, \bar{b'})$, y como $\langle \bar{a}, \bar{a'} \rangle^{\mathbf{A}} = \mathbf{A}$ y $\langle \bar{b}, \bar{b'} \rangle^{\mathbf{B}} \leq \mathbf{B}$, h es un homomorfismo de $\mathbf{A}$ en $\mathbf{B}$ tal que $h(\bar{a}) = \bar{b}$.

Supongamos que hay un homomorfismo $h : \mathbf{A} \to \mathbf{B}$ tal que $h(\bar{a}) = \bar{b}$. Como $\mathbf{A} \vDash \varphi[\bar{a}]$, entonces $\mathbf{A} \vDash \Delta^{+}_{(\bar{a},\bar{a'}),\mathbf{A}}[\bar{a}, \bar{a'}]$ y por Lema 10

$$\mathbf{B} \vDash \Delta^{+}_{(\bar{a},\bar{a'}),\mathbf{A}}[h(\bar{a}), h(\bar{a'})].$$

Entonces $\mathbf{B} \vDash \Delta^{+}_{(\bar{a},\bar{a'}),\mathbf{A}}[\bar{b}, \bar{b'}]$, por lo que claramente $\mathbf{B} \vDash \varphi[\bar{b}]$. □

El siguiente teorema establece que la definibilidad existencial positiva depende de la preservación en los homomorfismos entre las estructuras de $\mathcal{K}$. Notar que esta condición incluye a los embeddings entre estructuras de $\mathcal{K}$, chequeados para el Teorema 19, pero no a todos los homomorfismos entre subestructuras de estructuras de $\mathcal{K}$, chequeados para el Teorema 16, mientras que es disjunto con los isomorfismos entre subestructuras de estructuras de $\mathcal{K}$, chequeados para el Teorema 14.

Teorema 21. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden, sea $R \in \mathcal{L}' - \mathcal{L}$ un símbolo de relación n-ario. Para una clase finita $\mathcal{K}$ de $\mathcal{L}'$-estructuras, los siguientes son equivalentes:*

1. *Hay una fórmula en $\mathrm{E}^{+}(\mathcal{L})$ que define R en $\mathcal{K}$.*
2. *Para todas $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{K}$, todo homomorfismo $\gamma : \boldsymbol{A}_{\mathcal{L}} \to \boldsymbol{B}_{\mathcal{L}}$ preserva R.*

Demostración. Igual a la prueba del Teorema 19, pero utilizando las fórmulas dadas por el Lema 20. □

2.7. Definibilidad por conjunción de atómicas y primitivas positivas

En este tipo de definibilidad se determina una relación con el producto de estructuras en $\mathcal{K}$.

Lo primero es definir nuevamente el concepto de preservación pero esta vez a través del producto de estructuras. Podemos pensar la siguiente definición matricialmente, donde las filas pertenecen a D mientras que cada i-esima columna pertenece a R^{A_i}

Definición 22. Sean $A_1, \ldots, A_m, B$ conjuntos, $R^{A_i} \subseteq A_i^n$ para cada $i \in [1, m]$, $R^B \subseteq B^n$ y $h : D \subseteq A_1 \times \cdots \times A_m \to B$ una función. Diremos que h *preserva* R si dados

$$\bar{a}_1 = (a_{11}, \ldots, a_{1m}), \ldots, \bar{a}_n = (a_{n1}, \ldots, a_{nm}) \in D$$

tales que

$$(a_{1i}, \ldots, a_{ni}) \in R^{A_i} \text{para cada i} \in [1,m]$$

se tiene que $(h(\bar{a}_1), \ldots, h(\bar{a}_n)) \in R^B$.

El siguiente lema determina que el producto entre estructuras preserva las formulas atómicas.

Lema 23. *Sean* $\mathbf{A}_1, \ldots, \mathbf{A}_m$ *estructuras y* $\varphi = \exists \bar{y}\, \psi(\bar{x}, \bar{y})$ *con* ψ *una conjunción de fórmulas atómicas. Entonces son equivalentes:*

1. *Para todo* $i = 1, \ldots, m$ *se da que* $\mathbf{A}_i \vDash \varphi[a_{1i}, \ldots, a_{ni}]$
2. $\mathbf{A}_1 \times \cdots \times \mathbf{A}_m \vDash \varphi[\bar{a}_1, \ldots, \bar{a}_n]$, *con* $\bar{a}_j = (a_{j1}, \ldots, a_{jm})$

Demostración. Rutina. □

En el siguiente teorema se establece la dependencia de la definibilidad por conjunción de atómicas con la preservación en homomorfismos desde subestructuras de los productos de las estructuras de $\mathcal{K}$ en las estructuras de $\mathcal{K}$.

Teorema 24. *Sean* $\mathcal{L} \subseteq \mathcal{L}'$ *lenguajes de primer orden, sea* $R \in \mathcal{L}' - \mathcal{L}$ *un símbolo de relación n-ario. Para una clase finita* $\mathcal{K}$ *de* $\mathcal{L}'$*-estructuras, los siguientes son equivalentes:*

1. *Para cada* $m \geq 1$*, para todas* $\mathbf{A}_1, \ldots, \mathbf{A}_m, \boldsymbol{B} \in \mathcal{K}$*, para cada*
$$\boldsymbol{S} \leq (\mathbf{A}_1 \times \cdots \times \mathbf{A}_m)_{\mathcal{L}},$$
todo homomorfismo $h : \mathbf{S} \to \boldsymbol{B}_{\mathcal{L}}$ *preserva* R.
2. *Hay una conjunción finita de* $\mathcal{L}$*-fórmulas atómicas que define* R *en* $\mathcal{K}$.

Demostración. 2⇒1) Sea $\varphi(\bar{x})$ conjunción de atómicas que define a R en $\mathcal{K}$. Sean $m \geq 1$, $\mathbf{A}_1, \ldots, \mathbf{A}_m, \boldsymbol{B} \in \mathcal{K}$, $\boldsymbol{S} \leq (\mathbf{A}_1 \times \cdots \times \mathbf{A}_m)_{\mathcal{L}}$, y $h : \mathbf{S} \to \boldsymbol{B}_{\mathcal{L}}$ un homomorfismo. Sean

$$\bar{s}_1 = (s_{11}, \ldots, s_{1m}), \ldots, \bar{s}_n = (s_{n1}, \ldots, s_{nm}) \in S$$

tales que

$$(s_{1i}, \ldots, s_{ni}) \in R^{A_i} \text{ para cada } i \in \{1, \ldots, m\}.$$

Veamos que $(h(\bar{s}_1), \ldots, h(\bar{s}_n)) \in R^B$. Como $(s_{1i}, \ldots, s_{ni}) \in R^{A_i}$ y φ define a R en $\mathcal{K}$, tenemos que

$$\mathbf{A}_i \vDash \varphi[s_1, \ldots, s_{ni}] \text{ para } i \in \{1, \ldots, m\}.$$

Luego, por el Lema 23,

$$\mathbf{A}_1 \times \cdots \times \mathbf{A}_m \vDash \varphi[\bar{s}_1, \ldots, \bar{s}_n].$$

Como φ es abierta y $\boldsymbol{S} \leq (\mathbf{A}_1 \times \cdots \times \mathbf{A}_m)_{\mathcal{L}}$, entonces el Lema 7 implica que $\mathbf{S} \vDash \varphi[\bar{s}_1, \ldots, \bar{s}_n]$, y aplicando luego el Lema 10 obtenemos que

$$\mathbf{B} \vDash \varphi[h(\bar{s}_1), \ldots, h(\bar{s}_n)].$$

Luego, como φ define a R en $\mathcal{K}$ y $\mathbf{B} \in \mathcal{K}$, concluimos que

$$(h(\bar{s}_1), \ldots, h(\bar{s}_n)) \in R^{\mathbf{B}}.$$

1⇒2) Supongamos $\mathcal{K} = \{\mathbf{A}_1, \ldots, \mathbf{A}_m\}$, R n-aria. Sean

$$\bar{x}_1, \ldots, \bar{x}_n \in \mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|}$$

tales que

$$\bar{x}_i = \left(x_{i1}^1, \ldots, x_{i|R^{\mathbf{A}_1}|}^1, \ldots, x_{i1}^m, \ldots, x_{i|R^{\mathbf{A}_m}|}^m\right)$$

para cada $i \in \{1, \ldots, m\}$ con

$$\bigcup R^{\mathbf{A}_j} = \left\{\left(x_{1i}^j, \ldots, x_{ni}^j\right) \text{ con } i \in \left\{1, \left|R^{\mathbf{A}_j}\right|\right\}\right\}$$

para cada $i \in \{1, \ldots, m\}$. Sea

$$\varphi = \bigwedge_{\alpha \in \Delta^+_{\mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|}, (\bar{x}_1, \ldots, \bar{x}_n)}} \alpha(\bar{x}).$$

Supongamos $\mathbf{B} \vDash \varphi[b_1, \ldots, b_n]$ y veamos que $(b_1, \ldots, b_n) \in R^{\mathbf{B}}$. Como

$$\mathbf{B} \vDash \Delta^+_{\mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|}, (\bar{x}_1, \ldots, \bar{x}_n)} [b_1, \ldots, b_n],$$

por Lema 12 hay h homomorfismo de $\langle \bar{x}_1, \ldots, \bar{x}_n \rangle^{\mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|}}$ en $\langle b_1, \ldots, b_n \rangle^{\mathbf{B}}$ tal que $h(\bar{x}_i) = b_i$. Como por hipótesis h preserva R, entonces como $\left(x_{1i}^j, \ldots, x_{ni}^j\right) \in R^{\mathbf{A}_j}$, $(h(\bar{x}_1), \ldots, h(\bar{x}_n)) \in R^{\mathbf{B}}$, que es exactamente $(b_1, \ldots, b_n) \in R^{\mathbf{B}}$.

Ahora supongamos $(b_1, \ldots, b_n) \in R^{\mathbf{B}}$ y veamos que $\mathbf{B} \vDash \varphi[b_1, \ldots, b_n]$, luego hay algún j tal que $\mathbf{A}_j = \mathbf{B}$ y hay algún k y algún j tal que

$$\left(x_{1k}^j, \ldots, x_{nk}^j\right) = (b_1, \ldots, b_n).$$

Entonces como

$$\mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \mathbf{B}^{|R^{\mathbf{B}}|} \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|} \vDash \varphi[\bar{x}_1, \ldots, \bar{x}_n]$$

por Lema 23 $\mathbf{B} \vDash \varphi\left[x_{1k}^j, \ldots, x_{nk}^j\right]$, por lo tanto $\mathbf{B} \vDash \varphi[b_1, \ldots, b_n]$. □

El siguiente teorema establece la dependencia de la definibilidad primitiva positiva con la preservación en homomorfismos de los productos de las estructuras de $\mathcal{K}$ en las estructuras de $\mathcal{K}$.

