Universidad Nacional Autónoma de México

Facultad de Ciencias

La equivalencia entre lógica y autómatas

T E S I S

QUE PARA OBTENER EL TÍTULO DE:
LIC. EN CIENCIAS DE LA COMPUTACIÓN
PRESENTA:

PABLO ENRIQUE ZENIL RIVAS

DIRECTOR DE TESIS:
FAVIO EZEQUIEL MIRANDA PEREA

Ciudad Universitaria, Cd. Mx, 2016

1. Datos del alumno.
    Zenil
    Rivas
    Pablo Enrique
    5539591652
    Universidad Nacional Autónoma de México
    Facultad de Ciencias
    Ciencias de la Computación
    097178383
2. Datos del tutor.
    Dr.
    Favio Ezequiel
    Miranda
    Perea
3. Datos del sinodal 1
    Dr
    David
    Meza
    Alcántara
4. Datos del sinodal 2
    Dr
    Carlos
    Torres
    Alcaraz
5. Datos del sinodal 3
    M. en C.
    Araceli Liliana
    Reyes
    Cabello
6. Datos del sinodal 4
    Dra.
    Lourdes del Carmen
    Gonzáles
    Huesca
3. Datos del trabajo escrito
    La equivalencia entre lógica y autómatas
    110 p.
    2016

Índice general

Introducción

La teoría de autómatas y la lógica matemática permiten estudiar los conceptos relacionados a los lenguajes formales y las máquinas de estados, ilustrando las profundas conexiones que estos temas tienen con el proceso de razonamiento formal y demostrando propiedades de los sistemas computacionales. El razonamiento formal sobre los sistemas computacionales cobra más importancia debido al uso de estos en sistemas críticos de los que pueden llegar a depender la seguridad y la vida de los usuarios. El *software* y *hardware* usados en este tipo de aplicaciones se ha vuelto increíblemente complejo. La teoría de autómatas y la lógica juegan un papel importante en el modelado y la verificación de dichos sistemas mediante el empleo de herramientas de verificación de modelos (*model checking*) para demostrar formalmente que operan de una forma correcta. Por lo anterior, resulta importante estudiar las distintas relaciones formales entre la lógica y los autómatas.

El objetivo de este trabajo es presentar detalladamente la relación de equivalencia existente entre ciertas clases de autómatas y la lógica monádica de segundo orden. Esta equivalencia es con respecto a la expresividad, es decir, cualquier lenguaje reconocible mediante un autómata es definible en la lógica monádica de segundo orden y viceversa. Más aún, las transformaciones entre ambos formalismos son algorítmicas.

Nos vamos a centrar en el desarrollo de los teoremas de Büchi que establecen la equivalencia entre las clases de los lenguajes S1S-definibles, siendo S1S la signatura de un lenguaje lógico monádico de segundo orden, y las clases de lenguajes regulares y ω-regulares. Estos últimos de cadenas infinitas, útiles en aplicaciones como el modelado de sistemas operativos y sistemas de control de tráfico aéreo.

Para lograr nuestro objetivo, debemos recurrir a la caracterización de los lenguajes regulares y ω-regulares mediante autómatas finitos y autómatas de Büchi, respectivamente, y así poder hacer uso de las propiedades de cerradura

bajo intersección, unión y complemento de estas dos clases de autómatas. Esta última propiedad no es trivial en el caso de los autómatas de Büchi por lo que la discutimos detalladamente.

Una de las consecuencias inmediatas de los teoremas de Büchi es la decidibilidad de la lógica de segundo orden S1S. Como una aplicación interesante de este resultado, mostramos, mediante una codificación adecuada, la decidibilidad de la aritmética de Presburger.

Capítulo 1

Preliminares

Un lenguaje se define como un conjunto de palabras o cadenas, construidas a partir de símbolos de un alfabeto dado, en esta definición se incluyen tanto los lenguajes naturales como los lenguajes formales, estos últimos son los de nuestro interés. Aunque suponemos que el lector está familiarizado con la teoría de lenguajes formales, damos aquí algunos pormenores para fijar la notación.

Definición 1.1. *Un alfabeto es un conjunto finito no vacío de símbolos, denotado por* Σ.

Definición 1.2. *Una palabra o cadena* w *sobre* Σ *es una sucesión finita de símbolos* $w = a_0a_1...a_n$ *donde cada* $a_i \in \Sigma$.

Usaremos $u, v, w...$ para denotar cadenas y $a, b, c...$ para denotar símbolos del alfabeto, la cadena vacía se denotará por ε. También podemos representar w como una función $w : \{0, ..., n\} \to \Sigma$, donde $w(i)$ denota el símbolo que está en la i-ésima posición, es decir, $w(i) = a_i$.

Definición 1.3. *La longitud de una cadena* w*, la cual denotaremos por* $|w|$*, es el número de símbolos de* w.

Definición 1.4. *La concatenación de dos cadenas* u *y* v*, denotada por* uv *o por* $u \cdot v$ *es la cadena que se obtiene de pegar los símbolos de* u *seguidos de los símbolos de* v*. Es decir, si* $u = a_0a_1...a_n$ *y* $v = b_0b_1...b_m$*, entonces* $u \cdot v = a_0a_1...a_nb_0b_1...b_m$.

Cabe resaltar que la concatenación es una operación no conmutativa entre cadenas.

Definición 1.5. *Σ^* es el conjunto de todas las palabras finitas sobre el alfabeto Σ.*

Definición 1.6. *Un lenguaje L no es otra cosa que un conjunto de cadenas $L \subseteq \Sigma^*$.*

Ahora centraremos nuestra atención en los lenguajes regulares.

1.1. Lenguajes regulares y autómatas finitos

Definición 1.7. *Un lenguaje L es regular si se obtiene a partir de los lenguajes básicos $\emptyset$ o $\{a\}$, con $a \in \Sigma$, mediante las siguientes operaciones:*

- *Unión. Dados dos lenguajes L_1 y L_2, se define su unión como el lenguaje:*
$$L_1 \cup L_2 = \{w \mid w \in L_1 \text{ o } w \in L_2\}$$
- *Concatenación. Dados dos lenguajes L_1 y L_2, se define su concatenación como el lenguaje:*
$$L_1 \cdot L_2 = \{uv \mid u \in L_1 \text{ y } v \in L_2\}$$
- *Cerradura de Kleene. La cerradura de Kleene de un lenguaje L sobre el alfabeto Σ es el conjunto de todas las cadenas que se obtienen al concatenar cualquier número finito de cadenas de L:*
$$L^* = \{w_1 w_2 ... w_n \mid w_i \in L,\ 1 \leq i \leq n\}$$

En particular la cadena vacía ε pertenece a L^*.

A continuación recordaremos el concepto de autómata finito y su relación con los lenguajes regulares.

Un *autómata finito* es un formalismo matemático que provee un modelo de cómputo mediante el cual se pueden reconocer lenguajes regulares, éste consiste de entradas y salidas discretas definidas por un conjunto finito de estados y transiciones.

Definición 1.8. *Un autómata finito (AF) $\mathcal{A}$ es una estructura algebraica de la forma $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$, donde:*

- $Q \neq \emptyset$ *es el conjunto finito de estados.*
- Σ *es un alfabeto de entrada.*
- $q_0 \in Q$ *es el estado inicial.*
- $F \subseteq Q$ *es el conjunto de estados finales o estados de aceptación.*
- $\Delta \subseteq Q \times \Sigma \times Q$ *es una relación llamada la relación de transición.*

La definición usual de un autómata finito utiliza, en vez de la relación Δ, una función total de transición δ. Sin embargo, elegimos la definición anterior por adecuarse más a nuestros intereses.

Definición 1.9. *Sea $\mathcal{A}$ un autómata finito. Decimos que $\mathcal{A}$ es determinista si Δ es una función total $\delta : Q \times \Sigma \to Q$ a la que llamamos función de transición y lo denotamos como AFD. De lo contrario, decimos que es no determinista y lo denotamos con AFN.*

Definición 1.10. *Sean $\mathcal{A}$ un AFD, $w \in \Sigma^*$ una palabra finita y $a \in \Sigma$. Extendemos la función de transición δ para cadenas, con la función $\delta^* : Q \times \Sigma^* \to Q$, de la siguiente manera:*

- $\delta^*(q, \varepsilon) = q$
- $\delta^*(q, wa) = \delta(\delta^*(q, w), a)$.

En adelante y cuando no haya confusión, utilizaremos δ para referirnos a la función de transición extendida de un AFD.

Definición 1.11. *Sean $\mathcal{A}$ un AF y $w \in \Sigma^*$. Una secuencia o ejecución ρ_w de $\mathcal{A}$ para w, es una secuencia finita de estados $\rho_w = \rho_0 ... \rho_n$, tal que si $0 \leq i < n$, entonces $\rho_i \in Q$, $\rho_0 = q_0$ y $(\rho_i, w(i), \rho_{i+1}) \in \Delta$.*

Aquí cabe destacar que en el caso de ser $\mathcal{A}$ un AFD, entonces $\rho_{i+1} = \delta(\rho_i, w(i))$.

Definición 1.12. *Una ejecución $\rho_w = \rho_0 ... \rho_n$ de $\mathcal{A}$ para $w \in \Sigma^*$, es una secuencia o ejecución de aceptación si $\rho_n \in F$. Decimos que w es aceptada por $\mathcal{A}$ si existe una ejecución de aceptación de $\mathcal{A}$ para w.*

Definición 1.13. *Al conjunto de palabras en Σ^* aceptadas por $\mathcal{A}$ lo llamamos el lenguaje aceptado por $\mathcal{A}$ y lo denotamos con $L(\mathcal{A})$.*

Es importante notar que $L(\mathcal{A}) \subseteq \Sigma^*$.

El siguiente resultado es de gran importancia.

Lema 1.1. *Dado un AF $\mathcal{A}$ podemos determinar si $L(\mathcal{A}) \neq \emptyset$.*

Demostración. Ya que el conjunto de estados de $\mathcal{A}$ es finito, la relación de transición también lo es. Basta con determinar si en la relación de transición, existe una secuencia de estados en la que un estado final es alcanzable desde un estado inicial.

□

Uno de los principales resultados en la teoría de autómatas es el que establece la equivalencia entre los AFD y los AFN y que es enunciado en el siguiente teorema.

Teorema 1.1. *Sea $L \subseteq \Sigma^*$. L es reconocido por un AFD si y sólo si L es reconocido por un AFN. Es decir, existe un AFD $\mathcal{A}$ tal que $L = L(\mathcal{A})$ si y sólo si existe un AFN $\mathcal{B}$ tal que $L = L(\mathcal{B})$.*

Demostración. Es claro que todo lenguaje aceptado por un AFD también es aceptado por un AFN. En la otra dirección, el algoritmo de construcción de subconjuntos sirve de demostración y se puede consultar en cualquier texto de introducción a la teoría de autómatas. Por ejemplo, [13].

□

Otra forma de interpretar el teorema anterior es que la clase de los lenguajes aceptados por los AFD es la misma que la clase de los lenguajes aceptados por los AFN. Con lo que, por su definición, podemos ver a los AFD como un caso particular de los AFN. Lo que nos lleva al siguiente resultado.

Corolario 1.1. *Sea $\mathcal{A}$ un AFN. Se puede construir, a partir de $\mathcal{A}$, un AFD $\mathcal{B}$ tal que $L(\mathcal{A}) = L(\mathcal{B})$.*

Ya establecida la equivalencia de los distintos autómatas finitos, podemos darle nombre a la clase de lenguajes que aceptan. Uno de los principales resultados en la teoría de los lenguajes regulares es el teorema de Kleene, el cual muestra la equivalencia entre los lenguajes regulares y los lenguajes aceptados por los autómatas finitos.