Teorema 25. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden, sea $R \in \mathcal{L}' - \mathcal{L}$ un símbolo de relación n-ario. Para una clase finita $\mathcal{K}$ de $\mathcal{L}'$-estructuras, los siguientes son equivalentes:*

1. *Para cada $m \geq 1$, para todas $\mathbf{A}_1, \ldots, \mathbf{A}_m, \boldsymbol{B} \in \mathcal{K}$, todo homomorfismo $h : (\mathbf{A}_1 \times \cdots \times \mathbf{A}_m)_{\mathcal{L}} \to \boldsymbol{B}_{\mathcal{L}}$ preserva R.*

2. *Hay una $\mathcal{L}$-fórmula primitiva positiva que define R en $\mathcal{K}$.*

Demostración. 2⇒1) Sea $\varphi = \exists \bar{y}\, \psi(\bar{x}, \bar{y})$ primitiva positiva que define a R en $\mathcal{K}$. Sean $m \geq 1$, $\mathbf{A}_1, \ldots, \mathbf{A}_m, \boldsymbol{B} \in \mathcal{K}$ y $h : (\mathbf{A}_1 \times \cdots \times \mathbf{A}_m)_{\mathcal{L}} \to \boldsymbol{B}_{\mathcal{L}}$ un homomorfismo. Sean

$$\bar{a}_1 = (a_{11}, \ldots, a_{1m}), \ldots, \bar{a}_n = (a_{n1}, \ldots, a_{nm}) \in S$$

tales que

$$(a_{1i}, \ldots, a_{ni}) \in R^{A_i} \text{ para cada } i \in \{1, \ldots, m\}.$$

Veamos que $(h(\bar{a}_1), \ldots, h(\bar{a}_n)) \in R^B$. Como $(a_{1i}, \ldots, a_{ni}) \in R^{A_i}$ y φ define a R en $\mathcal{K}$, tenemos que

$$\mathbf{A}_i \vDash \varphi[a_1, \ldots, a_{ni}] \text{ para } i \in \{1, \ldots, m\}.$$

Luego, por el Lema 23,

$$\mathbf{A}_1 \times \cdots \times \mathbf{A}_m \vDash \varphi[\bar{a}_1, \ldots, \bar{a}_n].$$

Entonces hay $b_1, \ldots, b_k \in A_1 \times \cdots \times A_m$, tales que

$$\mathbf{A}_1 \times \cdots \times \mathbf{A}_m \vDash \psi[\bar{a}_1, \ldots, \bar{a}_n, b_1, \ldots, b_k].$$

Como ψ es abierta positiva, aplicando el Lema 10 obtenemos que

$$\mathbf{B} \vDash \varphi[h(\bar{a}_1), \ldots, h(\bar{a}_n)].$$

Luego, como φ define a R en $\mathcal{K}$ y $\mathbf{B} \in \mathcal{K}$, concluimos que

$$(h(\bar{a}_1), \ldots, h(\bar{a}_n)) \in R^{\mathbf{B}}.$$

1⇒2)Supongamos $\mathcal{K} = \{\mathbf{A}_1, \ldots, \mathbf{A}_m\}$, R n-aria. Sean

$$\bar{x}_1, \ldots, \bar{x}_n \in \mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|}$$

tales que

$$\bar{x}_i = \left(x_{i1}^1, \ldots, x_{i|R^{\mathbf{A}_1}|}^1, \ldots, x_{i1}^m, \ldots, x_{i|R^{\mathbf{A}_m}|}^m\right)$$

para cada $i \in \{1, \ldots, m\}$ con

$$\bigcup R^{\mathbf{A}_j} = \left\{\left(x_{1i}^j, \ldots, x_{ni}^j\right) \text{ con } i \in \left\{1, \left|R^{\mathbf{A}_j}\right|\right\}\right\}$$

para cada $i \in \{1, \ldots, m\}$. Sean

$$\bar{x'}_1, \ldots, \bar{x'}_k \in \mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|},$$

tales que

$$\left\langle \bar{x}_1, \ldots, \bar{x}_n, \bar{x'}_1, \ldots, \bar{x'}_k \right\rangle^{\mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|}} = \mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|}.$$

Sea

$$\varphi = \exists \bar{y} \bigwedge_{\alpha \in \Delta^+_{\mathbf{A}_1^{|R^{\mathbf{A}_1}|} \times \cdots \times \mathbf{A}_m^{|R^{\mathbf{A}_m}|}, (\bar{x}_1, \ldots, \bar{x}_n, \bar{x'}_1, \ldots, \bar{x'}_k)}} \alpha(\bar{x}, \bar{y}).$$

Supongamos $\mathbf{B} \vDash \varphi[b_1, \ldots, b_n]$ y veamos que $(b_1, \ldots, b_n) \in R^{\mathbf{B}}$. Entonces hay $b'_1, \ldots, b'_k$ tal que

$$\mathbf{B} \vDash \Delta^+_{\mathbf{A}_1^{\left|R^{\mathbf{A}_1}\right|} \times \cdots \times \mathbf{A}_m^{\left|R^{\mathbf{A}_m}\right|}, \left(\bar{x}_1, \ldots, \bar{x}_n, \bar{x'}_1, \ldots, \bar{x'}_k\right)} [b_1, \ldots, b_n, b'_1, \ldots, b'_k],$$

por Lema 12 hay h homomorfismo de

$$\left\langle \bar{x}_1, \ldots, \bar{x}_n, \bar{x'}_1, \ldots, \bar{x'}_k \right\rangle^{\mathbf{A}_1^{\left|R^{\mathbf{A}_1}\right|} \times \cdots \times \mathbf{A}_m^{\left|R^{\mathbf{A}_m}\right|}}$$

en

$$\langle b_1, \ldots, b_n, b'_1, \ldots, b'_k \rangle^{\mathbf{B}}$$

tal que $h(\bar{x}_i) = b_i$ y $h\left(\bar{x'}_i\right) = b'_i$, claramente h es homomorfismo de

$$\mathbf{A}_1^{\left|R^{\mathbf{A}_1}\right|} \times \cdots \times \mathbf{A}_m^{\left|R^{\mathbf{A}_m}\right|}$$

en $\mathbf{B}$. Como por hipótesis h preserva R, entonces como $\left(x^j_{1i}, \ldots, x^j_{ni}\right) \in R^{\mathbf{A}_j}$ se da que $(h(\bar{x}_1), \ldots, h(\bar{x}_n)) \in R^{\mathbf{B}}$, que es exactamente $(b_1, \ldots, b_n) \in R^{\mathbf{B}}$.

Ahora supongamos $(b_1, \ldots, b_n) \in R^{\mathbf{B}}$ y veamos que $\mathbf{B} \vDash \varphi[b_1, \ldots, b_n]$, luego hay algún j tal que $\mathbf{A}_j = \mathbf{B}$ y hay algún k y algún j tal que

$$\left(x^j_{1k}, \ldots, x^j_{nk}\right) = (b_1, \ldots, b_n).$$

Entonces como

$$\mathbf{A}_1^{\left|R^{\mathbf{A}_1}\right|} \times \cdots \times \mathbf{B}^{\left|R^{\mathbf{B}}\right|} \times \cdots \times \mathbf{A}_m^{\left|R^{\mathbf{A}_m}\right|} \vDash \varphi[\bar{x}_1, \ldots, \bar{x}_n]$$

por Lema 23 $\mathbf{B} \vDash \varphi\left[x^j_{1k}, \ldots, x^j_{nk}\right]$, por lo tanto $\mathbf{B} \vDash \varphi[b_1, \ldots, b_n]$. $\square$

Capítulo 3

Algoritmos de chequeo de definibilidad

En este capítulo describimos algoritmos para el chequeo de definibilidad de relaciones en conjuntos finitos de estructuras finitas por fragmentos de primer orden.

3.1. Definiciones e ideas preliminares

Para evitar las complicaciones resultantes de que tanto una estructura $\mathbf{A}$ y una subestructura propia $\mathbf{A}_0$ pertenezcan al conjunto $\mathcal{K}$, definiremos el concepto de clase *disjunta*. En general asumiremos que nuestra clase $\mathcal{K}$ es siempre disjunta. Notar que no hay perdida de generalidad ya que bastaría con renombrar elementos para convertir una clase no disjunta en una disjunta equivalente. Una clase $\mathcal{K}$ de estructuras será *disjunta* si dadas $\mathbf{A}, \mathbf{B} \in \mathcal{K}$ se tiene que $A \cap B \neq \emptyset$ si y solo si $\mathbf{A} = \mathbf{B}$. La clase $\mathcal{K}$ será *normal* si es disjunta, finita y cada una de las estructuras en $\mathcal{K}$ es finita.

A la hora de reducir la complejidad de nuestros algoritmos en la búsqueda de homomorfismos utilizaremos conjuntos reducidos de morfismos, que *generan* el resto. Dados $\mathcal{F}_0$ y $\mathcal{F}$ dos conjuntos de funciones, diremos que $\mathcal{F}_0$ *genera* a $\mathcal{F}$ si $\mathcal{F}_0 \subseteq \mathcal{F}$ y para toda $f \in \mathcal{F}$ hay $f_1, \ldots, f_n \in \mathcal{F}_0$(quizás repetidas) tales que $f = f_1 \circ \cdots \circ f_n$.

Dado un conjunto $\mathcal{K}$ de estructuras, para referirnos a los morfismos nombrados en los Teoremas 17, 14, 21, 16 definimos los siguientes conjuntos:

$$\operatorname{iso}(\mathcal{K}) = \{\gamma : \gamma \text{ isomorfismo de } \mathbf{A} \text{ en } \mathbf{B} \text{ con } \mathbf{A}, \mathbf{B} \in \mathcal{K}\}$$

$$\operatorname{sub\,iso}(\mathcal{K}) = \{\gamma : \gamma \text{ isomorfismo de } \mathbf{A}_0 \text{ en } \mathbf{B}_0 \text{ con } \mathbf{A}_0, \mathbf{B}_0 \in \mathbb{S}(\mathcal{K})\}$$

$$\operatorname{hom}(\mathcal{K}) = \{\gamma : \gamma \text{ homomorfismo de } \mathbf{A} \text{ en } \mathbf{B} \text{ con } \mathbf{A}, \mathbf{B} \in \mathcal{K}\}$$

$$\operatorname{sub\,hom}(\mathcal{K}) = \{\gamma : \gamma \text{ homomorfismo de } \mathbf{A}_0 \text{ en } \mathbf{B}_0 \text{ con } \mathbf{A}_0, \mathbf{B}_0 \in \mathbb{S}(\mathcal{K})\}$$

3.2. Preprocesamiento

Una vez que tenemos una clase normal $\mathcal{K}$ en la que chequear definibilidad tratamos de reducir su redundancia basándonos en el siguiente resultado:

Lema 26. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden, $\mathcal{K}$ una clase normal de $\mathcal{L}'$-estructuras y $\mathbf{A}, \tilde{\mathbf{A}} \in \mathcal{K}$, con $\mathbf{A} \neq \tilde{\mathbf{A}}$, tales que hay un $\mathcal{L}$-isomorfismo $\gamma : A \to \tilde{A}$. Entonces para toda $R \in \mathcal{L}' - \mathcal{L}$ y toda $\mathcal{L}$-fórmula φ son equivalentes:*

1. *φ define a R en $\mathcal{K}$.*

2. *γ es un $\mathcal{L} \cup \{R\}$-isomorfismo y φ define a R en $\mathcal{K} - \{\tilde{\mathbf{A}}\}$.*

Demostración. 1⇒2) Trivial.
2⇒1) $\tilde{\mathbf{A}} \vDash R[\bar{a}]$ sii $\mathbf{A} \vDash R\left[\gamma^{-1}(\bar{a})\right]$ sii $\mathbf{A} \vDash \varphi\left[\gamma^{-1}(\bar{a})\right]$ sii $\tilde{\mathbf{A}} \vDash \varphi[\bar{a}]$. □

Independientemente del tipo de definibilidad que uno quiera chequear el Lema 26 sugiere que uno puede comenzar por reducir $\mathcal{K}$ eliminando copias de estructuras $\mathcal{L} \cup \{R\}$-isomorfas. El Algoritmo 3.1 aprovecha este resultado aprovechando además para chequear por $\mathcal{L}$-isomorfismos y al descubrir uno de estos comprobar si es un $\mathcal{L} \cup \{R\}$-isomorfismo, lo que permite encontrar contraejemplos de definibilidad desde esta etapa ya que el Teorema 17 determina que los isomorfismos entre estructuras de $\mathcal{K}$ deben preservar para que R sea definible en primer orden.

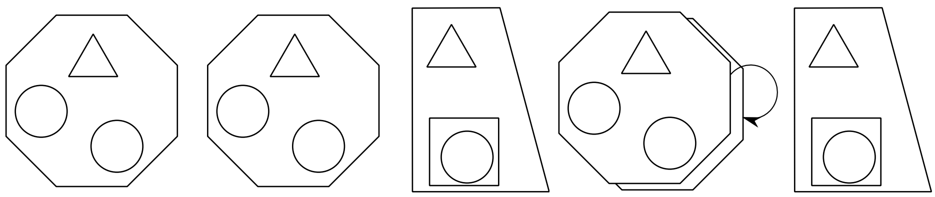
Algorithm 3.1 Preprocesamiento

for $\mathbf{A} \in \mathcal{K}$ **do**
  for $\mathbf{B} \in \mathcal{K} - \{\mathbf{A}\}$ **do**
    if hay $\gamma : \mathbf{A} \to \mathbf{B}$ $\mathcal{L}$-isomorfismo **then**
      if γ es un $\mathcal{L} \cup \{R\}$-isomorfismo **then**
        $\mathcal{K} = \mathcal{K} - \{\mathbf{B}\}$
      else
        return γ ▷ R no es definible y γ es contraejemplo
      end if
    end if
  end for
end for
return $\mathcal{K}$ ▷ $\mathcal{K}$ sin estructuras $\mathcal{L}$-isomorfas

Notar que los isomorfismos revisados por este algoritmo deben ser necesariamente revisados para comprobar definibilidad en cualquiera de los formatos, con la ventaja de que al chequear estos en primer lugar, podría reducirse la clase $\mathcal{K}$ en el proceso.

Observar además que, para poder aplicar los teoremas de definibilidad de las secciones 2.2 y 2.3, habría que verificar que cada uno de los $\mathcal{L}$-isomorfismos entre estructuras de $\mathcal{K}$ preserven R y vemos que basta con chequear solo uno gracias al Lema 26.

Para ejemplificar el funcionamiento del algoritmo, supongamos que $\mathcal{K}$ está dado por la Figura 3.2.1a, donde cada tipo de isomorfismo está representado por una forma geométrica.



(a) $\mathcal{K}$ antes del preprocesamiento (b) $\mathcal{K}$ luego del preprocesamiento

Figura 3.2.1

El algoritmo de preprocesamiento detecta el $\mathcal{L}$-isomorfismo entre los dos octágonos y revisa que sea también un $\mathcal{L} \cup \{R\}$-isomorfismo, dando lugar a la Figura 3.2.1b, donde la flecha es el isomorfismo a chequear.

3.3. Definibilidad abierta

A partir del Teorema 14, habría que chequear todo isomorfismo entre subestructuras de estructuras de $\mathcal{K}$, como en el siguiente lema.

Lema 27. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden tales que $R \in \mathcal{L}' - \mathcal{L}$ es un símbolo de relación n-ario y $\mathcal{K}$ un conjunto normal de $\mathcal{L}'$-estructuras. Entonces si cada $\gamma \in \operatorname{sub iso}(\mathcal{K}_{\mathcal{L}})$ preserva R, R es definible por una $\mathcal{L}$-fórmula abierta en $\mathcal{K}$.*

Demostración. Directo del Teorema 14.

En el teorema enunciado a continuación, describimos un conjunto suficiente de morfismos para generar $\operatorname{sub iso}(\mathcal{K}_{\mathcal{L}})$. Notar que al tomar $\mathcal{K}$ como un conjunto normal de $\mathcal{L}$-estructuras sin estructuras isomorfas estamos pensando en la aplicación previa del Algoritmo 3.1, basado en el Lema 26. $\square$

Teorema 28. *Sea $\mathcal{K}$ un conjunto normal de $\mathcal{L}$-estructuras donde no hay estructuras isomorfas. Sea $\mathcal{S} \subseteq \mathbb{S}(\mathcal{K})$ tal que contiene exactamente un representante por cada tipo de isomorfismo en $S(\mathcal{K})$. Sea $\mathcal{F} \subseteq \operatorname{sub iso}(\mathcal{K})$ tal que:*

1. *para cada $\mathbf{S} \in \mathcal{S}$ se tiene que $\operatorname{aut}(\mathbf{S}) \subseteq \mathcal{F}$,*
2. *para cada $\mathbf{A} \in S(\mathcal{K}) - \mathcal{S}$ hay $\gamma, \gamma^{-1} \in \mathcal{F}$ para algún isomorfismo γ de $\mathbf{A}$ en el representante del tipo de isomorfismo de $\mathbf{A}$ en $\mathcal{S}$.*

Entonces $\mathcal{F}$ genera $\operatorname{sub iso}(\mathcal{K})$.

Demostración. Sea $\gamma \in \operatorname{sub iso}(\mathcal{K})$, entonces $\gamma : \mathbf{A}_0 \leq \mathbf{A} \to \mathbf{B}_0 \leq \mathbf{B}$ isomorfismo, con $\mathbf{A}, \mathbf{B} \in \mathcal{K}$.

Si $\mathbf{A}_0, \mathbf{B}_0 \in \mathcal{S}$, como no hay estructuras isomorfas en $\mathcal{S}$, $\mathbf{A}_0 = \mathbf{B}_0$, por lo tanto $\gamma \in \operatorname{aut}(\mathbf{A}_0) \subseteq \mathcal{F}$.

Si $\mathbf{A}_0 \in \mathcal{S}$ pero $\mathbf{B}_0 \notin \mathcal{S}$, luego $\mathbf{A}_0$ es el representante de $\mathbf{B}_0$ en $\mathcal{S}$, y por lo tanto hay un isomorfismo $\delta : \mathbf{B}_0 \to \mathbf{A}_0$ tal que $\delta, \delta^{-1} \in \mathcal{F}$. Claramente $\delta\gamma = \lambda \in \operatorname{aut}(\mathbf{A}_0) \subseteq \mathcal{F}$, por lo tanto $\gamma = \delta^{-1}\lambda$.

Si $\mathbf{A}_0, \mathbf{B}_0 \notin \mathcal{S}$, hay $\mathbf{C}_0 \in \mathcal{S}$ que es representante de $\mathbf{A}_0$ y $\mathbf{B}_0$. Luego hay $\delta : \mathbf{A}_0 \to \mathbf{C}_0$ y $\delta' : \mathbf{B}_0 \to \mathbf{C}_0$ tales que $\delta, \delta^{-1}, \delta', \delta'^{-1} \in \mathcal{F}$. Claramente $\delta'\gamma\delta^{-1} = \lambda \in \operatorname{aut}(\mathbf{C}_0) \subseteq \mathcal{F}$, por lo tanto $\gamma = \delta'^{-1}\lambda\delta$. $\square$

Corolario 29. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden tales que $R \in \mathcal{L}' - \mathcal{L}$ es un símbolo de relación n-ario y $\mathcal{K}$ un conjunto normal de $\mathcal{L}'$-estructuras. Supongamos $\mathcal{F}$ es como en el Teorema 28 para la familia $\mathcal{K}_{\mathcal{L}}$. Entonces si cada $\gamma \in \mathcal{F}$ preserva R, se tiene que R es definible por una $\mathcal{L}$-fórmula abierta en $\mathcal{K}$.*

Demostración. Directa ya que la composición de funciones que preservan R, trivialmente también preserva R. Luego aplicando el Teorema28 y el Lema 27 queda probado. □

El Teorema 28 sugiere un subconjunto de isomorfismos que bastan para revisar definibilidad abierta a través del Corolario 29.

Algorithm 3.2 Chequeo de definibilidad abierta

$\mathcal{S} = \emptyset$
for $\mathbf{A} \in \mathcal{K}$, desde la mayor a la menor cardinalidad **do**
  $Sub = [\mathbf{C} : \mathbf{C} \in \mathbb{S}_{\mathcal{L}}(\mathbf{A}),$ de la mayor a menor cardinalidad$]$
  for $\mathbf{B} \in Sub$ **do**
    (iso, ce) = TIENEREPRESENTANTE($\mathbf{B}, \mathcal{S}$) ▷ hay un $\mathcal{L}$-iso de $\mathbf{B}$ en $\mathbf{S} \in \mathcal{S}$
    if ce $\neq$ Null **then**
      return ce ▷ ce es contraejemplo
    end if
    if iso **then**
      $Sub = [\mathbf{C} : \mathbf{C} \in Sub$ tal que $C \nsubseteq B]$
    else
      $\mathcal{S} = \mathcal{S} \cup \{\mathbf{B}\}$
      for $\alpha \in \mathrm{Aut}_{\mathcal{L}}(\mathbf{B})$ **do**
        if α no preserva R **then**
          return α ▷ α es contraejemplo
        end if
      end for
    end if
  end for
end for
return ▷ R es abierta-definible

function TIENEREPRESENTANTE($\mathbf{A}_0, \mathcal{S}$)
  for $\mathbf{S} \in \mathcal{S}$, con $|\mathbf{A}_0| = |\mathbf{S}|$ **do**
    if hay $\gamma : \mathbf{A}_0 \to \mathbf{S}$ $\mathcal{L}$-isomorfismo **then**
      if γ no es un $\mathcal{L} \cup \{R\}$-isomorfismo **then**
        return (True, γ) ▷ γ es contraejemplo
      else
        return (True, Null) ▷ γ es isomorfismo con el representante
      end if
    end if
  end for
  return (False, Null) ▷ No tiene representante
end function

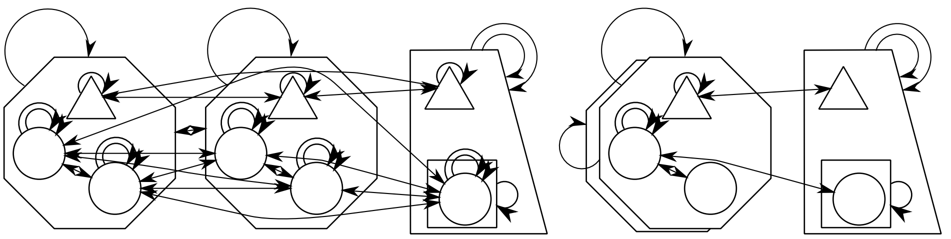
En el Algoritmo 3.2 se construye $\mathcal{S}$ a la vez que se van revisando los isomorfismos en $\mathcal{F}$. Para esto recorremos las estructuras en $\mathcal{K}$ de mayor a menor tamaño. Al momento de procesar una $\mathbf{A} \in \mathcal{K}$ lo primero que se hace es buscar un $\mathcal{L}$-isomorfismo γ entre $\mathbf{A}$ y un miembro $\mathcal{S}$. Si un tal γ existe, tanto γ como su inversa son revisados (miembros de $\mathcal{F}$ correspondientes al punto (2) del Teorema 28). Si no hay un tal γ, la estructura $\mathbf{A}$ es agregada a $\mathcal{S}$ y sus $\mathcal{L}$-automorfismos son revisados (miembros de $\mathcal{F}$ correspondientes al punto (1) del Teorema 28). A continuación las subestructuras de $\mathbf{A}$ son procesadas de la misma manera.

Notar que en la linea 14 no se necesita revisar si los automorfismos son $\mathcal{L} \cup \{R\}$-automorfismos ya que, como se recorren todos su inversa también es revisada por preservación.

Además, una vez que una subestructura resulta ser $\mathcal{L} \cup \{R\}$-isomorfa a un representante en $\mathcal{S}$, sabemos que todas sus subestructuras ya tienen representante en $\mathcal{S}$ y que los isomorfismos internos preservan. Esto permite disminuir la cantidad de subestructuras a revisar, como se ve en la linea 10.