Teorema 1.2 (Kleene)**.** *Un lenguaje $L \subseteq \Sigma^*$ es regular si y sólo si existe un AF $\mathcal{A}$ tal que $L = L(\mathcal{A})$. Es decir, L es aceptado por algún AF $\mathcal{A}$.*

Demostración. La demostración se puede consultar en [13].

□

El estudio de los lenguajes regulares es un área muy importante en la teoría de lenguajes formales. Ésta relaciona la lógica, combinatoria, y álgebra a la teoría de autómatas; y es ampliamente aplicada en todas las ramas de las ciencias de la computación.

A continuación presentamos un resultado fundamental de la teoría de lenguajes regulares que nos servirá para futuras demostraciones.

1.2. El teorema de Myhill-Nerode

El lema de bombeo para lenguajes regulares describe una propiedad esencial de todos ellos. De manera informal, nos dice que toda palabra suficientemente larga, en un lenguaje regular, se puede *bombear*. Esto es, tomar una sección media de la palabra y repetirla un número finito arbitrario de veces para producir una nueva palabra que también está en el lenguaje.

Es importante aclarar que mientras el lema de bombeo establece que todos los lenguajes regulares satisfacen esta condición, la inversa de esta proposición no es verdadera, es decir, un lenguaje que satisface esta condición puede no ser regular. En otras palabras, el lema de bombeo provee una condición necesaria pero no suficiente para que un lenguaje sea regular.

Por otro lado, el teorema de Myhill-Nerode también describe una propiedad esencial de los lenguajes regulares. Nos dice que si el número de clases de una relación de equivalencia, determinada por un lenguaje, es finito, entonces ese lenguaje es regular. A diferencia del lema de bombeo, la condición que proveé el teorema de Myhill-Nerode, para que un lenguaje sea regular, es una condición necesaria y suficiente.

Ambos resultados pueden ser utilizados para probar que un lenguaje en particular no es regular. Una prueba por contradicción, usando el lema de bombeo, consiste en exhibir una palabra que carezca de la propiedad que establece dicho lema y que esté en el lenguaje. Utilizando el teorema de Myhill-Nerode, la prueba consiste en ver que el índice de la partición, inducida por la relación de equivalencia, es infinito.

Parte fundamental de este trabajo se basa en la caracterización de los lenguajes regulares y ω-regulares a través de clases de equivalencia sobre Σ^*, con lo que el teorema de Myhill–Nerode cobra mucha importancia. Damos, en esta sección, las bases sobre las que recae dicho teorema.

Definición 1.14. *Una relación de equivalencia $\equiv$ es de índice finito si el número de clases de equivalencia inducidas por $\equiv$ es finito.*

Como un lenguaje formal no es más que un subconjunto de cadenas de Σ^*, podemos pensar en describir propiedades de las palabras a través de alguna relación de equivalencia $\equiv$ definida sobre Σ^*.

Definición 1.15. *Decimos que una relación de equivalencia $\equiv$ sobre Σ^* es:*

- *Invariante por la derecha (respecto a la concatenación), si*

$$\forall zxy \in \Sigma^*, x \equiv y \Rightarrow xz \equiv yz$$

- *Invariante por la izquierda (respecto a la concatenación), si*

$$\forall zxy \in \Sigma^*, x \equiv y \Rightarrow zx \equiv zy$$

- *Congruencia, si es invariante por la derecha e invariante por la izquierda.*

Las propiedades anteriores son aplicables a cualquier relación de equivalencia definida sobre Σ^*. Lo siguiente es atender a algunas relaciones en concreto.

Definición 1.16. *Sea $L \subseteq \Sigma^*$ un lenguaje. Definimos la relación $\equiv_L$ sobre Σ^* como sigue:*

$$\forall x, y \in \Sigma^*,\ x \equiv_L y \text{ si y sólo si } \forall z \in \Sigma^*, xz \in L \Leftrightarrow yz \in L$$

Lema 1.2. *La relación $\equiv_L$ es una relación de equivalencia invariante por la derecha sobre Σ^*.*

Demostración. Es claro que, por definición, $\equiv_L$ es invariante por la derecha y es una relación de equivalencia porque su definición está dada en términos de una equivalencia lógica.

□

Ya que las cadenas también pueden ser aceptadas o rechazadas por un AF, podemos pensar en una relación sobre Σ^* determinada por algún AFD $\mathcal{A}$.

Definición 1.17. *Sea $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ un AFD. Definimos la relación $\equiv_{\mathcal{A}}$ sobre Σ^*, como sigue:*

$$\forall x, y \in \Sigma^*,\ x \equiv_{\mathcal{A}} y \text{ si y sólo si } \delta(q_0, x) = \delta(q_0, y)$$

Lema 1.3. *La relación $\equiv_{\mathcal{A}}$ es una congruencia de índice finito sobre Σ^*.*

Demostración. Es claro que $\equiv_{\mathcal{A}}$ es una relación de equivalencia, ya que está definida directamente en términos de la igualdad y una función, y es de índice finito porque cada clase de equivalencia está determinada por cada $q \in Q$ y Q es finito. Por lo tanto, $\equiv_{\mathcal{A}}$ es de índice no mayor a $|Q|$.

Sean $u, v, x \in \Sigma^*, q \in Q$ tales que $u \equiv_{\mathcal{A}} v$. Por lo que tenemos: $\delta(q, u) = \delta(q, v)$ con lo que $\delta(q, ux) = \delta(\delta(q, u), x) = \delta(\delta(q, v), x) = \delta(q, vx)$ y esto es si y sólo si $ux \equiv_{\mathcal{A}} vx$ y por lo tanto $\equiv_{\mathcal{A}}$ es invariante por la derecha.

Por otro lado $\delta(q, xu) = \delta(\delta(q, x), u) = \delta(\delta(q, x), v) = \delta(q, xv)$ y esto es si y sólo si $xu \equiv_{\mathcal{A}} xv$ y por lo tanto $\equiv_{\mathcal{A}}$ es invariante por la izquierda.

Entonces $\equiv_{\mathcal{A}}$ es de índice finito y es compatible con la concatenación en Σ^*, es decir, $\equiv_{\mathcal{A}}$ es una congruencia de índice finito sobre Σ^*.

□

Con lo anterior tenemos todas la bases para enunciar y demostrar el teorema que le da nombre a esta sección.

Teorema 1.3 (Myhill-Nerode)**.** *Las siguientes condiciones, sobre un lenguaje $L \subseteq \Sigma^*$, son equivalentes.*

1. *L es regular.*
2. *L es la unión de algunas clases de equivalencia inducidas por una relación de equivalencia de índice finito e invariante por la derecha sobre Σ^*.*
3. *La relación de equivalencia invariante por la derecha $\equiv_L$ tiene índice finito.*

Demostración. Bastará con demostrar las siguientes implicaciones $(1 \to 2)$, $(2 \to 3)$ y $(3 \to 1)$.

- $(1 \to 2)$. Sea L regular, entonces hay un AFD $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ tal que $L = L(\mathcal{A})$. Por el lema (1.3), sabemos que $\equiv_{\mathcal{A}}$ es invariante por la derecha y de índice finito.

Sea $L_q = \{x \in \Sigma^* \mid \delta(q_0, x) = q\}$, con lo que $L = \bigcup_{q \in F} L_q$.

Sean $w \in L$ y $q_f \in F$ tal que $\delta(q_0, w) = q_f$.

Es claro que $L_{q_f} = [w]_{\equiv_{\mathcal{A}}}$, por lo que L es la unión de algunas clases de equivalencia de $\equiv_{\mathcal{A}}$.

- $(2 \to 3)$. Sea L la unión de algunas clases de una relación de equivalencia $\equiv$ de índice finito e invariante por la derecha sobre Σ^*.

  Sean $w_0, ..., w_n \in L$ tal que $L = \bigcup_{0 \leq i \leq n} [w_i]_{\equiv}$.

  Sean $x, y \in \Sigma^*$ tal que $x \equiv y$. Como $\equiv$ es invariante por la derecha, $\forall z \in \Sigma^*$, $xz \equiv yz$.

  Por lo tanto, $\forall z \in \Sigma^*$, $xz \in L$ si y sólo si $xz \in [w_i]_{\equiv}$. Con lo que tenemos:

  $$\begin{array}{lll} xz \in L & \text{si y sólo si} & xz \in [w_i]_{\equiv} \\ & \text{si y sólo si} & xz \equiv w_i \\ & \text{si y sólo si} & yz \equiv w_i \qquad \text{hipótesis} \\ & \text{si y sólo si} & yz \in [w_i]_{\equiv} \\ & \text{si y sólo si} & yz \in L \end{array}$$

  Con lo cual, tenemos que $\forall z \in \Sigma^*$, $xz \in L$ si y sólo si $yz \in L$ y por lo tanto $x \equiv_L y$.

  Por lo anterior, tenemos que si $x \equiv y$, entonces $x \equiv_L y$, es decir, si $y \in [x]_{\equiv}$, entonces $y \in [x]_{\equiv_L}$. Esto es, $\equiv$ es un refinamiento de $\equiv_L$ y como $\equiv$ es de índice finito, entonces $\equiv_L$, es de índice finito.

- $(3 \to 1)$. Sea L tal que $\equiv_L$ es de índice finito sobre Σ^*. Sabemos que L es la unión de algunas clases de equivalencia de $\equiv_L$.

  Para esto, construiremos un AFD $\mathcal{A}$ tal que $L = L(\mathcal{A})$.

  Ya que $\equiv_L$ es de índice finito, podemos tomar $[x_0], [x_1], [x_2], ..., [x_n]$ clases de $\equiv_L$ tales que:

  $$\Sigma^* = \bigcup_{0 \leq i \leq n} [x_i]$$

  con $x_i \in \Sigma^*$ y $[x_0] = [\varepsilon]$.

  Construimos $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ de la siguiente manera:

- $Q = \{[\varepsilon], [x_1], ..., [x_n]\}$.
- $q_0 = [\varepsilon]$.
- Para toda $x \in \Sigma^*$ y toda $a \in \Sigma$, $\delta([x], a) = [xa]$.
- $F = \{[x] \mid x \in L\}$.

Primero tenemos que verificar que δ está bien definida, esto es, que hay exactamente una clase de equivalencia $[xa]$ para cada clase de equivalencia $[x]$ y cada $a \in \Sigma$.

Por el lema (1.2) sabemos que $\equiv_L$ es invariante por la derecha, por lo que, para cualesquiera $x, y \in \Sigma^*$ y $a \in \Sigma$, se tiene que si $x \equiv_L y$, entonces $xa \equiv_L ya$ y esto pasa si y sólo si $[xa] = [ya]$ con lo que se tiene que $\delta([x], a) = \delta([y], a)$ y por lo tanto δ está bien definida.

Veamos que está definición se extiende correctamente para δ^*. Esto es, para cualesquiera $x, y \in \Sigma^*$ se tiene que $\delta^*([x], y) = [xy]$.

Procedemos por inducción sobre y:

$\delta^*([x], \varepsilon) = [x] = [x\varepsilon]$.

Sea $a \in \Sigma$, suponemos que $\delta^*([x], y) = [xy]$ y demostramos que $\delta^*([x], ya) = [xya]$.

$$\begin{array}{lll} \delta^*([x], ya) & = \delta(\delta^*([x], y), a) & \text{definición de } \delta^*. \\ & = \delta([xy], a) & \text{hipótesis de inducción.} \\ & = [xya] & \text{definición de } \delta. \end{array}$$

Por lo tanto para cualquier $x \in \Sigma^*$ se tiene que $x \in L(\mathcal{A})$ si y sólo si $\delta^*([\varepsilon], x) \in F$ si y sólo si $[x] \in F$, y por la definición de F, $[x] \in F$ si y sólo si $x \in L$.

Con lo que podemos concluir que $L(\mathcal{A}) = L$ y por tanto L es regular.

□

Es importante recordar que el teorema de Myhill-Nerode provee una condición necesaria y suficiente para que un lenguaje sea regular. Con lo que el siguiente resultado es casi inmediato.