Para esquematizar la ganancia obtenida en cuanto al chequeo de una menor cantidad de morfismos podemos observar en la Figura 3.3.1a el conjunto $\mathcal{K}$ junto con todas los morfismos que deberían ser revisados según el Teorema 14. Mientras que en la Figura 3.3.1b, se puede ver los morfismos que bastan para comprobar definibilidad abierta según el Corolario 29 a la manera del algoritmo anterior.

Notar que en la Figura 3.3.1b, la aplicación del Algoritmo 3.1 (Preprocesamiento) genera que uno de los octágonos aparezca como *sombra* del otro y baste solo revisar el isomorfismo entre ellos por el Lema 26.



(a) Morfismos a revisar según el Teorema 14

(b) Morfismos que bastan según el Corolario 29

Figura 3.3.1

3.4. Definibilidad abierta-positiva

A partir del Teorema 16, habría que chequear todo homomorfismo entre subestructuras de estructuras de $\mathcal{K}$, como en el siguiente lema.

Lema 30. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden tales que $R \in \mathcal{L}' - \mathcal{L}$ es un símbolo de relación n-ario y $\mathcal{K}$ un conjunto normal de $\mathcal{L}'$-estructuras. Entonces si cada $\gamma \in \operatorname{sub}\hom(\mathcal{K}_{\mathcal{L}})$ preserva R, R es definible por una $\mathcal{L}$-fórmula abierta positiva en $\mathcal{K}$.*

Demostración. Directo del Teorema 16.

En el siguiente teorema describimos un conjunto suficiente de morfismos para generar $\operatorname{sub}\hom(\mathcal{K}_{\mathcal{L}})$. Notar que, nuevamente, al tomar $\mathcal{K}$ como un conjunto normal de $\mathcal{L}$-estructuras sin estructuras isomorfas estamos pensando en la aplicación previa del Algoritmo 3.1, basado en el Lema 26. □

Teorema 31. *Sea $\mathcal{K}$ un conjunto normal de $\mathcal{L}$-estructuras donde no hay estructuras isomorfas. Sea $\mathcal{S} \subseteq \mathbb{S}(\mathcal{K})$ tal que contiene exactamente un representante por cada tipo de isomorfismo en $\mathbb{S}(\mathcal{K})$. Sea $\mathcal{H} \subseteq \operatorname{sub}\hom(\mathcal{K})$ tal que:*

- *$\mathcal{F} \subseteq \mathcal{H}$ con $\mathcal{F}$ como en el Teorema 28,*
- *para cada $\mathbf{A}, \mathbf{B} \in \mathcal{S}$ todo $h : \mathbf{A} \to \mathbf{B}$ homomorfismo sobreyectivo está en $\mathcal{H}$.*

Entonces $\mathcal{H}$ genera $\operatorname{sub}\hom(\mathcal{K})$.

Demostración. Sea $h \in \operatorname{sub}\hom(\mathcal{K})$, entonces $h : \mathbf{A}_0 \leq \mathbf{A} \to \mathbf{B}_0 \leq \mathbf{B}$ homomorfismo, con $\mathbf{A}, \mathbf{B} \in \mathcal{K}$.

Si $\mathbf{A}_0, h(\mathbf{A}_0) \in \mathcal{S}$, h es homomorfismo sobreyectivo entre estructuras de $\mathcal{S}$, por lo que $h \in \mathcal{H}$.

Si $\mathbf{A}_0 \in \mathcal{S}$ pero $h(\mathbf{A}_0) \notin \mathcal{S}$, hay $\mathbf{S}$ representante de $h(\mathbf{A}_0)$ en $\mathcal{S}$ y un isomorfismo $\delta : h(\mathbf{A}_0) \to \mathbf{S}$ tal que $\delta, \delta^{-1} \in \mathcal{F} \subseteq \mathcal{H}$. Luego $\delta^{-1}h = f$ donde f es un homomorfismo claramente sobreyectivo de $\mathbf{A}_0$ en $\mathbf{S}$, por lo que $f \in \mathcal{H}$. Finalmente $h = \delta f$.

Si $\mathbf{A}_0, h(\mathbf{A}_0) \notin \mathcal{S}$, hay respectivos $\mathbf{S}, \mathbf{S}' \in \mathcal{S}$ representantes de $\mathbf{A}_0$ y $h(\mathbf{A}_0)$. Luego hay $\delta : \mathbf{A}_0 \to \mathbf{S}$ y $\delta' : h(\mathbf{A}_0) \to \mathbf{S}'$ tales que $\delta, \delta^{-1}, \delta', \delta'^{-1} \in \mathcal{F} \subseteq \mathcal{H}$. Luego $\delta' h \delta^{-1} = f$ donde f es un homomorfismo claramente sobreyectivo de $\mathbf{S}$ en $\mathbf{S}'$, por lo que $f \in \mathcal{H}$. Finalmente $h = \delta'^{-1} f \delta$. □

Corolario 32. *Sean $\mathcal{L} \subseteq \mathcal{L}'$ lenguajes de primer orden tales que $R \in \mathcal{L}' - \mathcal{L}$ es un símbolo de relación n-ario y $\mathcal{K}$ un conjunto normal de $\mathcal{L}'$-estructuras. Supongamos $\mathcal{F}$ es como en el Teorema 31 para la familia $\mathcal{K}_{\mathcal{L}}$. Entonces si cada $\gamma \in \mathcal{F}$ preserva R, se tiene que R es definible por una $\mathcal{L}$-fórmula abierta positiva en $\mathcal{K}$.*

Demostración. Directa ya que la composición de funciones que preservan R, trivialmente también preserva R. Luego aplicando el Teorema31 y el Lema 30 queda probado. □

El Teorema 31 sugiere un subconjunto de homomorfismos que bastan para revisar definibilidad abierta positiva a través del Corolario 32. A su vez, a la hora de buscar homomorfismos entre estructuras aprovechamos el siguiente resultado, que nos permite organizar la búsqueda de homomorfismos biyectivos entre dos subestructuras en un único sentido.

Teorema 33 (Las estructuras finitas cumplen la propiedad de Cantor-Bernstein)**.** *Dadas $\mathbf{A}$ y $\mathbf{B}$ estructuras finitas, si hay $f : A \to B$ y $g : B \to A$ homomorfismos inyectivos entonces hay un $\gamma : \mathbf{A} \to \mathbf{B}$ isomorfismo.*

Demostración. Sean $\mathbf{A}, \mathbf{B}$ estructuras finitas y $f : A \to B$ y $g : B \to A$ homomorfismos inyectivos. Notar que gf es una permutación de $\mathbf{A}$, y como $\mathbf{A}$ es finito hay $k \geq 1$ tal que $(gf)^k = \mathrm{Id}_A$. Luego como $g^{-1} = f(gf)^{k-1}$ vemos que g^{-1} es homomorfismo. □

En el Algoritmo 3.3 se construye $\mathcal{S}$ a la vez que se van revisando los homomorfismos en $\mathcal{F}$. Para esto recorremos las estructuras en $\mathcal{K}$ de mayor a menor tamaño. Al momento de procesar una $\mathbf{A} \in \mathcal{K}$, mientras que para definibilidad abierta lo primero que se hacia era buscar un $\mathcal{L}$-isomorfismo para cada $\mathbf{S} \in \mathcal{S}$, ahora se buscan homomorfismos biyectivos primero, chequeándolos por preservación. Esto da lugar a tres casos:

1. Uno de estos homomorfismos biyectivos es un $\mathcal{L}$-isomorfismo.

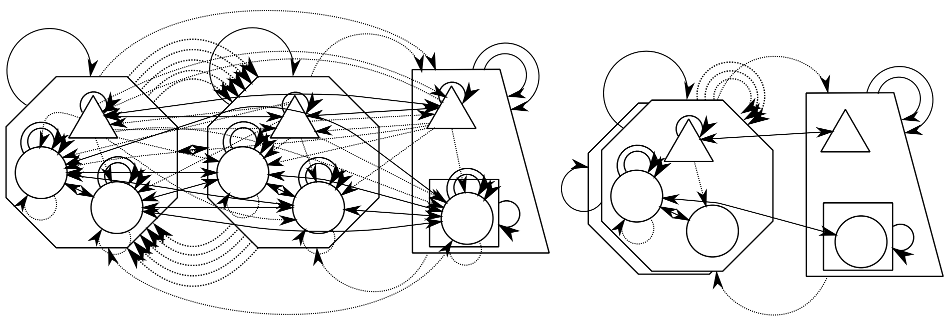
2. Hay homomorfismos biyectivos de $\mathbf{A}$ en $\mathbf{S}$ pero ninguno es un $\mathcal{L}$-isomorfismo.

3. No hay homomorfismos biyectivos de $\mathbf{A}$ en $\mathbf{S}$.

En el primer caso se procede como en el Algoritmo 3.2. En el segundo caso podemos estar seguros de que no hay homomorfismos biyectivos de $\mathbf{S}$ en $\mathbf{A}$, ya que por el Teorema 33, debería haber un $\mathcal{L}$-isomorfismo entre ellos. Mientras que en el tercer caso deberemos buscar homomorfismos biyectivos de $\mathbf{S}$ en $\mathbf{A}$, para revisar su preservación.

Esto nos permite estar seguros de que todos los homomorfismos biyectivos entre estructuras de $\mathcal{S}$ han sido chequeados. Finalmente lo único que falta es chequear los homomorfismos sobreyectivos pero no inyectivos entre estructuras de $\mathcal{S}$. Para lo cual basta con buscar homomorfismos sobreyectivos de $\mathbf{A}$ en $\mathbf{B}$ tales que $|A| > |B|$.

Para esquematizar la ganancia obtenida en cuanto al chequeo de una menor cantidad de morfismos podemos observar en la Figura 3.4.1a el conjunto $\mathcal{K}$ junto con todas los morfismos que deberían ser revisados según el Teorema 16, donde las flechas punteadas son los homomorfismos. Mientras que en la Figura 3.4.1b, se puede ver los morfismos que bastan para comprobar definibilidad abierta según el Teorema 31 a la manera del algoritmo anterior.



(a) Morfismos a revisar según el Teorema 16

(b) Morfismos que bastan según el Teorema 31

Figura 3.4.1

Algorithm 3.3 Chequeo de definibilidad abierta-positiva

```
$\mathcal{S} = \emptyset$
for $\mathbf{A} \in \mathcal{K}$, desde la mayor a la menor cardinalidad do
    $Sub = [\mathbf{C} : \mathbf{C} \in \mathbb{S}_{\mathcal{L}}(\mathbf{A})$, de la mayor a menor cardinalidad$]$
    for $\mathbf{B} \in Sub$ do
        (iso,ce) = tieneRepresentante($\mathbf{B}, \mathcal{S}$)    ▷ hay un $\mathcal{L}$-iso de $\mathbf{B}$ en $\mathbf{S} \in \mathcal{S}$
        if ce $\neq$ null then
            return ce    ▷ ce es contraejemplo
        end if
        if iso then
            $Sub = [\mathbf{C} : \mathbf{C} \in Sub$ tal que $C \nsubseteq B]$
        else
            $\mathcal{S} = \mathcal{S} \cup \{\mathbf{B}\}$
            for $\alpha \in \mathrm{Aut}_{\mathcal{L}}(\mathbf{B})$ do
                if $\alpha$ no preserva $R$ then
                    return $\alpha$    ▷ $\alpha$ es contraejemplo
                end if
            end for
        end if
    end for
end for
for $\mathbf{A} \in \mathcal{S}$ do
    for $\mathbf{B} \in \mathcal{S}$, con $|\mathbf{B}| < |\mathbf{A}|$ do
        for $\gamma : \mathbf{A} \to \mathbf{B}$ con $\gamma$ $\mathcal{L}$-homomorfismo sobreyectivo do
            if $\gamma$ no preserva $R$ then
                return $\gamma$    ▷ $\gamma$ es contraejemplo
            end if
        end for
    end for
end for
return    ▷ $R$ es definible por una abierta positiva

function tieneRepresentante($\mathbf{A}_0, \mathcal{S}$)
    for $\mathbf{S} \in \mathcal{S}$, con $|\mathbf{A}_0| = |\mathbf{S}|$ do
        bihomos =[$\gamma$: $\gamma : \mathbf{A}_0 \to \mathbf{S}$ con $\gamma$ $\mathcal{L}$-homomorfismo biyectivo
        for $\gamma \in$ bihomos do
            if $\gamma$ es un $\mathcal{L}$-isomorfismo, con $\mathbf{S} \in \mathcal{S}$ then
                if $\gamma$ no es un $\mathcal{L} \cup \{R\}$-isomorfismo then
                    return (True, $\gamma$)    ▷ $\gamma$ es contraejemplo
                else
                    return (True, Null)    ▷ $\gamma$ es isomorfismo con el representante
                end if
            else if $\gamma$ no preserva $R$ then
                return (False, $\gamma$)    ▷ $\gamma$ es contraejemplo
            end if
        end for
        if bihomos = [] then
            for $\gamma : \mathbf{S} \to \mathbf{A}_0$ con $\gamma$ $\mathcal{L}$-homomorfismo biyectivo do
                if $\gamma$ no preserva $R$ then
                    return (False, $\gamma$)    ▷ $\gamma$ es contraejemplo
                end if
            end for
        end if
        return (False, Null)    ▷ No tiene representante
    end for
end function
```

3.5. Definibilidad de primer orden

Por el Teorema 17, luego de haber aplicado el Algoritmo 3.1 para preprocesamiento, solo basta con revisar los automorfimos para $\mathbf{A} \in \mathcal{K}$, como hacemos en el Algoritmo 3.4.

Algorithm 3.4 Chequeo de definibilidad de primer orden

for $\mathbf{A} \in \mathcal{K}$ **do**
  for $\alpha \in \mathrm{Aut}_{\mathcal{L}}(\mathbf{A})$ **do**
    if α no es un $\mathcal{L} \cup \{R\}$-automorfismo **then**
      return α ▷ α es contraejemplo
    end if
  end for
end for
return ▷ R es definible en primer orden

3.6. Definibilidad existencial

Por el Teorema 19, luego de aplicar el Algoritmo 3.1 de preprocesamiento, solo basta con revisar los embeddings entre $\mathbf{A}, \mathbf{B} \in \mathcal{K}$, como hacemos en el Algoritmo 3.5.

Algorithm 3.5 Chequeo de definibilidad existencial

for $\mathbf{A} \in \mathcal{K}$ **do**
  for $\mathbf{B} \in \mathcal{K}$ **do**
    for $\gamma \in \mathrm{Emb}_{\mathcal{L}}(\mathbf{A}, \mathbf{B})$ **do**
      if γ no es un $\mathcal{L} \cup \{R\}$-embedding **then**
        return γ ▷ γ es contraejemplo
      end if
    end for
  end for
end for
return ▷ R es existencial definible

3.7. Definibilidad existencial-positiva

Por el Teorema 21, luego de aplicar el Algoritmo 3.1 de preprocesamiento, solo basta con revisar los homomorfismos cada par $\mathbf{A}, \mathbf{B} \in \mathcal{K}$, como hacemos en el Algoritmo 3.6.

Algorithm 3.6 Chequeo de definibilidad existencial-positiva

for $\mathbf{A} \in \mathcal{K}$ **do**
  for $\mathbf{B} \in \mathcal{K}$ **do**
    for $\alpha \in \mathrm{Hom}_{\mathcal{L}}(\mathbf{A}, \mathbf{B})$ **do**
      if h no preserva R **then**
        return h ▷ h es contraejemplo
      end if
    end for
  end for
end for
return ▷ R es existencial positiva definible

3.8. Detección de homomorfismos

Para la detección de homomorfismos modelizamos el problema como un CSP, y utilizamos Minion ([Gent, Jefferson y Miguel 2006]) como solucionador.

3.8.1. CSP

Un CSP (constraint satisfaction problem) (ver, e.g., [Russell y Norvig 2010, Capítulo 6]) es definido como una terna $\langle X, D, C\rangle$, donde

$$X = \{X_1, \ldots, X_n\} \text{ es un conjunto de variables}$$

$$D = \{D_1, \ldots, D_n\} \text{ es un conjunto con los respectivos dominios de los valores}$$

$$C = \{C_1, \ldots, C_m\} \text{ es un conjunto de restricciones}$$

Cada variable X_i se mueve en los valores del respectivo dominio no vacío D_i. Cada restricción $C_j \in C$ es un par $\langle t_j, R_j\rangle$, donde t_j es una k-upla de variables y R_j es una relación k-aria en el correspondiente dominio de cada variable. Una valuación sobre las variables es una función desde un subconjunto de las variables a un particular conjunto de valores en los correspondientes dominios de valores. Una valuación v satisface la restricción $\langle t_j, R_j\rangle$ si los valores asignados a las variables t_j satisfacen la relación R_j.

Una valuación es consistente si no viola ninguna de las restricciones. Una valuación es completa si incluye todas las variables. Una valuación es solución si es consistente y completa. En ese caso diremos que la valuación resuelve el CSP.

3.8.2. CSP para calcular homomorfismos

Sean $\mathbf{A}$,$\mathbf{B}$ $\mathcal{L}$-estructuras, queremos encontrar homomorfismos de $\mathbf{A}$ en $\mathbf{B}$ resolviendo una instancia de CSP. Tomamos

$$X = \{X_i : i \in A\}$$

$$D_i = B$$

$$C = C^R \cup C^f$$

donde C^R es tal que para cada $R \in \mathcal{L}$ n-aria, para cada $(a_1, \ldots, a_n) \in R^{\mathbf{A}}$, se da que $\left((X_{a_1}, \ldots, X_{a_n}), R^{\mathbf{B}}\right) \in C^R$ y C^f es tal que para cada $f \in \mathcal{L}$ n-aria, para cada $(a_1, \ldots, a_n, a') \in \operatorname{graph}\left(f^{\mathbf{A}}\right)$, se da que $\left((X_{a_1}, \ldots, X_{a_n}, X_{a'}), \operatorname{graph}\left(f^{\mathbf{B}}\right)\right) \in C^f$.

Supongamos que $V : X \to B$ es una valuación solución de (X, D, C), entonces el homomorfismo γ determinado por V será $\gamma(a) = V(X_a)$.

Si además quisiéramos que el homomorfismo fuera inyectivo, bastaría con agregar una restricción:

$$((X_1, \ldots, X_m), \text{para todo } (i, j) \text{ con } i \neq j \text{ y } i, j \in \{1, \ldots, m\} \text{ se da } X_i \neq X_j)$$

donde $X_1, \ldots, X_m$ son todas las variables en X.

Para que fuera sobreyectivo bastaría agregar la siguiente restricción:

$$\left((X_1, \ldots, X_m), \bigcup_{i=1}^{m} \{X_i\} = B\right)$$

En el caso particular de la búsqueda de automorfismos, el lema 34 nos permite simplificar la búsqueda, encontrando endomorfismos inyectivos.

Lema 34. *Sea* $\mathbf{A}$ *una estructura finita, entonces todo endomorfismo inyectivo de* $\mathbf{A}$ *es un automorfismo de* $\mathbf{A}$.

Demostración. Sea γ un endomorfismo inyectivo de $\mathbf{A}$. Como A es finito tenemos que γ es sobre y además $\gamma^{-1} = \gamma^k$ para algún $k \geq 1$. □

3.9. Generación de subestructuras

Dada una $\mathcal{L}$-estructura $\mathbf{A}$, generamos sus subestructuras recorriendo $\mathcal{P}(A)$, desde la mayor a la menor cardinalidad filtrando aquellos $A_0 \in \mathcal{P}(A)$ que no son cerrados bajo $\mathcal{L}$.

Además cuando detectamos que una subestructura $\mathbf{A}_0$ ya tiene representante en $\mathcal{S}$ durante nuestros algoritmos, disminuimos los subconjuntos a chequear, revisando solo $\mathcal{P}(A) - \mathcal{P}(A_0)$, ya que si $\mathbf{A}_0$ tiene representante en $\mathcal{S}$, entonces cada subestructura de $\mathbf{A}_0$ tiene representante en $\mathcal{S}$.

Capítulo 4

Álgebras de Lindenbaum

4.1. Definiciones e ideas básicas

Sea $\mathbf{A}$ una $\mathcal{L}$-estructura, y sea

$$\Sigma \in \left\{\mathrm{Fo}\left(\mathcal{L}\right), \mathrm{E}\left(\mathcal{L}\right), \mathrm{E}^{+}\left(\mathcal{L}\right), \mathrm{Op}\left(\mathcal{L}\right), \mathrm{Op}^{+}\left(\mathcal{L}\right)\right\}.$$

Notar que la colección de relaciones n-arias definibles en $\mathbf{A}$ por fórmulas en Σ es cerrada bajo uniones e intersecciones (y también bajo complementación cuando $\Sigma \in \{\mathrm{Fo}\left(\mathcal{L}\right), \mathrm{Op}\left(\mathcal{L}\right)\}$). Llamaremos *Álgebra de Lindenbaum* (de relaciones n-arias definibles por Σ en $\mathbf{A}$) al reticulado distributivo (o álgebra de Boole) resultante. Utilizaremos la siguiente notación:

- $\mathbf{Fo}_n(\mathbf{A}) =$ Álgebra de Boole de relaciones n-arias definibles en $\mathbf{A}$,
- $\mathbf{E}_n(\mathbf{A}) =$ Reticulado de relaciones n-arias definibles por existenciales en $\mathbf{A}$,
- $\mathbf{E}_n^{+}(\mathbf{A}) =$ Reticulado de relaciones n-arias definibles por existenciales positivas en $\mathbf{A}$,
- $\mathbf{Op}_n(\mathbf{A}) =$ Álgebra de Boole de relaciones n-arias definibles por abiertas en $\mathbf{A}$,
- $\mathbf{Op}_n^{+}(\mathbf{A}) =$ Reticulado de relaciones n-arias definibles por abiertas positivas en $\mathbf{A}$

Definimos:

$$\mathrm{sub\,hom}\left(\mathbf{A}\right) = \{\gamma : \gamma \text{ homomorfismo de } \mathbf{A}_0 \text{ en } \mathbf{B}_0 \text{ con } \mathbf{A}_0, \mathbf{B}_0 \in \mathbb{S}\left(\mathbf{A}\right)\}$$

$$\mathrm{sub\,iso}\left(\mathbf{A}\right) = \{\gamma : \gamma \text{ isomorfismo de } \mathbf{A}_0 \text{ en } \mathbf{B}_0 \text{ con } \mathbf{A}_0, \mathbf{B}_0 \in \mathbb{S}\left(\mathbf{A}\right)\}$$

Dado que estas álgebras podrían ser muy grandes utilizamos los siguientes resultados para representarlas mediante un subconjunto de sus elementos.