Lema 1.4. *Sea $\equiv$ una congruencia sobre Σ^* de índice finito. Cada clase de equivalencia de $\equiv$ es regular.*

Demostración. Como $\equiv$ es una congruencia, en particular, es invariante por la derecha. Además, $\forall w \in \Sigma^*$ se tiene que $[w] = [w] \cup [w]$, por lo que $[w]$ es la unión de algunas clases de equivalencia de una relación de equivalencia de índice finito e invariante por la derecha. Entonces, por el teorema de Myhill-Nerode (1.3) y ya que w es arbitraria, cada clase de equivalencia de $\equiv$, es regular.

□

Con esto hemos sentado las base teóricas de lenguajes regulares y autómatas finitos.

1.3. Lógica de primer orden

Una parte fundamental de este trabajo recae en el lenguaje de la lógica formal. No es posible hablar de un solo lenguaje para la lógica de predicados. Dependiendo de la estructura semántica que tengamos en mente, será necesario agregar símbolos particulares para denotar objetos y relaciones entre ellos.

1.3.1. Sintaxis

Empezamos definiendo la parte común a todos los lenguajes de primer orden determinada por los símbolos lógicos y auxiliares, y después el tipo de semejanza o signatura del lenguaje que nos interesa.

Definición 1.18. *La parte común a todos los lenguajes lógicos de primer orden consta de:*

- *Un conjunto infinito numerable de símbolos de variable de primer orden* $V_1 = \{x_0, x_1, ...\}$.
- *La constante lógica:* $\top$
- *Los conectivos lógicos:* $\neg, \vee$
- *El cuantificador existencial:* $\exists$
- *El símbolo de igualdad:* $=$
- *Los símbolos auxiliares:* $(,)$ *y* $,$

Observemos que los operadores primitivos son: $\neg, \vee, \exists$. También utilizaremos los símbolos $\rightarrow, \wedge, \leftrightarrow, \bot, \forall$, pensando en sus definiciones usuales.

Definición 1.19. *La signatura de un lenguaje en particular está dada por:*

- *Un conjunto, posiblemente vacío, de símbolos o letras de predicado* $\mathcal{P} = \{P_0, P_1, ...\}$

  A cada símbolo P_i *se le asigna un número natural* (m)*, llamado el índice o aridad, el cual indica el número de argumentos que recibe dicho símbolo, por ejemplo,* $P_0^{(m)}$*.*

- *Un conjunto, posiblemente vacío, de símbolos de función* $\mathcal{F} = \{f_0, f_1, ...\}$

  De la misma manera, a cada símbolo de función se le asigna un número natural (m)*, llamado el índice o aridad, el cual indica el número de argumentos que recibe dicho símbolo.*

- *Un conjunto, posiblemente vacío, de símbolos de constante* $\mathcal{C} = \{c_0, c_1, ...\}$

Dado que un lenguaje $\mathcal{L}$ de primer orden queda determinado de manera única por su signatura podemos denotarlo como una estructura algebraica de la forma $\mathcal{L} = \langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$

Definición 1.20. *Los términos del lenguaje de primer orden son aquellas expresiones que representarán objetos bajo alguna interpretación. El conjunto* $T_{\mathcal{L}}$ *de términos de* $\mathcal{L}$ *se define como:*

- *Si* $c \in \mathcal{C}$*, entonces* $c \in T_{\mathcal{L}}$*.*
- *Si* $x \in V_1$*, entonces* $x \in T_{\mathcal{L}}$*.*
- *Si* $f^{(n)} \in \mathcal{F}$ *y* $t_1, ..., t_n \in T_{\mathcal{L}}$*, entonces* $f(t_1, ..., t_n) \in T_{\mathcal{L}}$*.*

Definición 1.21. *El conjunto* $A_{\mathcal{L}}$ *de fórmulas atómicas de* $\mathcal{L}$ *se define como:*

- $\top \in A_{\mathcal{L}}$*.*
- *Si* $P^{(n)} \in \mathcal{P}$ *y* $t_1, ..., t_n \in T_{\mathcal{L}}$*, entonces* $P(t_1, ..., t_n) \in A_{\mathcal{L}}$*.*
- *Si* $t_1, t_2 \in T_{\mathcal{L}}$*, entonces* $t_1 = t_2 \in A_{\mathcal{L}}$*.*

Definición 1.22. *El conjunto* $\Phi_{\mathcal{L}}$ *de fórmulas de* $\mathcal{L}$ *se define como:*

- *Si* $\varphi \in A_{\mathcal{L}}$, *entonces* $\varphi \in \Phi_{\mathcal{L}}$.
- *Si* $\varphi \in \Phi_{\mathcal{L}}$, *entonces* $\neg\varphi \in \Phi_{\mathcal{L}}$
- *Si* $\varphi, \psi \in \Phi_{\mathcal{L}}$, *entonces* $\varphi \vee \psi \in \Phi_{\mathcal{L}}$
- *Si* $\varphi \in \Phi_{\mathcal{L}}$, $x \in V_1$, *entonces* $\exists x\varphi \in \Phi_{\mathcal{L}}$

Como es usual extendemos $\Phi_{\mathcal{L}}$ definiendo fórmulas con los conectivos lógicos $\wedge, \rightarrow, \leftrightarrow$ y el cuantificador universal $\forall$, de la siguiente manera:

$$\begin{aligned} \bot &\equiv_{def} \neg\top \\ \varphi \wedge \psi &\equiv_{def} \neg(\neg\varphi \vee \neg\psi) \\ \varphi \rightarrow \psi &\equiv_{def} \neg\varphi \vee \psi \\ \varphi \leftrightarrow \psi &\equiv_{def} (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi) \\ \forall x\varphi &\equiv_{def} \neg\exists x\neg\varphi \end{aligned}$$

donde $x \in V_1$ y $\psi, \varphi \in \Phi_{\mathcal{L}}$.

En la lógica de predicados los cuantificadores $\forall x$ y $\exists x$ son ejemplos del fenómeno de ligado que también surge en la mayoría de lenguajes de programación. La idea general es que ciertas presencias de variables son presencias *ligadas* o simplemente *ligas*, cada una de las cuales se asocia con una expresión llamada su *alcance*. A continuación damos las definiciones para dichos conceptos.

Definición 1.23. *Dada una cuantificación* $\forall x\varphi$ *o* $\exists x\varphi$, *la presencia de* x *en* $\forall x$ *o* $\exists x$ *es la variable que liga el cuantificador correspondiente, mientras que la fórmula* φ *se llama el alcance, ámbito o radio de acción del cuantificador.*

Definición 1.24. *Una presencia de la variable* x *en la fórmula* φ *está ligada o acotada si es la variable que liga a un cuantificador de* φ *o si figura en el alcance de un cuantificador* $\forall x$ *o* $\exists x$ *de* φ. *Si una presencia de la variable* x *en la fórmula* φ *no está ligada, decimos que está libre en* φ.

Obsérvese que una misma variable x puede figurar tanto libre como ligada en una fórmula.

Definición 1.25. *Sea* φ *una fórmula. El conjunto de variables libres de* φ, *se denota* $FV(\varphi)$. *Es decir,* $FV(\varphi) = \{x \in V \mid x \text{ está libre en } \varphi\}$. *La notación* $\varphi(x_1, ..., x_n)$ *quiere decir que* $\{x_1, ..., x_n\} \subseteq FV(\varphi)$.

Definición 1.26. *Una fórmula* φ *es cerrada si no tiene variables libres, es decir, si* $FV(\varphi) = \emptyset$. *Una fórmula cerrada también se conoce como enunciado o sentencia.*

1.3.2. Semántica

Ya una vez definida la sintaxis, debemos darle un significado de manera formal.

Definición 1.27. *Sea $\mathcal{L} = \langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$ un lenguaje formal de primer orden. Una estructura o interpretación para $\mathcal{L}$ es un par $\mathcal{M} = \langle M, \mathcal{I} \rangle$, donde $M \neq \emptyset$ es un conjunto llamado el universo de la estructura e $\mathcal{I}$ es una función con dominio $\mathcal{L}$, tal que:*

- *Si $P^{(n)} \in \mathcal{P}$ entonces $\mathcal{I}(P) \subseteq M^n$*
- *Si $f^{(n)} \in \mathcal{F}$ entonces $\mathcal{I}(f) : M^n \to M$*
- *Si $c \in \mathcal{C}$ entonces $\mathcal{I}(c) \in M$*

Si $\mathcal{M} = \langle M, \mathcal{I} \rangle$ es una interpretación para $\mathcal{L}$, también decimos que $\mathcal{M}$ es una $\mathcal{L}$-interpretación o $\mathcal{L}$-estructura.

Dada una interpretación $\mathcal{M} = \langle M, \mathcal{I} \rangle$ simplificamos la notación con las siguientes definiciones:

$$P^{\mathcal{I}} := \mathcal{I}(P)$$
$$f^{\mathcal{I}} := \mathcal{I}(f)$$
$$c^{\mathcal{I}} := \mathcal{I}(c)$$

Las siguientes definiciones dependen todas de una $\mathcal{L}$-interpretación arbitraria pero fija $\mathcal{M} = \langle M, \mathcal{I} \rangle$.

Definición 1.28. *Un estado, asignación o valuación de las variables es una función $\sigma : V_1 \to M$. Tal que si $x \in V_1$, $\sigma(x) \in M$.*

Definición 1.29. *Sean $y \in V_1$ una variable y σ un estado de las variables. Dadas las variables $x_1, ..., x_n \in V_1$ y los elementos $m_1, ..., m_n \in M$ definimos el estado modificado o actualizado en $\vec{x} = (x_1, ..., x_n)$ por $\vec{m} = (m_1, ..., m_n)$, denotado por $\sigma[x_1, ..., x_n/m_1, ..., m_n]$ o $\sigma[\vec{x}/\vec{m}]$, como sigue:*

$$\sigma[\vec{x}/\vec{m}](y) = \begin{cases} \sigma(y) & si\ y \notin \{x_1, ..., x_n\} \\ m_i & si\ y = x_i, 1 \leq i \leq n \end{cases}$$

Obsérvese que $\sigma[\vec{x}/\vec{m}]$ es un estado que difiere de σ únicamente en los valores de $\vec{x}$ y que la expresión $\sigma[\vec{x}/\vec{m}]$ es sólo el nombre del estado, la aplicación a una variable es $\sigma[\vec{x}/\vec{m}](y)$.

Definición 1.30 (**Interpretación de términos**). *Sea σ un estado de las variables. Definimos la función de interpretación de términos con respecto a σ, $\mathcal{I}_\sigma : T_\mathcal{L} \to M$, como sigue:*

$$\begin{aligned} \mathcal{I}_\sigma(x) &= \sigma(x) \\ \mathcal{I}_\sigma(c) &= c^\mathcal{I} \\ \mathcal{I}_\sigma(f(t_1, ..., t_n)) &= f^\mathcal{I}(\mathcal{I}_\sigma(t_1), ..., \mathcal{I}_\sigma(t_n)) \end{aligned}$$

con $x \in V_1$, $c \in \mathcal{C}$, $f \in \mathcal{F}$ y $t_1, ..., t_n \in T_\mathcal{L}$.

Lema 1.5. (De coincidencia para términos). *Sean $t \in T$ y σ_1, σ_2 dos estados de las variables tales que $\sigma_1(x) = \sigma_2(x)$ para toda variable x que figura en t. Entonces $\mathcal{I}_{\sigma_1}(t) = \mathcal{I}_{\sigma_2}(t)$.*

Demostración. Inducción sobre t.

□

Ya definido el proceso para interpretar términos podemos definir la interpretación de las fórmulas.