Teorema 35 (Teorema de representación de Stone)**.** *Toda álgebra de Boole finita $\mathbf{A}$ es isomorfa al álgebra definida por $\mathcal{P}\left(At\right)$ donde $At \subseteq A$ es el conjunto de átomos en $\mathbf{A}$.*

Teorema 36 (Teorema de representación de Birkhoff). *Todo reticulado distributivo finito* $\mathbf{L}$ *es isomorfo al reticulado de conjuntos descendientes del poset de elementos join-irreducibles en* $\mathbf{L}$.

Lema 37. *Dada un álgebra de Boole finita* $\mathbf{A}$, *un elemento es un átomo sii es join-irreducible.*

Los resultados anteriores nos permiten centrarnos únicamente en los elementos join-irreducibles de estas álgebras. Mientras que los Lemas 38 y 39, presentados a continuación, sugieren una manera clara de calcularlos.

Lema 38. *Una relación* $r \subseteq A^n$ *es join-irreducible en* $\mathbf{Op}(\mathbf{A})$ *sii hay* $\bar{a} \in A^n$ *tal que* $r = \{h(\bar{a}) : h \in \mathrm{sub\,iso}\,(\mathbf{A})\}$.

Demostración. Supongamos que $r \subseteq A^n$ es join-irreducible en $\mathbf{Op}(\mathbf{A})$. Como r es cerrado bajo $\mathrm{sub\,iso}\,(\mathbf{A})$ por pertenecer a $\mathbf{Op}(\mathbf{A})$, es claro que

$$r = \bigcup_{\bar{x} \in r} \{h\,(\bar{x}) : h \in \mathrm{sub\,iso}\,(\mathbf{A})\},$$

pero como r es join-irreducible hay una $\bar{a} \in r$ tal que $r = \{h\,(\bar{a}) : h \in \mathrm{sub\,iso}\,(\mathbf{A})\}$.

Supongamos que $r = \{h\,(\bar{a}) : h \in \mathrm{sub\,iso}\,(\mathbf{A})\}$ para algún $\bar{a} \in A^n$. Sean $r_1, \ldots, r_m \in \mathbf{Op}(\mathbf{A})$ tales que

$$r = r_1 \cup \cdots \cup r_m,$$

es claro que hay un r_j tal que $\bar{a} \in r_j$. Además como r_j es cerrado bajo $\mathrm{sub\,iso}\,(\mathbf{A})$ por pertenecer a $\mathbf{Op}(\mathbf{A})$, luego $r = r_j$. □

Lema 39. *Una relación* $r \subseteq A^n$ *es join-irreducible en* $\mathbf{Op}^+(\mathbf{A})$ *sii hay* $\bar{a} \in A^n$ *tal que* $r = \{h(\bar{a}) : h \in \mathrm{sub\,hom}\,(\mathbf{A})\}$.

Demostración. Igual a la del Lema 38 □

4.2. Algoritmos

Con la intención de simplificar la exposición, supondremos que $\mathcal{K}$ tiene una única estructura $\mathbf{A}$. La generalización es fácilmente deducible ya que los algoritmos son casi iguales a los presentados en el Capítulo 3.

Como un álgebra de Lindenbaum de relaciones definibles puede llegar a ser un objeto muy grande y difícil de manipular, nos basta generar los elementos join-irreducibles del álgebra según los Teoremas 35 para aquellas álgebras de Lindenbaum que son álgebras de Boole y los átomos (que también son los elementos join-irreducibles según el Lema 37) según el Teorema 36, para aquellas que son reticulados.

Para generar los elementos join-irreducibles del álgebra de Lindenbaum de relaciones de ancho n definibles por abiertas en $\mathbf{A}$ basta con obtener el conjunto $\mathcal{F}$ de morfismos que se chequean por preservación en los algoritmos del Capítulo 3 que genera $\mathrm{sub\,iso}\,(\mathbf{A})$ por el Teorema 28, y luego calcular $\{h(\bar{a}) : h \in \mathrm{sub\,iso}\,(\mathbf{A})\}$ para cada $\bar{a} \in A^n$, ya que por el Lema 38, todos los elementos join-irreducibles son de esta forma. De la misma manera podemos construir el álgebra de Lindenbaum de relaciones de ancho n definibles por abiertas positivas en $\mathbf{A}$, basándonos en el Teorema 31 y en el Lema 39.

Los algoritmos 4.1 y 4.2, generan el conjunto $\mathcal{F}$ definidos por los Teoremas 28 y 31.

Algorithm 4.1 Construcción de $\mathbf{Op}_n(\mathbf{A})$

$\mathcal{S} = \emptyset$
$\mathcal{F} = \emptyset$
$Sub = [\mathbf{C} : \mathbf{C} \in \mathbb{S}_{\mathcal{L}}(\mathbf{A}),$ de la mayor a menor cardinalidad$]$
for $\mathbf{B} \in Sub$ **do**
    iso $=$ BUSCAISO$(\mathbf{B}, \mathcal{S})$
    if iso $\neq$ Null **then**
        $\mathcal{F} = \mathcal{F} \cup \{\text{iso}\}$
        $Sub = [\mathbf{C} : \mathbf{C} \in Sub$ tal que $C \nsubseteq B]$
    else
        $\mathcal{S} = \mathcal{S} \cup \{\mathbf{B}\}$
        for $\alpha \in \mathrm{Aut}_{\mathcal{L}}(\mathbf{B})$ **do**
            $\mathcal{F} = \mathcal{F} \cup \{\alpha\}$
        end for
    end if
end for
return $\mathcal{F}$ ▷ R es abierta-definible

function BUSCAISO$(\mathbf{A}_0, \mathcal{S})$
    for $\mathbf{S} \in \mathcal{S}$, con $|\mathbf{A}_0| = |\mathbf{S}|$ **do**
        if hay $\gamma : \mathbf{A}_0 \to \mathbf{S}$ isomorfismo **then**
            return γ ▷ γ es isomorfismo con el representante
        end if
    end for
    return Null ▷ No tiene representante
end function

Algorithm 4.2 Construcción de $\mathbf{Op}_n^+(\mathbf{A})$

$\mathcal{S} = \emptyset$
$\mathcal{F} = \emptyset$
$Sub = [\mathbf{C} : \mathbf{C} \in \mathbb{S}_{\mathcal{L}}(\mathbf{A})$, de la mayor a menor cardinalidad$]$
for $\mathbf{B} \in Sub$ **do**
    (iso,$\mathcal{F}_0$) $=$ BUSCAISO_BIHOMOS$(\mathbf{B}, \mathcal{S})$
    $\mathcal{F} = \mathcal{F} \cup \mathcal{F}_0$
    if iso $\neq$ null **then**
        $\mathcal{F} = \mathcal{F} \cup \{\text{iso}\}$
        $Sub = [\mathbf{C} : \mathbf{C} \in Sub$ tal que $C \nsubseteq B]$
    else
        $\mathcal{S} = \mathcal{S} \cup \{\mathbf{B}\}$
        for $\alpha \in \mathrm{Aut}_{\mathcal{L}}(\mathbf{B})$ **do**
            $\mathcal{F} = \mathcal{F} \cup \{\alpha\}$
        end for
    end if
end for
for $\mathbf{A} \in \mathcal{S}$ **do**
    for $\mathbf{B} \in \mathcal{S}$, con $|\mathbf{B}| < |\mathbf{A}|$ **do**
        for $\gamma : \mathbf{A} \to \mathbf{B}$ con γ homomorfismo sobreyectivo **do**
            $\mathcal{F} = \mathcal{F} \cup \{\gamma\}$
        end for
    end for
end for
return $\mathcal{F}$ ▷ R es definible por una abierta positiva

function BUSCAISO_BIHOMOS$(\mathbf{A}_0, \mathcal{S})$
    $\mathcal{F}_0 = \emptyset$
    for $\mathbf{S} \in \mathcal{S}$, con $|\mathbf{A}_0| = |\mathbf{S}|$ **do**
        $bihomo = False$
        for $\gamma : \mathbf{A}_0 \to \mathbf{S}$ con γ homomorfismo biyectivo **do**
            $bihomo = True$
            if γ es un $\mathcal{L}$-isomorfismo, con $\mathbf{S} \in \mathcal{S}$ **then**
                return $(\gamma, \emptyset)$ ▷ γ es isomorfismo con el representante
            else
                $\mathcal{F}_0 = \mathcal{F}_0 \cup \{\gamma\}$
            end if
        end for
        if $\neg bihomo$ **then**
            for $\gamma : \mathbf{S} \to \mathbf{A}_0$ con γ homomorfismo biyectivo **do**
                $\mathcal{F}_0 = \mathcal{F}_0 \cup \{\gamma\}$
            end for
        end if
        return $(\text{Null}, \mathcal{F}_0)$ ▷ No tiene representante
    end for
end function

Una vez calculado $\mathcal{F}$ basta con calcular $\{h(\bar{a}) : h \in \mathcal{F}\}$ para cada $\bar{a} \in A^n$, ya que $\mathcal{F}$ genera al conjunto correspondiente al tipo de definibilidad. Esto lo hacemos con el Algoritmo 4.3, lo cual ya nos da los elementos join-irreducibles o los átomos, según el tipo de definibilidad en cuestión.

Algorithm 4.3 Saturación por un conjunto de morfismos

$\mathcal{A} = \emptyset$
for $\bar{a} \in A^n$ **do**
    $\mathcal{A} = \mathcal{A} \cup \{\textsc{clausura}(\bar{a}, \mathcal{E})\}$
end for
return $\mathcal{A}$ ▷ R es abierta-definible

function $\textsc{clausura}(\bar{a}, \mathcal{E})$
    $e = \{\bar{a}\}$
    while e crezca **do**
        for $\bar{a} \in e$ **do**
            for $\gamma \in \mathcal{E}$ **do**
                $e = e \cup \left\{\gamma\left(\bar{b}\right)\right\}$
            end for
        end for
    end while
    return e
end function

Para los demás tipos de definibilidad, el procedimiento seria análogo. Generar $\mathcal{F}$ basados en los algoritmos del capítulo 3 y saturar según el Algoritmo 4.3.

4.3. Ejemplos y aplicación

En esta sección mostramos cómo la utilización del paquete permitió conjeturar una caracterización de las relaciones binarias definibles por existenciales positivas en reticulados distributivos.

4.3.1. Utilizando el paquete Definability para SageMath

A manera de prueba de las herramientas utilizamos Definability para recorrer todas las relaciones binarias definibles en el reticulado $\mathbf{2} \times \mathbf{2}$, representado en la figura 4.3.1.

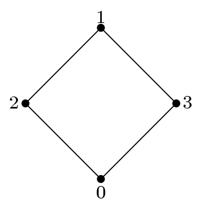


Figura 4.3.1: Reticulado $\mathbf{2} \times \mathbf{2}$

Al correr el algoritmo de generación de los elementos join-irreducibles de $\mathbf{E}_2\left(\mathbf{2} \times \mathbf{2}\right)$, Definability nos devolvió las siguientes relaciones:

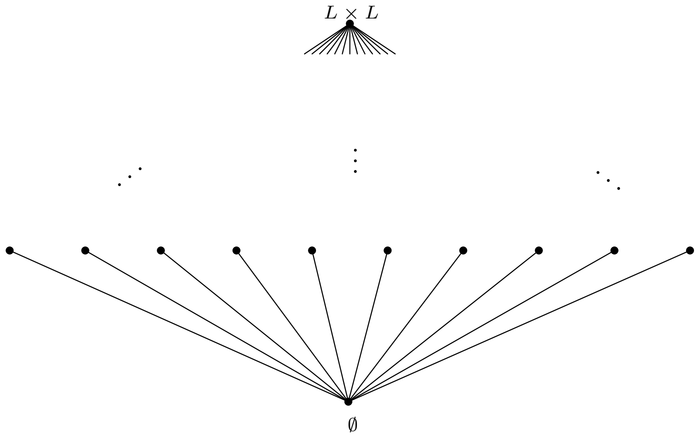


Figura 4.3.2: Reticulado $\mathbf{E}_2(\mathbf{2}\times\mathbf{2})$

- $\{(0,0)\}$
- $\{(0,1)\}$
- $\{(0,2),(0,3)\}$
- $\{(1,0)\}$
- $\{(1,1)\}$
- $\{(1,2),(1,3)\}$
- $\{(2,0),(3,0)\}$
- $\{(2,1),(3,1)\}$
- $\{(2,2),(3,3)\}$
- $\{(2,3),(3,2)\}$

Aquí se puede notar las ventajas de utilizar el Teorema 36, ya que el reticulado descripto por estos 10 elementos (representado en la figura ??) tiene cardinalidad $2^{10} = 1024$, ya que son todas las uniones entre elementos join-irreducibles.

En cuanto a los elementos join-irreducibles de $\mathbf{E}_2^+(\mathbf{2}\times\mathbf{2})$ nos devolvió:

- $\{(0,0),(1,1),(2,2),(3,3)\}$
- $\{(0,0),(0,1),(0,2),(0,3),(1,1),(2,1),(2,2),(3,1),(3,3)\}$
- $\{(0,0),(1,0),(1,1),(1,2),(1,3),(2,0),(2,2),(3,0),(3,3)\}$
- $\{(0,0),(0,1),(0,2),(0,3),\ldots,(3,0),(3,1),(3,2),(3,3)\}$

Las cuales pueden ser interpretadas respectivamente como:

- Δ
- $\{(x,y) : x \leq y\}$

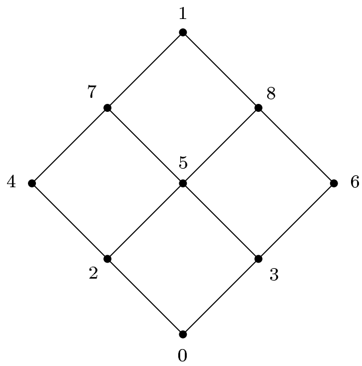


Figura 4.3.4: Reticulado $\mathbf{3} \times \mathbf{3}$

- $\{(x, y) : x \geq y\}$
- A^2

Lo que mediante el Teorema 36, define un subreticulado de $\mathbf{E}_2\,(\mathbf{2} \times \mathbf{2})$ anterior, mucho más pequeño, como se puede ver en la Figura 4.3.3, donde los nodos sin rellenar son los elementos join-irreducibles devueltos por el paquete.

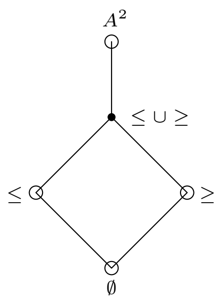


Figura 4.3.3: Reticulado de relaciones binarias definibles por existenciales positivas en $\mathbf{2} \times \mathbf{2}$

Luego probamos en el reticulado $\mathbf{3} \times \mathbf{3}$ representado en la figura 4.3.4.

Al interpretar los elementos join-irreducibles de $\mathbf{E}_2^+\,(\mathbf{3} \times \mathbf{3})$ calculados por el paquete, nos dio nuevamente:

- Δ
- $\{(x, y) : x \leq y\}$
- $\{(x, y) : x \geq y\}$
- A^2

Lo que representa un reticulado isomorfo al presentado en 4.3.3. Esto nos llevo a conjeturar el Teorema 43, que probamos a continuación.

4.3.2. Caracterización de las relaciones binarias definibles por existenciales positivas en reticulados distributivos

Lema 40. *Sea* $\mathbf{L}$ *un reticulado y* R *una relación binaria no vacía sobre* L *preservada por endomorfismos en* $\mathbf{L}$. *Entonces vale que* $\Delta^L \subseteq R$.

Demostración. Esto es una consecuencia directa de que para cada $a \in L$ la función de L en L que vale constantemente a es un endomorfismo. □

Teorema 41 (Teorema del filtro primo)**.** *Sea* $\langle L, \vee, \wedge \rangle$ *un reticulado distributivo y* F *un filtro. Supongamos* $x \in L - F$. *Entonces hay un filtro primo* P *tal que* $x \notin P$ *y* $F \subseteq P$.

Lema 42. *Sea* $\mathbf{L}$ *un reticulado distributivo y* R *una relación binaria no vacía sobre* L *preservada por endomorfismos en* $\mathbf{L}$. *Si hay* $(a,b) \in R$ *tal que* $a \not\leq b$ *(respectivamente* $b \not\leq a$*), entonces* $\{(x,y) : x \geq y\} \subseteq R$ *(respectivamente* $\{(x,y) : x \leq y\} \subseteq R$*).*

Demostración. Fijamos $(a,b) \in R$ tal que $a \not\leq b$, y sean $c, d \in L$ tales que $c \geq d$. Veremos que $(c,d) \in R$. Por el Teorema 41 hay un filtro primo P que contiene al filtro generado por a y además $b \notin P$. Definimos $h : L \to L$ por

$$h\left(x\right) = \begin{cases} c & \text{si } x \in P, \\ d & \text{si } x \notin P. \end{cases}$$

Es fácil ver que h es un endomorfismo. Finalmente como $(a,b) \in R$ y h preserva R, $(h\left(a\right), h\left(b\right)) = (c,d) \in R$. □

Teorema 43. *Sea* $\mathbf{L}$ *un reticulado distributivo y* R *una relación binaria sobre* L *definible por un fórmula existencial positiva en* $\mathbf{L}$. *Se da una de las siguientes:*

- $R = \Delta$,
- $R = \{(x,y) : x \leq y\}$,
- $R = \{(x,y) : x \geq y\}$,
- $R = \{(x,y) : x \leq y\} \cup \{(x,y) : x \geq y\}$,
- $R = L \times L$.

Demostración. Si $R \subseteq \Delta$, por el Lema 40 $R = \Delta$.

Si $R \subseteq \{(x,y) : x \leq y\}$, pero $R \nsubseteq \Delta$, hay $(a,b) \in R$ tales que $a < b$, luego $a \ngeq b$ y por el Lema 42 $R = \{(x,y) : x \leq y\}$. Análogo para $R \subseteq \{(x,y) : x \geq y\}$.

Si $R \nsubseteq \{(x,y) : x \leq y\}$ y $R \nsubseteq \{(x,y) : x \geq y\}$, aplicando dos veces el Lema 42 $\{(x,y) : x \leq y\} \cup \{(x,y) : x \geq y\} \subseteq R$. Si además $(a,b) \in R$ pero $a \not\leq b$ y $a \ngeq b$, tomamos $(c,d) \in R$, si son comparables ya están en R por lo anterior. Si son incomparables aplico dos veces el Teorema 41 y tomamos dos filtros primos P y Q tales que $a \in P$ pero $b \notin P$ y $b \in Q$ pero $a \notin Q$. Ahora definimos la siguiente función

$$h\left(x\right) = \begin{cases} c \vee d & \text{si } x \in P \cup Q \\ c & \text{si } x \in P - Q \\ d & \text{si } x \in Q - P \\ c \wedge d & \text{si } x \notin P \text{ y } x \notin Q \end{cases}$$

que fácilmente puede verse que es un endomorfismo. Por lo tanto como $(a, b) \in R$ y h preserva R, $(h(a), h(b)) = (c, d) \in R$. Por lo tanto $R = L \times L$. $\square$

Capítulo 5

Documentación de Definability

5.1. Introducción

Definability es un paquete que desarrollamos para SageMath, un software matemático licenciado bajo la GPL. Nuestro paquete implementa los algoritmos vistos en las secciones anteriores, permitiendo, dada una clase de estructuras, decidir definibilidad o generar las álgebras de Lindenbaum de relaciones definibles.

5.2. Arquitectura del paquete

Nuestro paquete se basa en parte en las librerías desarrolladas por Peter Jipsen para utilizar Minion, Universal Álgebra Calculator, Mace4 y Prover9 desde SageMath que se pueden obtener en http://math.chapman.edu/∼jipsen/sagepkg/.

El paquete se desarrolla en https://github.com/pablogventura/definability.

5.2.1. Programas externos necesarios

Definability, así como SageMath han sido desarrollados en Python 2.7, por lo que necesitan del interprete instalado. Dado que Definability es un paquete desarrollado para SageMath, necesita de una instalación de éste. Además, utiliza interfaces con otros programas. Particularmente utiliza Minion, y LADR (interfaz de linea de comandos de Prover9, Mace4, y otros programas). Definability ha sido probado en versiones particulares de estos programas y no se sabe si funcionara en otras. Las versiones utilizadas son las siguientes:

- Python 2.7.9,
- SageMath 6.7,
- Minion 1.8,
- LADR versión de noviembre de 2009.

En la siguiente sección explicaremos nuestra motivación detrás de desarrollar un paquete para SageMath y en las siguientes dos secciones explicaremos brevemente qué hacen los programas Minion y LADR.

5.2.1.1. SageMath

SageMath es un software matemático libre licenciado bajo la GPL, con el objetivo de ser una alternativa libre a Magma, Maple, Mathematica y Matlab.

SageMath es un software en pleno desarrollo que incluye herramientas para trabajar con reticulados y muchas otras estructuras algebraicas por lo que nos pareció interesante utilizarlo como base para nuestra herramienta.

5.2.1.2. Minion

Minion es un solucionador de CSP, rápido y escalable respecto del incremento del tamaño del problema. Es un solucionador de propósito general con un lenguaje de entrada muy expresivo. Minion es software libre, licenciado bajo la GPL 2.

En nuestro caso utilizamos la interfaz con Minion a la hora de resolver nuestros CSP para buscar morfismos.

5.2.1.3. LADR

LADR es la interfaz de linea de comandos para Prober9 y Mace4. Prover9 es un probador automático de teoremas para primer orden y lógica ecuacional y Mace4 es un software capaz de encontrar modelos finitos y contraejemplos.

En nuestra herramienta solo se utiliza la interfaz con Mace4. Esto es para generar automáticamente modelos de teorías de primer orden.

5.2.2. Módulos y organización del código fuente

Cada archivo .py en la carpeta src implementa los siguientes módulos:

config Maneja las rutas para acceder a Minion y a LADR.

minion Implementa la interfaz con Minion.

mace4 Implementa la interfaz con LADR, para utilizar Mace4.

fotype Implementa los tipos de primer orden

model Implementa la clase FO_Model que representa las estructuras de primer orden.

fotheory Implementa las teorías de primer orden.

functions Implementa la clase general de las funciones, de las que heredan los morfismos y las funciones y relaciones de primer orden.

fofunctions Implementa las funciones y relaciones de primer orden.

morphisms Implementa los morfismos.

constellation Implementa la clase Constellation que representa a la clase de estructuras $\mathcal{K}$, y contiene las implementaciones de los algoritmos vistos en el Capítulo 3.

lindenbaum Implementa los algoritmos de generación de álgebras de Lindenbaum presentados en el Capítulo 4.

examples Incluye algunos ejemplos de estructuras, y de tipos de primer orden utilizados fundamentalmente para testing.

fotheories Incluye varias teorías de primer orden, para generar estructuras automáticamente mediante Mace4.

sage_to Implementa entre conversión de estructuras algebraicas de Sage a nuestra clase FO_Model.