Definición 1.31 (**Interpretación de fórmulas**). *Sea σ un estado de las variables. Definimos la función de interpretación de fórmulas con respecto a σ, $\mathcal{I}_\sigma : \Phi_\mathcal{L} \to \{0, 1\}$, como sigue:*

$$\begin{aligned} \mathcal{I}_\sigma(\bot) &= 0 \\ \mathcal{I}_\sigma(\top) &= 1 \\ \mathcal{I}_\sigma(t_0 = t_1) &= 1 \quad \textit{si y sólo si } \mathcal{I}_\sigma(t_0) = \mathcal{I}_\sigma(t_1) \\ \mathcal{I}_\sigma(P(t_1, ..., t_n)) &= 1 \quad \textit{si y sólo si } (\mathcal{I}_\sigma(t_1), ..., \mathcal{I}_\sigma(t_n)) \in P^\mathcal{I} \\ \mathcal{I}_\sigma(\neg\varphi) &= 1 \quad \textit{si y sólo si } \mathcal{I}_\sigma(\varphi) = 0 \\ \mathcal{I}_\sigma(\psi \wedge \varphi) &= 1 \quad \textit{si y sólo si } \mathcal{I}_\sigma(\psi) = \mathcal{I}_\sigma(\varphi) = 1 \\ \mathcal{I}_\sigma(\psi \vee \varphi) &= 0 \quad \textit{si y sólo si } \mathcal{I}_\sigma(\psi) = \mathcal{I}_\sigma(\varphi) = 0 \\ \mathcal{I}_\sigma(\psi \to \varphi) &= 0 \quad \textit{si y sólo si } \mathcal{I}_\sigma(\psi) = 0 \textit{ o } \mathcal{I}_\sigma(\varphi) = 1 \\ \mathcal{I}_\sigma(\psi \leftrightarrow \varphi) &= 0 \quad \textit{si y sólo si } \mathcal{I}_\sigma(\psi) = \mathcal{I}_\sigma(\varphi) \\ \mathcal{I}_\sigma(\forall x\varphi) &= 1 \quad \textit{si y sólo si } \mathcal{I}_{\sigma[x/m]}(\varphi) = 1 \textit{ para todo } m \in M \\ \mathcal{I}_\sigma(\exists x\varphi) &= 1 \quad \textit{si y sólo si } \mathcal{I}_{\sigma[x/m]}(\varphi) = 1 \textit{ para algún } m \in M \end{aligned}$$

con $t_i \in T_\mathcal{L}$, $P \in \mathcal{P}$, $x \in V_1$ y $\varphi, \psi \in \Phi_\mathcal{L}$.

Obsérvese que todas las definiciones anteriores dependen del lenguaje $\mathcal{L}$ y de la $\mathcal{L}$-interpretación particular $\mathcal{M}$.

Análogamente al caso de términos, tenemos el lema de coincidencia.

Lema 1.6. (De coincidencia para fórmulas). *Sean $\varphi \in \Phi$ y σ_1, σ_2 dos estados de las variables tales que $\sigma_1(x) = \sigma_2(x)$ para toda variable $x \in FV(\varphi)$. Entonces:*

$$\mathcal{I}_{\sigma_1}(\varphi) = \mathcal{I}_{\sigma_2}(\varphi)$$

Demostración. Inducción sobre φ.

□

Definición 1.32 (Satisfacibilidad, verdad y falsedad). *Sean $\varphi \in \Phi_{\mathcal{L}}$, $\Gamma \subseteq \Phi_{\mathcal{L}}$ y $\mathcal{M} = \langle M, \mathcal{I} \rangle$ una interpretación. Entonces:*

- *φ es satisfacible en $\mathcal{M}$ si existe un estado de las variables σ tal que $\mathcal{I}_\sigma(\varphi) = 1$ y lo denotamos con $\mathcal{M} \models \varphi[\sigma]$, o con $\mathcal{M} \models_\sigma \varphi$.*
- *φ es verdadera en $\mathcal{M}$ si para todo estado de las variables σ se tiene que $\mathcal{I}_\sigma(\varphi) = 1$ y lo denotamos con $\mathcal{M} \models \varphi$. En tal caso decimos que $\mathcal{M}$ es un modelo de φ.*
- *φ es falsa en $\mathcal{M}$ ($\mathcal{I}_\sigma(\varphi) = 0$) si y sólo si $\mathcal{M} \models \neg\varphi$.*
- *Γ es consistente o satisfacible en $\mathcal{M}$ si existe un estado de las variables σ tal que $\mathcal{M} \models_\sigma \phi$ para toda $\phi \in \Gamma$.*

Cabe destacar que el conjunto Γ puede ser infinito.

En la lógica proposicional las nociones de ser falsa y no ser verdadera coinciden. Si una fórmula φ es falsa en una interpretación $\mathcal{I}$, entonces $\mathcal{I}(\varphi) = 0$, lo cual sucede si y sólo si $\mathcal{I}(\varphi) \neq 1$, es decir, si y sólo si φ no es verdadera en $\mathcal{I}$. Por otro lado, en la lógica de predicados, tal equivalencia se pierde, puesto que si φ no es verdadera (es decir, si $M \not\models \varphi$), entonces, de acuerdo a la definición de verdad, existe un estado σ tal que $M \not\models_\sigma \varphi$, es decir $\mathcal{I}_\sigma(\varphi) = 0$ o bien φ es insatisfacible en el estado σ. Por lo tanto, una fórmula no verdadera es aquella tal que es insatisfacible en algún estado de sus variables, o bien tal que su negación es satisfacible en algún estado de sus variables. Sin embargo, para poder afirmar que φ es falsa, por definición, tendríamos que mostrar que $M \models \neg\varphi$, es decir, que $\neg\varphi$ es satisfacible en todos los estados posibles. Por lo tanto, la noción de falsedad es más fuerte que la noción de no ser verdadera. En resumen, si una fórmula φ no es verdadera, no tenemos derecho a concluir que φ es falsa.

Con todo lo anterior, sólo queda destacar una propiedad importante de los enunciados en la lógica de primer orden y es que al no tener variables libres, cada enunciado se comporta de la misma manera que las fórmulas en la lógica proposicional, tal como lo asegura la siguiente proposición.

Proposición 1.1. *Sean φ un enunciado y $\mathcal{M} = \langle M, \mathcal{I} \rangle$ una interpretación. Entonces, se cumple una y sólo una de las siguientes condiciones:*

1. *$\mathcal{M} \models \varphi$, es decir, φ es verdadero en $\mathcal{M}$.*
2. *$\mathcal{M} \models \neg\varphi$, es decir, φ es falso en $\mathcal{M}$.*

Demostración. Esto es consecuencia del lema de coincidencia para fórmulas (1.6).

(1). Sea σ un estado cualquiera. Si $\mathcal{I}_\sigma(\varphi) = 1$, entonces, como φ no tiene variables libres, cualquier otro estado σ' coincide con σ en $FV(\varphi) = \emptyset$. Por lo tanto, por el lema de coincidencia, $\mathcal{I}_{\sigma'}(\varphi) = 1$ y así se cumple que $\mathcal{I}_\sigma(\varphi) = 1$ para cualquier estado σ.

(2). Por otra parte, si $\mathcal{I}_\sigma(\varphi) = 0$, entonces, $\mathcal{I}_\sigma(\neg\varphi) = 1$ y se procede de la misma forma para obtener que $\neg\varphi$ es verdadero, es decir, φ es falso.

□

El siguiente corolario es inmediato.

Corolario 1.2. *Si φ es un enunciado entonces φ es falso si y sólo si φ no es verdadero en $\mathcal{M}$.*

Hemos definido la sintaxis y la semántica que todo lenguaje lógico de primer orden comparte, lo siguiente es definir la signatura de un lenguaje en particular con el que podamos trabajar.

1.4. La lógica S0S

Queremos caracterizar a los lenguajes regulares a través de la lógica, para eso debemos definir la signatura del lenguaje lógico de manera que nos permita describir con precisión la propiedad de que un conjunto de palabras en un alfabeto Σ sea reconocible por un AF. Para esto, definiremos la signatura de una lógica de primer orden sobre palabras a la que llamaremos S0S.

Los objetos sobre los cuales vamos a predicar son las posiciones de los símbolos en una palabra dada sobre un alfabeto Σ. Con esto en mente, podemos pensar en una relación sucesor $S(x, y)$ que nos diga que la posición y es el sucesor de la posición x, es decir, $y = x + 1$; y para cada símbolo a en Σ, un predicado $Q_a(x)$ que exprese que en la posición x se encuentra el símbolo a.

Definición 1.33. *Sea Σ un alfabeto finito. La sintaxis del lenguaje S0S extiende la sintaxis de la parte común a todo lenguaje lógico de primer orden mediante la siguiente signatura:*

- *El conjunto de símbolos de predicado: $\mathcal{P} = \{S^{(2)}\} \cup \{Q_a^{(1)} \mid a \in \Sigma\}$*
- *El conjunto de símbolos de función: $\mathcal{F} = \emptyset$*
- *El conjunto de símbolos de constante: $\mathcal{C} = \emptyset$*

Con esto podemos definir la interpretación que dotará de significado a lo que podamos enunciar en S0S. Ya que un lenguaje formal es un conjunto de palabras sobre algún alfabeto Σ, y la signatura nos permite predicar sobre posiciones de palabras, cada palabra determinará una interpretación para S0S de la siguiente manera:

Definición 1.34. *Sea Σ un alfabeto y sea $w = b_0...b_{n-1} \in \Sigma^*$. El modelo $\hat{w} = \langle \mathbf{n}, \mathcal{I} \rangle$ se conoce como el modelo de palabra para w y éste es una interpretación para S0S, donde $\mathbf{n} = \{0, ..., n-1\}$ es el conjunto de posiciones de los símbolos de w e $\mathcal{I}$ es la función de interpretación definida como:*

- *$S^{\mathcal{I}} = \{(i, i+1) \mid 0 \leq i < n-1\}$ es la relación sucesor en $\mathbf{n}$.*
- *$Q_a^{\mathcal{I}} = \{i \in \mathbf{n} \mid w(i) = a\}$ son predicados unarios que determinan el conjunto de posiciones de símbolos de w que son iguales al símbolo a.*

Observemos que con la definición anterior y para un estado de variables σ, tenemos que:

- $\mathcal{I}_\sigma(S(x, y)) = 1$ si y sólo si $\sigma(y) = \sigma(x) + 1$.
- $\mathcal{I}_\sigma(Q_a(x)) = 1$ si y sólo si $b_{\sigma(x)} = a$.

Definición 1.35. *Sea $w \in \Sigma^*$. Decimos que w satisface al enunciado φ en S0S si $\hat{w} \models \varphi$.*

Usando la definición anterior, ya podemos hablar de palabras que satisfacen o son descritas por un enunciado en S0S y por tanto podemos hablar de conjuntos de esas palabras.

Definición 1.36. *El lenguaje definido por el enunciado φ en S0S es:*

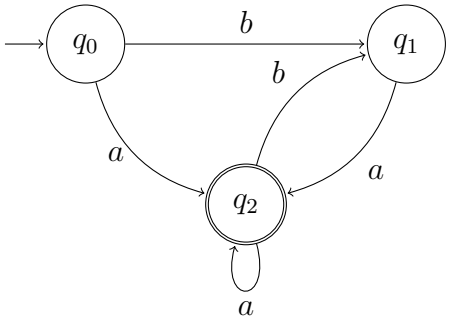
$$L(\varphi) = \{w \in \Sigma^* \mid \hat{w} \models \varphi\}$$

Es decir, $L(\varphi)$ es el conjunto de todas las palabras que satisfacen a φ. Este concepto genera la siguiente noción de definibilidad.

Definición 1.37. *Decimos que un lenguaje $L \subseteq \Sigma^*$ es S0S-definible si existe un enunciado ψ en S0S tal que $L = L(\psi)$.*

Con esto tenemos otra manera de determinar un lenguaje formal sobre Σ^*. Recordemos que sólo estamos interesados en los lenguajes regulares, por lo que queremos establecer un vínculo entre éstos y los lenguajes definibles.

Veamos un ejemplo simple que explica la descripción de los lenguajes regulares por fórmulas lógicas de S0S. El siguiente autómata finito $\mathcal{A}$:



acepta todas las palabras sobre el alfabeto $\Sigma = \{a, b\}$ donde:

1. Toda b es seguida de una a.
2. El último símbolo es a.

Estas condiciones pueden ser expresadas con los siguientes enunciados (respectivamente):

$$\varphi_1 =_{def} \forall x(Q_b(x) \to \exists y(S(x, y) \wedge Q_a(y)))$$
$$\varphi_2 =_{def} \exists v(\neg\exists z S(v, z) \wedge Q_a(v))$$

Notemos que la subfórmula $\neg\exists z S(v,z)$ quiere decir que v es la última posición. Así, el lenguaje aceptado por $\mathcal{A}$ es S0S-definible por el enunciado:

$$\varphi =_{def} \varphi_1 \wedge \varphi_2$$

Ahora podemos verificar que si $w \in L(\mathcal{A})$, entonces $w \in L(\varphi)$.

Recordemos que cada $w \in L(\mathcal{A})$ cumple con las condiciones (1) y (2). Supongamos que $|w| = n$, nos basta ver que el modelo de palabra $\hat{w} = \langle \mathbf{n}, \mathcal{I} \rangle$ satisface φ. Así:

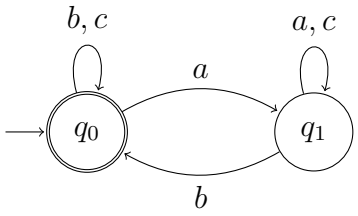
$\hat{w} \models \varphi$	si y sólo si $\mathcal{I}(\varphi) = 1$	
si y sólo si	$\mathcal{I}(\ \forall x(Q_b(x) \to \exists y(S(x,y) \wedge Q_a(y))) \wedge$ $\exists v(\neg\exists z S(v,z) \wedge Q_a(v))\ ) = 1$	
si y sólo si	$\mathcal{I}(\ \forall x(Q_b(x) \to \exists y(S(x,y) \wedge Q_a(y)))\ ) = 1$ y $\mathcal{I}(\ \exists v(\neg\exists z S(v,z) \wedge Q_a(v))\ ) = 1$	
si y sólo si	$\mathcal{I}_{\sigma[x/p]}(\ Q_b(x) \to$ $\exists y(S(x,y) \wedge Q_a(y))\ ) = 1$ y $\mathcal{I}_{\sigma[v/r]}(\ \neg\exists z S(v,z) \wedge Q_a(v)\ ) = 1$	para toda $p \in \mathbf{n}$ y para alguna $r \in \mathbf{n}$
si y sólo si	$\mathcal{I}_{\sigma[x/p]}(\ \neg Q_b(x)\ ) = 1$ o $\mathcal{I}_{\sigma[x/p]}(\ \exists y(S(x,y) \wedge Q_a(y))\ ) = 1$ y $\mathcal{I}_{\sigma[v/r]}(\ \neg\exists z S(v,z)\ ) = 1$ y $\mathcal{I}_{\sigma[v/r]}(\ Q_a(v)\ ) = 1$	para toda $p \in \mathbf{n}$ y para alguna $r \in \mathbf{n}$
si y sólo si	$\mathcal{I}_{\sigma[x/p]}(\ Q_b(x)\ ) = 0$ o $\mathcal{I}_{\sigma[x,y/p,q]}(\ S(x,y)\ ) = 1$ y $\mathcal{I}_{\sigma[x,y/p,q]}(\ Q_a(y)\ ) = 1$ y $\mathcal{I}_{\sigma[v,z/r,s]}(\ S(v,z)\ ) = 0$ y $\mathcal{I}_{\sigma[v/r]}(\ Q_a(v)\ ) = 1$	para toda $p \in \mathbf{n}$, para alguna $q \in \mathbf{n}$, para alguna $r \in \mathbf{n}$ y para toda $s \in \mathbf{n}$
si y sólo si	$w(p) \neq b$ o $q = p+1$ y $w(q) = a$ y $s \neq r+1$ y $w(r) = a$	para toda $p \in \mathbf{n}$, para alguna $q \in \mathbf{n}$, para alguna $r \in \mathbf{n}$ y para toda $s \in \mathbf{n}$

La primera parte, $w(p) \neq b$ o $q = p+1$ y $w(q) = a$, es la definición de la condición (1). Esto es, si $w(p) = b$, entonces $w(p+1) = a$.

En la segunda parte se tiene que para toda s y para alguna r, $s \neq r+1$ y esto se cumple si y sólo si $r = n-1$, es decir, r es la última posición, y además $w(r) = a$. Esto es la definición de la condición (2).

Así, el modelo de cada palabra $w \in L(\mathcal{A})$ -que cumple las condiciones (1) y (2)- satisface φ y por lo tanto $w \in L(\varphi)$.

Consideremos otro ejemplo. El siguiente autómata finito $\mathcal{A}$:



acepta todas las palabras sobre el alfabeto $\Sigma = \{a, b, c\}$ tales que:

1. No tienen el símbolo a o
2. Toda a tiene una b en algún lugar a su derecha.

Trataremos de expresar este lenguaje con un enunciado φ de S0S de la forma $\varphi_1 \vee \varphi_2$.

El punto (1) es sencillo y podemos expresarlo con el enunciado:

$$\varphi_1 =_{def} \forall x(Q_b(x) \vee Q_c(x))$$

Para el punto (2), consideremos la fórmula:

$$\psi_1 =_{def} Q_a(x) \rightarrow \exists y(S(x, y) \wedge Q_b(y))$$

con la que podemos definir una primera aproximación al enunciado φ_2:

$$\varphi_2' =_{def} \forall x(\psi_1)$$

El enunciado φ_2' describe a las palabras sobre Σ en las que todo símbolo a tiene un símbolo b exactamente un lugar a su derecha. Por ejemplo: *ab*, *cccabab* y *bbbabc* son palabras descritas por φ_2' que además están en $L(\mathcal{A})$. Notemos que cualquier palabra de $L(\mathcal{A})$ en la que todo símbolo a tiene un símbolo b dos o más lugares a la derecha, no está en $L(\varphi_2')$. Podemos definir una fórmula ψ_2 que describa a las palabras en las que todo símbolo a tiene un símbolo b exactamente dos lugares a la derecha, de la siguiente manera:

$$\psi_2 =_{def} Q_a(x) \rightarrow \exists y \exists z(S(x, z) \wedge S(z, y) \wedge Q_b(y))$$

con lo que podemos definir una nueva aproximación a φ_2:

$$\varphi_2'' =_{def} \forall x(\psi_1 \vee \psi_2)$$

De esta manera, el enunciado φ_2'' describe a las palabras sobre Σ en las que todo símbolo a tiene un símbolo b uno o dos lugares a la derecha. Por ejemplo: *aab*, *abbab* y *acbabc* son palabras descritas por φ_2'' que además están en $L(\mathcal{A})$. De nuevo, notemos que cualquier palabra de $L(\mathcal{A})$ en la que todo símbolo a tiene un símbolo b tres o más lugares a la derecha, no está en $L(\varphi_2'')$. Siguiendo el mismo razonamiento, podemos definir una fórmula ψ_3 que describa a las palabras en las que todo símbolo a tiene un símbolo b exactamente tres lugares a la derecha, de la siguiente manera:

$$\psi_3 =_{def} Q_a(x) \rightarrow \exists y \exists z \exists v(S(x,z) \wedge S(z,v) \wedge S(v,y) \wedge Q_b(y))$$

con lo que volvemos a definir una nueva aproximación a φ_2:

$$\varphi_2''' =_{def} \forall x(\psi_1 \vee \psi_2 \vee \psi_3)$$

Así, el enunciado φ_2''' describe a las palabras sobre Σ en las que todo símbolo a tiene un símbolo b, uno, dos o tres lugares a la derecha.

Siguiendo este procedimiento podemos construir, para una n dada, un enunciado que describa a las palabras donde todo símbolo a tiene un símbolo b en, a lo más, n lugares a la derecha. Sin embargo, no podemos construir el enunciado φ_2 para un número arbitrario de lugares ya que obtendríamos algo de la forma:

$$\varphi_2 =_{def} \forall x(\psi_1 \vee \psi_2 \vee \psi_3 \vee ...)$$

el cual es una fórmula *infinita* y por lo tanto mal formada, es decir, no pertenece a nuestro lenguaje formal lógico de primer orden.

El problema radica en expresar, mediante la relación sucesor, el enunciado: *toda a tiene una b en algún lugar a su derecha*; que en otras palabras quiere decir: *para toda posición x en la que se encuentra un símbolo a existe una posición z tal que $x < z$ y en la que se encuentra un símbolo b*. La idea es poder recorrer un camino finito de longitud arbitraria desde una posición con un símbolo a hasta llegar a la posición con un símbolo b.

Está demostrado que la noción de un camino finito de longitud arbitraria, también llamada noción de accesibilidad, no puede ser definida en un lenguaje de primer orden que cuente solamente con la relación sucesor y la igualdad [6]. Dicha demostración es una consecuencia de el teorema de compacidad para la lógica de primer orden enunciado a continuación.

Teorema (Teorema de compacidad)**.** *Sea* Γ *un conjunto de fórmulas de un lenguaje* $\mathcal{L}$ *de primer orden. Si todos los subconjuntos finitos de* Γ *son satisfacibles, entonces* Γ *es satisfacible.*

Enunciamos a continuación el teorema que establece que la noción de accesibilidad no puede ser expresada en la lógica de primer orden con la relación sucesor S.

Teorema 1.4. *La accesibilidad no es expresable en la lógica de primer orden. Es decir, no existe una fórmula* $\phi(u, v)$ *con sólo dos variables libres* (u, v) *y el símbolo de predicado* S *tal que sea satisfecha si y sólo si existe un camino finito desde la posición determinada por* u *hasta la posición determinada por* v*.*

Demostración. La demostración se puede consultar en [6].

□

El poder expresivo de la lógica de primer orden S0S restringe al conjunto de lenguajes que se pueden obtener a partir de enunciados en S0S [15]. Por lo tanto, tendremos que recurrir a una lógica con mayor poder expresivo.

Capítulo 2

Lógica monádica de segundo orden

La lógica monádica de segundo orden es la extensión de la lógica de primer orden que permite la cuantificación sobre predicados monádicos (unitarios). Es importante observar que en una lógica monádica también pueden existir predicados de otro índice pero no está permitido cuantificar sobre ellos.

Vamos a definir la lógica monádica a partir de la de primer orden, extendiéndola con una secuencia de variables de conjuntos cuantificables (de segundo orden) y con nuevas fórmulas de la forma: $X(x)$ y $\exists X\varphi$, donde x es una variable de primer orden y X una variable de conjunto de índice 1, llamada *predicado monádico*, que se interpretará sobre subconjuntos de un universo M.

2.1. Sintaxis

Presentamos a continuación la sintaxis de esta nueva lógica.

Definición 2.1. *La lógica monádica de segundo orden extiende a la parte común a todos los lenguajes de primer orden con:*

- *Un conjunto infinito de símbolos de variables de segundo orden* $V_2 = \{X_0, X_1, ...\}$.

Ya que la sintaxis de la lógica de segundo orden extiende la de primer orden y dado que un lenguaje $\mathcal{L}$ de segundo orden también queda determinado

de manera única por su signatura, podemos denotarlo, al igual que en el caso de primer orden, como una estructura algebraica de la forma $\mathcal{L} = \langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$.

Si bien la signatura de un lenguaje no cambia al pasar de primer a segundo orden, el conjunto de términos sí, de acuerdo a la siguiente definición.

Definición 2.2. *El conjunto de términos $\mathcal{L}$ se extiende de la siguiente manera:*

- *Si $X \in V_2$, entonces $X \in T_{\mathcal{L}}$.*

Con estos, podemos podemos extender el conjunto de fórmulas.

Definición 2.3. *El conjunto $A_{\mathcal{L}}$ de fórmulas atómicas de $\mathcal{L}$ se extiende de la siguiente manera:*

- *Si $x \in V_1$, $X \in V_2$, entonces $X(x) \in A_{\mathcal{L}}$.*

Definición 2.4. *El conjunto $\Phi_{\mathcal{L}}$ de fórmulas de $\mathcal{L}$ se extiende de la siguiente manera:*

- *Si $\varphi \in \Phi_{\mathcal{L}}$, $X \in V_2$, entonces $\exists X \varphi \in \Phi_{\mathcal{L}}$.*

Aquí es importante recordar que $\Phi_{\mathcal{L}}$ ya contiene la definición usual de las fórmulas con los conectivos lógicos $\wedge, \rightarrow, \leftrightarrow$, la constante lógica $\bot$ y el cuantificador universal $\forall$ para variables de primer orden. Ahora extendemos $\Phi_{\mathcal{L}}$ con el cuantificador universal para variables de segundo orden, de la manera usual.

$$\forall X \varphi \equiv_{def} \quad \neg \exists X \neg \varphi$$

donde $X \in V_2$ y $\varphi \in \Phi_{\mathcal{L}}$.

Ya definida la sintaxis, ahora le daremos un significado formal.

2.2. Semántica

Sea $\mathcal{L} = \langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$ un lenguaje formal monádico de segundo orden. La semántica de la lógica monádica de segundo orden se obtiene extendiendo a la semántica de la lógica de primer orden, de la siguiente manera:

Definición 2.5. *Un estado, asignación o valuación de las variables es una función* $\sigma : V_1 \cup V_2 \to M \cup \mathcal{P}(M)$. *Tal que:*

- *Si* $x \in V_1$, *entonces* $\sigma(x) \in M$.
- *Si* $X \in V_2$, *entonces* $\sigma(X) \subseteq M$.

Definición 2.6. *Sean* $Y \in V_2$ *una variable y* σ *un estado de las variables. Dadas las variables* $X_1, ..., X_n$ *y los elementos* $K_1, ..., K_n \subseteq M$, *definimos el estado modificado o actualizado para variables de segundo orden en* $\vec{X} = (X_1, ..., X_n)$ *por* $\vec{K} = (K_1, ..., K_n)$, *denotado por* $\sigma[X_1, ..., X_n/K_1, ..., K_n]$ *o* $\sigma[\vec{X}/\vec{K}]$, *como sigue:*

$$\sigma[\vec{X}/\vec{K}](Y) = \begin{cases} \sigma(Y) & si\ Y \notin \{X_1, ..., X_n\} \\ K_i & si\ Y = K_i, 1 \leq i \leq n \end{cases}$$

Observese que análogamente a la lógica de primer orden, $\sigma[\vec{X}/\vec{K}]$ es un estado que difiere de σ únicamente en los valores de $\vec{X}$ y que la expresión $\sigma[\vec{X}/\vec{K}]$ es sólo el nombre del estado.

Definición 2.7 (**Interpretación de términos).** *Sea* σ *un estado de las variables. Extendemos la función de interpretación de términos con respecto a* σ, $\mathcal{I}_\sigma : T_\mathcal{L} \to M \cup \mathcal{P}(M)$, *para un lenguaje monádico de segundo orden, como sigue:*

$$\mathcal{I}_\sigma(X) = \quad \sigma(X)$$

con $X \in V_2$.

Ya definido el proceso para interpretar términos podemos definir la interpretación de las fórmulas.

Definición 2.8 (**Interpretación de fórmulas).** *Sean* σ *un estado de las variables y* $\mathcal{M} = \langle M, \mathcal{I} \rangle$ *una interpretación. Extendemos la función de interpretación de fórmulas con respecto a* σ, $\mathcal{I}_\sigma : \Phi_\mathcal{L} \to \{0, 1\}$, *para un lenguaje monádico de segundo orden, como sigue:*

$$\begin{array}{ll} \mathcal{I}_\sigma(X(x)) = 1 & si\ y\ sólo\ si\ \sigma(x) \in \sigma(X) \\ \mathcal{I}_\sigma(\exists X \varphi) = 1 & si\ y\ sólo\ si\ \mathcal{I}_{\sigma[X/K]}(\varphi) = 1\ para\ algún\ K \subseteq M \end{array}$$

Así, de la misma manera que lo hicimos con S0S, definiremos la signatura del lenguaje lógico de segundo orden que nos permitirá caracterizar a los lenguaje regulares.

2.3. La lógica S1S

La lógica S1S es la extensión monádica de segundo orden de la lógica S0S y se conoce como la lógica monádica de segundo orden con sucesor. Es decir, también contamos con la fórmula $S(x, y)$ para decir que y es sucesor de x y con la familia de fórmulas $Q_a(x)$ para decir que en la posición x se encuentra el símbolo a.

Definición 2.9. *Sea Σ un alfabeto. La sintaxis del lenguaje S1S extiende la sintaxis de la parte común a todo lenguaje lógico monádico de segundo orden y definimos su signatura como:*

- *El conjunto de símbolos de predicado: $\mathcal{P} = \{S^{(2)}\} \cup \{Q_a^{(1)} \mid a \in \Sigma\}$*
- *El conjunto de símbolos de función: $\mathcal{F} = \emptyset$*
- *El conjunto de símbolos de constante: $\mathcal{C} = \emptyset$*

Definiremos una fórmula auxiliar que nos permita expresar la propiedad de un conjunto de tener un solo elemento y que nos será de gran utilidad.

$$Sing(X) \equiv_{def} \exists x(X(x) \wedge \forall y(X(y) \rightarrow x = y))$$

La fórmula $Sing(X)$ expresa que existe un elemento x que pertenece a X y que cualquier otro elemento en X es igual a x. Esto es, X tiene un solo elemento.

De la misma manera que lo hicimos para S0S, definiremos la interpretación que será determinada por una palabra dada sobre algún alfabeto Σ.

Definición 2.10. *Sea Σ un alfabeto y sea $w = a_0...a_{n-1}$ en Σ^*. El modelo $\hat{w} = \langle \mathbf{n}, \mathcal{I} \rangle$ se conoce como el modelo de palabra para w y éste es una interpretación para S1S, donde $\mathbf{n} = \{0, ..., n-1\}$ es el conjunto de posiciones de los símbolos de w e $\mathcal{I}$ es la función de interpretación definida como:*

- *$S^{\mathcal{I}} = \{(i, i+1) \mid 0 \leq i \leq n-1\}$ es la relación sucesor en $\mathbf{n}$.*
- *$Q_a^{\mathcal{I}} = \{i \in \mathbf{n} \mid w(i) = a\}$ son predicados unarios que determinan el conjunto de posiciones de símbolos de w que son iguales al símbolo a.*

Las siguientes definiciones son análogas al caso de la lógica S0S.

Definición 2.11. *Sea $w \in \Sigma^*$. Decimos que w satisface un enunciado φ en S1S si $\hat{w} \models \varphi$.*

Con lo anterior ya podemos definir un lenguaje a través de S1S.

Definición 2.12. *El lenguaje definido por el enunciado φ en S1S es:*

$$L(\varphi) = \{w \in \Sigma^* \mid \hat{w} \models \varphi\}$$

Definición 2.13. *Decimos que un lenguaje $L \subseteq \Sigma^*$ es S1S-definible si existe un enunciado ψ en S1S tal que $L = L(\psi)$.*

Vamos a simplificar las demostraciones que se deriven de la fórmula $Sing(X)$ dando una interpretación más simple de la que se obtiene de su definición.

Notemos que $Sing(X)$ no está definida en términos de alguna fórmula $Q_a(x)$, por lo que, para proporcionar una interpretación que la satisfaga, sólo necesitamos determinar a $\mathbf{n}$. Así, la interpretación $\mathcal{M} = \langle \mathbf{n}, \mathcal{I} \rangle$ nos basta para verificar que $Sing(X)$ sea satisfecha por algún conjunto de un solo elemento.

Lema 2.1. *Sea $\mathcal{M} = \langle \mathbf{n}, \mathcal{I} \rangle$ una interpretación para S1S y sea σ un estado de las variables. Entonces:*

$$\mathcal{I}_\sigma(Sing(X)) = 1 \text{ si y sólo si } \sigma(X) = \{p\} \text{ para algún } p \in \mathbf{n}$$

Demostración. $\mathcal{I}_\sigma(Sing(X)) = 1$

si y sólo si $\mathcal{I}_\sigma(\ \exists x(X(x) \wedge \forall y(X(y) \to x = y))\ ) = 1$

si y sólo si $\mathcal{I}_{\sigma[x/p]}(\ X(x) \wedge \forall y(X(y) \to x = y)\ ) = 1$ para algún $p \in \mathbf{n}$

si y sólo si $\mathcal{I}_{\sigma[x/p]}(\ X(x)\ ) = 1$ y $\mathcal{I}_{\sigma[x/p]}(\ \forall y(X(y) \to x = y)\ ) = 1$ para algún $p \in \mathbf{n}$

si y sólo si $\mathcal{I}_{\sigma[x/p]}(\ X(x)\ ) = 1$ y $\mathcal{I}_{\sigma[x,y/p,r]}(\ X(y) \to x = y\ ) = 1$ para algún $p \in \mathbf{n}$ y para toda $r \in \mathbf{n}$

si y sólo si $\mathcal{I}_{\sigma[x/p]}(\ X(x)\ ) = 1$ y $\mathcal{I}_{\sigma[x,y/p,r]}(\ \neg X(y) \vee x = y\ ) = 1$ para algún $p \in \mathbf{n}$ y para toda $r \in \mathbf{n}$

si y sólo si $\mathcal{I}_{\sigma[x/p]}(\ X(x)\ ) = 1$ y $\mathcal{I}_{\sigma[x,y/p,r]}(\ \neg X(y)\ ) = 1$ o $\mathcal{I}_{\sigma[x,y/p,r]}(\ x = y\ ) = 1$ para algún $p \in \mathbf{n}$ y para toda $r \in \mathbf{n}$

si y sólo si	$\mathcal{I}_{\sigma[x/p]}(\ X(x)\ ) = 1$ y $\mathcal{I}_{\sigma[x,y/p,r]}(\ X(y)\ ) = 0$ o $\mathcal{I}_{\sigma[x,y/p,r]}(\ x = y\ ) = 1$	para algún $p \in \mathbf{n}$ y para toda $r \in \mathbf{n}$
si y sólo si	$p \in \sigma(X)$ y $r \notin \sigma(X)$ o $p = r$	para algún $p \in \mathbf{n}$ y para toda $r \in \mathbf{n}$
si y sólo si	$\sigma(X) = \{p\}$	para algún $p \in \mathbf{n}$

□

A continuación definiremos la signatura de una lógica que nos facilitará la construcción de autómatas finitos que reconocerán a los lenguajes S1S-definibles.

2.4. La lógica S1S0

En esta sección definiremos el lenguaje lógico monádico de segundo orden S1S0 que nos servirá de puente entre los lenguajes S1S-definibles y los lenguajes reconocibles por autómatas finitos. En esta lógica vamos a deshacernos de las variables de primer orden, codificándolas en conjuntos de un sólo elemento para así poder expresar todo en términos de variables de segundo orden. En consecuencia, tendremos una teoría, equivalente a S1S, con un conjunto de fórmulas atómicas reducido que podremos representar más fácilmente con autómatas.

Definición 2.14. *La lógica S1S0 se obtiene a partir de la parte común a todos los lenguajes de segundo orden, eliminando el conjunto de variables de primer orden, es decir, definiendo:*

- *El conjunto de variables de primer orden vacío:* $V_1 = \emptyset$

Con esto, todo lo definido para variables de primer orden (términos, fórmulas, interpretaciones, etc..) queda fuera del lenguaje S1S0.

Definición 2.15. *Sea* Σ *un alfabeto finito. La signatura del lenguaje S1S0 se define como:*

- *El conjunto de símbolos de predicado:* $\mathcal{P} = \{\subseteq^{(2)}, S^{(2)}\}$
- *El conjunto de símbolos de función:* $\mathcal{F} = \emptyset$

- *El conjunto de símbolos de constante:* $\mathcal{C} = \{Q_a \mid a \in \Sigma\}$

Observemos que se ha agregado un símbolo de predicado para la relación de contención de conjuntos y los símbolos Q_a se han convertido en constantes. Más aún, el conjunto de términos para este lenguaje incluye únicamente a las constantes Q_a y a las variables de segundo orden.

Definición 2.16. *El conjunto* A_{S1S0} *de fórmulas atómicas se define como:*

- *Si* $X, Y \in V_2$*, entonces* $X \subseteq Y \in A_{S1S0}$
- *Si* $X, Y \in V_2$*, entonces* $S(X, Y) \in A_{S1S0}$
- *Si* $X \in V_2$ *y* $a \in \Sigma$*, entonces* $X \subseteq Q_a \in A_{S1S0}$

De la misma forma que en S1S, definiremos algunas fórmulas auxiliares que nos serán de utilidad.

$$\begin{array}{rl} X = Y \equiv_{def} & X \subseteq Y \wedge Y \subseteq X \\ NE(X) \equiv_{def} & \exists Z \neg (X \subseteq Z) \\ Sing(X) \equiv_{def} & NE(X) \wedge \forall Y((NE(Y) \wedge Y \subseteq X) \rightarrow X \subseteq Y) \end{array}$$

La fórmula $NE(X)$ expresa que el conjunto X es no vacío, ya que existe un conjunto que no lo contiene.

La fórmula $Sing(X)$ expresa que X es un conjunto con un solo elemento, ya que es no vacío y que todo subconjunto no vacío de X contiene a X.

A continuación definimos los modelos de palabra para S1S0.

Definición 2.17. *Sean* Σ *un alfabeto y* $w = a_0 ... a_{n-1}$ *en* Σ^**. El modelo de palabra* $\breve{w} = \langle \mathbf{n}, \mathcal{I} \rangle$ *para* w *es una interpretación para S1S0, donde* $\mathcal{I}$ *es la función de interpretación definida como:*

- $S^{\mathcal{I}} = \{(\{i\}, \{i+1\}) \mid 0 \leq i < n-1\}$ *es la relación sucesor para subconjuntos de un sólo elemento en* $\mathbf{n}$*.*
- $Q_a^{\mathcal{I}} = \{i \in \mathbf{n} \mid w(i) = a\}$ *son conjuntos de posiciones de símbolos de* w *que son iguales al símbolo* a*.*
- $\subseteq^{\mathcal{I}} = \{(N, K) \mid N, K \subseteq \mathbf{n},\ N \subseteq K\}$ *es la relación usual de contención entre conjuntos.*

Observese que la única diferencia, entre el modelo de palabra $\check{w}$ para S1S0 y el modelo $\hat{w}$ para S1S (2.10), está en la función de interpretación.

Definición 2.18. *Sea $w \in \Sigma^*$. Decimos que w satisface un enunciado φ en S1S0 si $\check{w} \models \varphi$.*

Con lo anterior ya podemos definir un lenguaje a través de S1S0.

Definición 2.19. *El lenguaje definido por el enunciado φ en S1S0 es*

$$L(\varphi) = \{w \in \Sigma^* \mid \check{w} \models \varphi\}$$

Definición 2.20. *Decimos que un lenguaje $L \subseteq \Sigma^*$ es S1S0-definible si existe un enunciado ψ en S1S0 tal que $L = L(\psi)$.*

De la misma manera que en S1S, daremos una interpretación más sencilla de las fórmulas $Sing(X)$, $X = Y$ y $NE(X)$.

Lema 2.2. *Sea $\mathcal{M} = \langle \mathbf{n}, \mathcal{I} \rangle$ una interpretación para S1S0 y sea σ un estado de las variables. Entonces:*

$$\begin{array}{rl} \mathcal{I}_\sigma(X = Y) = 1 & \textit{si y sólo si } \sigma(X) = \sigma(Y) \\ \mathcal{I}_\sigma(NE(X)) = 1 & \textit{si y sólo si } \sigma(X) \neq \emptyset \\ \mathcal{I}_\sigma(Sing(X)) = 1 & \textit{si y sólo si } \sigma(X) = \{m\} \textit{ para algún } m \in \mathbf{n} \end{array}$$

Demostración.

- $X = Y$.

  $\mathcal{I}_\sigma(X = Y) = 1$ si y sólo si $\mathcal{I}_\sigma(X \subseteq Y \wedge Y \subseteq X) = 1$

  si y sólo si $\mathcal{I}_\sigma(X \subseteq Y) = 1$ y $\mathcal{I}_\sigma(Y \subseteq X) = 1$

  si y sólo si $\sigma(X) \subseteq \sigma(Y)$ y $\sigma(Y) \subseteq \sigma(X)$

  Notemos que la definición de la igualdad es la usual para conjuntos.

- $NE(X)$.

$\mathcal{I}_\sigma(NE(X)) = 1$ si y sólo si $\mathcal{I}_\sigma(\exists Z \neg(X \subseteq Z)) = 1$

si y sólo si $\mathcal{I}_{\sigma[Z/R]}(\neg(X \subseteq Z)) = 1$ para algún $R \subseteq \mathbf{n}$

si y sólo si $\mathcal{I}_{\sigma[Z/R]}(X \subseteq Z) = 0$ para algún $R \subseteq \mathbf{n}$

si y sólo si $\sigma(X) \not\subseteq R$ para algún $R \subseteq \mathbf{n}$

Si hay un conjunto R que no contiene a $\sigma(X)$, entonces $\sigma(X) \neq \emptyset$.

- $Sing(X)$.

  $\mathcal{I}_\sigma(Sing(X)) = 1$

  si y sólo si $\mathcal{I}_\sigma(NE(X) \wedge \forall Y((NE(Y) \wedge (Y \subseteq X)) \rightarrow (X \subseteq Y))\ ) = 1$

  si y sólo si $\mathcal{I}_\sigma(NE(X)) = 1$ y $\mathcal{I}_\sigma(\ \forall Y((NE(Y) \wedge (Y \subseteq X)) \rightarrow (X \subseteq Y))\ ) = 1$

  si y sólo si $\mathcal{I}_\sigma(NE(X)) = 1$ y $\mathcal{I}_{\sigma[Y/P]}(\ (NE(Y) \wedge (Y \subseteq X)) \rightarrow (X \subseteq Y)\ ) = 1$ para todo $P \subseteq \mathbf{n}$

  si y sólo si $\mathcal{I}_\sigma(NE(X)) = 1$ y $\mathcal{I}_{\sigma[Y/P]}(\neg NE(Y) \vee \neg(Y \subseteq X)) = 1$ o $\mathcal{I}_{\sigma[Y/P]}(X \subseteq Y) = 1$ para todo $P \subseteq \mathbf{n}$

  si y sólo si $\mathcal{I}_\sigma(NE(X)) = 1$ y $\mathcal{I}_{\sigma[Y/P]}(\neg NE(Y)) = 1$ o $\mathcal{I}_{\sigma[Y/P]}(\neg(Y \subseteq X)) = 1$ o $\mathcal{I}_{\sigma[Y/P]}(X \subseteq Y) = 1$ para todo $P \subseteq \mathbf{n}$

  si y sólo si $\sigma(X) \neq \emptyset$ y $P = \emptyset$ o $P \not\subseteq \sigma(X)$ o $\sigma(X) \subseteq P$ para todo $P \subseteq \mathbf{n}$ Lema (NE) y definición de $\subseteq$

  Por lo tanto, si $\sigma(X) \neq \emptyset$ y para toda $P \subseteq \mathbf{n}$:

  1. Si $P = \emptyset$, $Sing(X)$ es satisfecha.
  2. Si $P \not\subseteq \sigma(X)$, $Sing(X)$ es satisfecha.

3. Si $\sigma(X) \subseteq P$, $Sing(X)$ es satisfecha.

Las condiciones, (1) y (2), (1) y (3) son mutuamente excluyentes, por lo que basta ver qué pasa con $Sing(X)$ cuando no se cumplen alguno de estos pares.

Si (1) y (2) no se cumplen, entonces $P \subseteq \sigma(X)$ y además, $Sing(X)$ es satisfecha si y sólo si se cumple (3) y esto se cumple si y sólo si $\sigma(X)$ tiene exactamente un elemento.

Si (1) y (3) no se cumplen, entonces $\sigma(X) \not\subseteq P$, por lo que $\sigma(X) \neq \emptyset$ y además, $Sing(X)$ es satisfecha si y sólo si se cumple (2) y esto se cumple si y sólo si $\sigma(X)$ tiene exactamente un elemento.

□

2.5. Equivalencia entre S1S y S1S0

Para verificar que todo lenguaje S1S-definible es aceptado por un autómata finito utilizaremos S1S0 como puente. Demostraremos que las lógicas S1S y S1S0 son equivalentes, es decir, tienen el mismo poder expresivo. Después veremos que para cada lenguaje S1S0-defininible podemos construir un atómata finito que lo acepte. Con esto, será fácil verificar que un lenguaje S1S-definible es támbien S1S0-definible y por lo tanto, podemos construir un autómata finito que lo acepte. La razón de utilizar S1S0 es que facilita considerablemente la construcción de dichos autómatas.

Necesitamos establecer una relación entre la interpretaciones para S1S y S1S0 a partir de los estados de las variables. Como en S1S0 no tenemos variables de primer orden, al momento de traducir desde S1S no podemos perder esta información y la debemos de codificar de alguna manera. Para esto, podemos asumir que tenemos una función inyectiva que a cada variable de primer orden en S1S le asigna un símbolo de variable de segundo orden en S1S0 de la siguiente manera:

$$x \longmapsto X_x$$

y a cada símbolo de variable de segundo orden le asigna el mismo símbolo:

$$X \longmapsto X$$

Con esto podemos definir una función de traducción de S1S a S1S0.

Definición 2.21. *La función de traducción* $\bullet : \Phi_{S1S} \to \Phi_{S1S0}$ *mapea fórmulas de S1S en fórmulas de S1S0 como sigue:*

$$\begin{aligned}
(x = y)^\bullet &= Sing(X_x) \wedge Sing(Y_y) \wedge X_x = Y_y \\
X(x)^\bullet &= Sing(X_x) \wedge X_x \subseteq X \\
S(x,y)^\bullet &= Sing(X_x) \wedge Sing(Y_y) \wedge S(X_x, Y_y) \\
Q_a(x)^\bullet &= Sing(X_x) \wedge X_x \subseteq Q_a \\
(\neg\varphi)^\bullet &= \neg\varphi^\bullet \\
(\varphi \vee \psi)^\bullet &= \varphi^\bullet \vee \psi^\bullet \\
(\exists x\varphi)^\bullet &= \exists X_x(Sing(X_x) \wedge \varphi^\bullet) \\
(\exists X\varphi)^\bullet &= \exists X\varphi^\bullet
\end{aligned}$$

Es importante notar que la traducción introduce una variable nueva por cada variable de primer orden que figure en la fórmula original. Además, esta nueva variable está acotada por un cuantificador si y sólo si la variable de primer orden está acotada en la fórmula original.

Nuestro objetivo ahora, es mostrar que esta traducción es fiel, en el sentido de que preserva satifacibilidad.

Sean $\mathcal{M} = \langle \mathbf{n}, \mathcal{I} \rangle$, $\widehat{\mathcal{M}} = \langle \mathbf{n}, \widehat{\mathcal{I}} \rangle$ interpretaciones para S1S y S1S0, respectivamente.

Las siguientes definiciones dependen de $\mathcal{M}$ y $\widehat{\mathcal{M}}$.

Definición 2.22. *Dado un estado de las variables* σ *en* $\mathcal{M}$*, definimos el estado de las variables* $\widehat{\sigma}$ *en* $\widehat{\mathcal{M}}$ *como:*

$$\widehat{\sigma}(X_x) = \{\sigma(x)\}$$
$$\widehat{\sigma}(X) = \sigma(X)$$

Veamos que los estados modificados se corresponden con esta definición.

Lema 2.3. *Sean* σ *un estado de las variables en S1S,* x *una variable de primer orden en S1S y* $p \in \mathbb{N}$*. Entonces:*

$$\widehat{\sigma}[X_x/\{p\}](X_x) = \widehat{\sigma[x/p]}(X_x)$$

Demostración.

$$\begin{array}{lll} \widehat{\sigma[x/p]}(X_x) & = \{\sigma[x/p](x)\} & \text{definición de } \widehat{\sigma} \\ & = \{p\} & \text{definición de estado modificado} \\ & = \widehat{\sigma}[X_x/\{p\}](X_x) & \text{definición de estado modificado} \end{array}$$

□

Ahora podemos verificar la traducción de S1S a S1S0. Primero veremos que $\bullet$ traduce correctamente cada una de las fórmulas atómicas de S1S.

Lema 2.4. *$\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (x = y)^\bullet$ si y sólo si $\mathcal{M} \models_\sigma x = y$.*

Demostración. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (x = y)^\bullet$

si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ (x = y)^\bullet\ ) = 1$

si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(X_x) \wedge Sing(Y_y) \wedge X_x = Y_y\ ) = 1$

si y sólo si $\begin{array}{l} \widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(X_x)\ ) = 1 \text{ y} \\ \widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(Y_y)\ ) = 1 \text{ y} \\ \widehat{\mathcal{I}}_{\widehat{\sigma}}(\ X_x = Y_y\ ) = 1 \end{array}$

si y sólo si $\begin{array}{ll} \widehat{\sigma}(X_x) = \{\sigma(x)\} \text{ y} & \text{Lema (2.2) y} \\ \widehat{\sigma}(Y_y) = \{\sigma(y)\} \text{ y} & \text{definición de } \widehat{\sigma} \\ \widehat{\sigma}(X_x) = \widehat{\sigma}(Y_y) & \end{array}$

si y sólo si $\sigma(x) = \sigma(y)$

si y sólo si $\mathcal{M} \models_\sigma x = y$

□

Lema 2.5. *$\widehat{\mathcal{M}} \models_{\widehat{\sigma}} X(x)^\bullet$ si y sólo si $\mathcal{M} \models_\sigma X(x)$.*

Demostración. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} X(x)^\bullet$

si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ X(x)^\bullet\ ) = 1$

si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(X_x) \wedge X_x \subseteq X\ ) = 1$

si y sólo si $\begin{array}{l} \widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(X_x)\ ) = 1 \text{ y} \\ \widehat{\mathcal{I}}_{\widehat{\sigma}}(\ X_x \subseteq X\ ) = 1 \end{array}$

si y sólo si $\begin{array}{ll} \widehat{\sigma}(X_x) = \{\sigma(x)\} \text{ y} & \text{Lema (2.2) y} \\ \widehat{\sigma}(X_x) \subseteq \widehat{\sigma}(X) & \text{definición de } \widehat{\sigma} \end{array}$

si y sólo si $\begin{array}{l}\widehat{\sigma}(X_x) = \{\sigma(x)\} \text{ y} \\ \widehat{\sigma}(X_x) \subseteq \sigma(X)\end{array}$ definición de $\widehat{\sigma}$

si y sólo si $\sigma(x) \in \sigma(X)$

si y sólo si $\mathcal{M} \models_\sigma X(x)$

□

Lema 2.6. *$\widehat{\mathcal{M}} \models_{\widehat{\sigma}} S(x,y)^\bullet$ si y sólo si $\mathcal{M} \models_\sigma S(x,y)$.*

Demostración. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} S(x,y)^\bullet$

si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ S(x,y)^\bullet\ ) = 1$

si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(X_x) \wedge Sing(Y_y) \wedge S(X_x, Y_y)\ ) = 1$

si y sólo si $\begin{array}{l}\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(X_x)\ ) = 1 \text{ y} \\ \widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(Y_y)\ ) = 1 \text{ y} \\ \widehat{\mathcal{I}}_{\widehat{\sigma}}(\ S(X_x, Y_y)\ ) = 1\end{array}$

si y sólo si $\begin{array}{l}\widehat{\sigma}(X_x) = \{\sigma(x)\} \text{ y} \\ \widehat{\sigma}(Y_y) = \{\sigma(y)\} \text{ y} \\ (\widehat{\sigma}(X_x), \widehat{\sigma}(Y_y)) \in S^{\widehat{\mathcal{I}}}\end{array}$ Lema (2.2) y definición de $\widehat{\sigma}$

si y sólo si $\sigma(y) = \sigma(x) + 1$ definición de $S^{\widehat{\mathcal{I}}}$

si y sólo si $\mathcal{M} \models_\sigma S(x,y)$ definición de $S^{\mathcal{I}}$

□

Lema 2.7. *$\widehat{\mathcal{M}} \models_{\widehat{\sigma}} Q_a(x)^\bullet$ si y sólo si $\mathcal{M} \models_\sigma Q_a(x)$.*

Demostración. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} Q_a(x)^\bullet$

si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Q_a(x)^\bullet\ ) = 1$

si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(X_x) \wedge X_x \subseteq Q_a\ ) = 1$

si y sólo si $\begin{array}{l}\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ Sing(X_x)\ ) = 1 \text{ y} \\ \widehat{\mathcal{I}}_{\widehat{\sigma}}(\ X_x \subseteq Q_a\ ) = 1\end{array}$

si y sólo si $\begin{array}{l}\widehat{\sigma}(X_x) = \{\sigma(x)\} \text{ y} \\ \widehat{\sigma}(X_x) \subseteq Q_a^{\widehat{\mathcal{I}}}\end{array}$ Lema (2.2) y definición de $\widehat{\sigma}$

si y sólo si $\begin{array}{l}\widehat{\sigma}(X_x) = \{\sigma(x)\} \text{ y} \\ \widehat{\sigma}(X_x) \subseteq Q_a^{\mathcal{I}}\end{array}$ $Q_a^{\widehat{\mathcal{I}}} = Q_a^{\mathcal{I}}$

si y sólo si $\sigma(x) \in Q_a^{\mathcal{I}}$

si y sólo si $\mathcal{M} \models_\sigma Q_a(x)$

□

Así, los lemas (2.4), (2.5), (2.6) y (2.7) sirven como caso base para la demostración por inducción sobre cualquier fórmula de S1S.

Teorema 2.1. *Sean $\varphi, \psi \in \Phi_{S1S}$. Se cumple lo siguiente:*

1. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (\neg\varphi)^\bullet$ *si y sólo si* $\mathcal{M} \models_\sigma \neg\varphi$.
2. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (\varphi \vee \psi)^\bullet$ *si y sólo si* $\mathcal{M} \models_\sigma \varphi \vee \psi$.
3. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (\exists x\varphi)^\bullet$ *si y sólo si* $\mathcal{M} \models_\sigma \exists x\varphi$.
4. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (\exists X\varphi)^\bullet$ *si y sólo si* $\mathcal{M} \models_\sigma \exists X\varphi$.

Demostración. Por hipótesis de inducción tenemos que:

- $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} \varphi^\bullet$ si y sólo si $\mathcal{M} \models_\sigma \varphi$, es decir, $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\varphi^\bullet) = \mathcal{I}_\sigma(\varphi)$.
- $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} \psi^\bullet$ si y sólo si $\mathcal{M} \models_\sigma \psi$, es decir, $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\psi^\bullet) = \mathcal{I}_\sigma(\psi)$.

Procedemos con el paso inductivo.

1. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (\neg\varphi)^\bullet$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ (\neg\varphi)^\bullet\ ) = 1$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ \neg\varphi^\bullet\ ) = 1$ definición de $\bullet$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ \varphi^\bullet\ ) = 0$

   si y sólo si $\mathcal{I}_\sigma(\ \varphi\ ) = 0$ H.I.

   si y sólo si $\mathcal{M} \models_\sigma \neg\varphi$

2. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (\varphi \vee \psi)^\bullet$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ (\varphi \vee \psi)^\bullet\ ) = 1$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ \varphi^\bullet \vee \psi^\bullet\ ) = 1$ definición de $\bullet$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ \varphi^\bullet\ ) = 1$ o $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ \psi^\bullet\ ) = 1$

   si y sólo si $\mathcal{I}_\sigma(\ \varphi\ ) = 1$ o $\mathcal{I}_\sigma(\ \psi\ ) = 1$ H.I.

   si y sólo si $\mathcal{M} \models_\sigma \varphi \vee \psi$.

3. Sea $p \in \mathbf{n}$. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (\exists x\varphi)^\bullet$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ (\exists x\varphi)^\bullet\ ) = 1$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ \exists X_x(Sing(X_x) \wedge \varphi^\bullet)\ ) = 1$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}[X_x/\{p\}]}(\ Sing(X_x)\ ) = 1$ y $\widehat{\mathcal{I}}_{\widehat{\sigma}[X_x/\{p\}]}(\ \varphi^\bullet\ ) = 1$

   si y sólo si $\widehat{\sigma}(X_x) = \{p\}$ y $\widehat{\mathcal{I}}_{\widehat{\sigma}[X_x/\{p\}]}(\ \varphi^\bullet\ ) = 1$ para alguna $p \in \mathbf{n}$ Lema (2.2)

   si y sólo si $\sigma(x) = p$ y $\widehat{\mathcal{I}}_{\widehat{\sigma[x/p]}}(\ \varphi^\bullet\ ) = 1$ para alguna $p \in \mathbf{n}$ Lema (2.3) y definición de $\widehat{\sigma}$

   si y sólo si $\mathcal{I}_{\sigma[x/p]}(\ \varphi\ ) = 1$ H.I.

   si y sólo si $\mathcal{M} \models_\sigma \exists x\varphi$

4. $\widehat{\mathcal{M}} \models_{\widehat{\sigma}} (\exists X\varphi)^\bullet$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ (\exists X\varphi)^\bullet\ ) = 1$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}}(\ \exists X\varphi^\bullet\ ) = 1$ definición de $\bullet$

   si y sólo si $\widehat{\mathcal{I}}_{\widehat{\sigma}[X/P]}(\ \varphi^\bullet\ ) = 1$ para algún $P \subseteq \mathbf{n}$

   si y sólo si $\mathcal{I}_{\sigma[X/P]}(\ \varphi\ ) = 1$ para algún $P \subseteq \mathbf{n}$ H.I.

   si y sólo si $\mathcal{M} \models_\sigma \exists X\varphi$

□

El siguiente corolario es inmediato

Corolario 2.1. *Si φ un enunciado en S1S, entonces:*

$$\widehat{\mathcal{M}} \models \varphi^{\bullet} \text{ si y sólo si } \mathcal{M} \models \varphi.$$

Así, cualquier fórmula de S1S tiene una fórmula equivalente en S1S0. Ahora veamos que el recíproco también se cumple.

Definición 2.23. *La función de traducción $* : \Phi_{S1S0} \to \Phi_{S1S}$ mapea fórmulas de S1S0 en fórmulas de S1S como sigue:*

$$\begin{aligned}
S(X,Y)^* &= Sing(X) \wedge Sing(Y) \wedge \exists x \exists y\ (