5.3. Instalación

5.3.1. Instalación del paquete en SageMath

Suponiendo que tenemos instalado SageMath en la ruta absoluta SAGEDIR, nos movemos al directorio donde se quiere descargar el paquete Definability, seguimos los siguientes comandos para instalar el paquete en nuestra instalación de SageMath:

```
$ git clone https://github.com/pablogventura/
    definability.git
$ cd tesis/package
$ ./make_spkg.sh
$ mv definability-hash.spkg SAGEDIR
$ cd SAGEDIR
$ ./sage -i definability-hash.spkg
```

5.4. Uso del paquete

Una vez instalado el paquete basta con arrancar Sage con el comando:

```
$ ./sage
```

y una vez en la consola de Sage, importamos el paquete con el comando:

```
import definability
```

5.5. Generación y entrada de estructuras

Al momento de ingresar estructuras para su chequeo puede optar por la generación automática de estructuras a partir de una teoría de primer orden mediante la interfaz a Mace4, o la entrada de una estructura en particular manualmente.

5.5.1. Generación de estructuras de una teoría de primer orden

Para definir un objeto del tipo FO_Theory que implementa a una teoría de primer orden basta con un nombre, una descripción, y una lista de axiomas en la sintaxis propia de Mace4. Por ejemplo para definir la teoría de reticulados bastaría con la siguiente linea:

```
Lat = definability.FO_Theory("Lat",
                             "Lattices",
                             ['(x v y) v z = x v (y v
                                 z)',
                              'x v y = y v x',
                              '(x^y)^z = x^(y^z)',
                              'x^y = y^x',
                              '(x v y)^x = x',
                              '(x^y) v x = x'])
```

Ahora para recorrer los reticulados (filtrando isomorfismos) basta con llamar al método find_models con la cardinalidad buscada, el cual devuelve un generador. Por ejemplo para recorrer los reticulados de 5 elementos basta con

```
list(Lat.find_models(5))
```

que devolverá una lista con todos los reticulados de 5 elementos filtrando isomorfismos.

Nuestro paquete incluye una gran variedad de teorías de primer orden ya cargadas en el modulo fotheories. Por ejemplo: Lat para reticulados, Graph para grafos, Grp para grupos, etc. La lista completa se puede revisar mediante listar el modulo fotheories de nuestro paquete con el siguiente comando:

```
dir(definability.fotheories)
```

5.5.2. Entrada manual de una estructura

Supongamos que queremos ingresar manualmente el reticulado $\mathbf{2} \times \mathbf{2}$. Lo primero es definir el tipo de primer orden al que pertenece. Para crear un objeto de la clase FO_Type basta con pasar dos diccionarios, el primero para las funciones y el segundo para las relaciones. Los diccionarios contienen las aridades y se indexan por los símbolos correspondientes. Por ejemplo, para definir el tipo de los reticulados, junto con una relación unaria P para chequear:

```
tlat = definability.FO_Type({'v': 2, '^': 2},{"P":1})
```

Luego se debe definir un universo para la estructura, como por ejemplo

```
universe = [0,1,2,3]
```

Luego se necesitan definir las funciones supremo e ínfimo, y una relación unaria P para su chequeo.

Una función, implementada por la clase FO_Operation se puede definir mediante un diccionario con claves de tuplas que apuntan al valor de la función, o en el caso de funciones binarias, una matriz A donde el valor $A_{i,j}$ se corresponde a la función evaluada en (i, j). Para definir una relación utilizando la clase

FO_Relation basta con una lista de las tuplas que pertenecen a la relación y el universo sobre el que la relación está definida.

Como ejemplo, definimos el ínfimo y el supremo utilizando cada una de las maneras para definir funciones, y definimos la relación P:

```
meet = definability.FO_Operation({(0, 0): 0,
                                  (0, 1): 0,
                                  (0, 2): 0,
                                  (0, 3): 0,
                                  (1, 0): 0,
                                  (1, 1): 1,
                                  (1, 2): 2,
                                  (1, 3): 3,
                                  (2, 0): 0,
                                  (2, 1): 2,
                                  (2, 2): 2,
                                  (2, 3): 0,
                                  (3, 0): 0,
                                  (3, 1): 3,
                                  (3, 2): 0,
                                  (3, 3): 3})
join = definability.FO_Operation([[0,1,2,3],
                                  [1,1,1,1],
                                  [2,1,2,1],
                                  [3,1,1,3]])
rP = definability.FO_Relation([(2,), (3,)], uni)
```

Finalmente, para definir el reticulado basta con hacer:

```
l2x2 = definability.FO_Model(tlat, uni, {'^': meet,
                                          'v': join},
                                         {'P': rP})
```

5.6. Chequeo de definibilidad

Una vez definidas las estructuras, para chequear definibilidad se utiliza un objeto de la clase Constellation, pasando una lista de las estructuras en $\mathcal{K}$. Esta clase implementa los algoritmos vistos en el Capítulo 3. Por ejemplo siguiendo con el ejemplo anterior:

```
c = definability.Constellation([l2x2])
```

Para chequear definibilidad están los siguientes métodos:

- is_existential_definable
- is_existential_positive_definable
- is_open_definable
- is_positive_open_definable

los cuales toman dos tipos $\mathcal{L} \subseteq \mathcal{L}'$ para decidir definibilidad de las relaciones del tipo $\mathcal{L}' - \mathcal{L}$ en $\mathcal{L}$, y devuelven una tupla con un booleano y un contraejemplo, si es que lo hay.

Por ejemplo para chequear definibilidad abierta de la relación P:

```
c.is_open_definable(tlat,tlat + definability.FO_Type
    ({},{"P":1}))
```

Lo cual en particular es falso, por lo que el paquete devuelve la tupla:

```
(False, Embedding([0] -> 2,
                  FO_Type({'v': 2, '^': 2},{}),
                  antitype= ['P'],
                  Injective, ))
```

Donde devuelve un contraejemplo, ya que el embedding del subreticulado formado por el elemento 0 en el subreticulado formado por el elemento 2 (un isomorfismo entre subestructuras de $\mathbf{2} \times \mathbf{2}$) no preserva P, negando el Teorema 14.

5.7. Generación de álgebras de Lindenbaum

Para la generación de las álgebras de Lindenbaum desarrolladas en el capítulo 4 el paquete dispone del modulo lindenbaum.

Este modulo contiene las siguientes funciones:

ji_of_existencial_definable_algebra para generar los elementos join-irreducibles de $\mathbf{E}_n(\mathcal{K})$

ji_of_existencial_positive_definable_algebra para generar los elementos join-irreducibles de $\mathbf{E}_n^+(\mathcal{K})$

atoms_of_open_definable_algebra para generar los átomos de $\mathbf{Op}_n(\mathcal{K})$

ji_of_open_positive_definable_algebra para generar los elementos join-irreducibles de $\mathbf{Op}_n^+(\mathcal{K})$

Estas funciones toman un objeto del tipo Constellation que determina $\mathcal{K}$, un tipo para los morfismos en cuestión y una aridad.

Por ejemplo para obtener los elementos join-irreducibles del reticulado de relaciones ternarias definibles en $\mathbf{2} \times \mathbf{2}$ bastaría con la siguiente linea:

```
definability.lindenbaum.
    ji_of_existencial_positive_definable_algebra(c,
    definability.examples.tiporet,3)
```

Lo cual devuelve una lista de 22 relaciones join-irreducibles, que acortamos como ejemplo:

```
[Relation([0, 0, 0],
          [1, 1, 1],
          [2, 2, 2],
          [3, 3, 3]),
.
```

```
.
.
Relation([0, 0, 0],
         [0, 1, 1],
         [0, 2, 2],
         [0, 3, 3],
         [1, 0, 0],
         [1, 1, 1],
         [1, 2, 2],
         [1, 3, 3],
         [2, 0, 0],
         [2, 1, 1],
         [2, 2, 2],
         [2, 3, 3],
         [3, 0, 0],
         [3, 1, 1],
         [3, 2, 2],
         [3, 3, 3])]
```

Capítulo 6

Conclusiones y trabajo futuro

6.1. Conclusiones

En este trabajo se hicieron los siguientes aportes:

- Se desarrollaron algoritmos para decidir definibilidad de relaciones en primer orden, en formulas abiertas, abiertas positivas, existenciales y existenciales positivas.

- Se desarrollaron los respectivos algoritmos para obtener el álgebra de relaciones definibles en cada fragmento de primer orden mencionado anteriormente.

- Se desarrolló una herramienta que implementa dichos algoritmos.

- Al probar la herramienta, se conjeturo y luego se probó una caracterización de las relaciones binarias definibles por existenciales positivas en reticulados distributivos (Teorema 43).

Dado lo inicial de la investigación, las conclusiones tienen que ver con la dirección futura de la investigación y las preguntas interesantes que han surgido durante este trabajo, que se encuentran en la siguiente sección.

Por otra parte, resulta muy alentador que el primer ejemplo visto seriamente haya permitido conjeturar el Teorema 43. Esperamos que sea una herramienta muy útil como asistente de investigación para el grupo de Semántica Algebraica.

6.2. Trabajo futuro

La experiencia durante el desarrollo nos ha hecho notar que la una gran parte de de los cálculos vienen por el calculo de morfismos. En relación a la incidencia que puedan tener propiedades de $\mathcal{K}$ en la complejidad de decidir definibilidad es interesante destacar el extenso trabajo existente acerca de la complejidad computacional del CSP (ver, e.g., [Creignou, Kolaitis y Vollmer 2008]). Es de esperar que si el problema de calcular morfismos para la clase $\mathcal{K}$ (y estructuras derivadas) está bien condicionado, esto tendrá un efecto positivo sobre el costo computacional de nuestro problema. Un artículo que merece mención especial es [Idziak y col. 2010], en donde se demuestra que el cálculo de morfismos se puede resolver en tiempo polinomial si se cuenta con la existencia de términos con ciertas propiedades ecuacionales (*cube terms*). Una situación particular donde se cuenta con cube terms es bajo la presencia de un *Near-Unanimy term*, como por ejemplo en el caso de estructuras que tienen un reducto de reticulado. Debido a esto resulta interesante el desarrollo de algoritmos que decidan definibilidad para este tipo de estructuras. Recientemente se ha demostrado que la presencia de cube terms es decidible [Horowitz 2015], aunque, dada la naturaleza teórica de este resultado, no está claro si el algoritmo descubierto tiene la eficiencia suficiente para que resulte relevante a nuestros propósitos.

Queda pendiente explorar y aprovechar en la implementación, la interacción entre los diferentes tipos de definibilidad. Queda muy claro que hay una jerarquía entre ellas que podría ser aprovechada, al calcular diferentes tipos de definibilidad sobre la misma clase $\mathcal{K}$.

También esperamos explorar mejores implementaciones para la generación de subestructuras, investigando a fondo los algoritmos utilizados por el Universal Álgebra Calculator.

En cuanto a nuestra implementación, pensamos que se podría mejorar portando algunas de las partes más críticas a C.

Bibliografía

Campercholi, Miguel y Diego Vaggione (2015). *Semantical conditions for the definability of functions and relations*. eprint: 1506.07501. URL: http://www.arxiv.org/abs/1506.07501.

Creignou, Nadia, Phokion Kolaitis y Heribert Vollmer (2008). *Complexity of Constraints: An Overview of Current Research Themes*. Springer.

Gent, Ian, Chris Jefferson y Ian Miguel (2006). “MINION: A Fast, Scalable, Constraint Solver”. En: *Proceedings of the 2006 Conference on ECAI 2006: 17th European Conference on Artificial Intelligence August 29 – September 1, 2006, Riva Del Garda, Italy*. IOS Press, págs. 98-102.

Horowitz, Jonah (2015). “Testing for edge terms is decidable”. En: *Algebra universalis* 73.3-4, págs. 321-334.

Idziak, PaweŁ y col. (2010). “Tractability and Learnability Arising from Algebras with Few Subpowers”. En: *SIAM Journal on Computing* 39.7, págs. 3023-3037.

Russell, Stuart y Peter Norvig (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall.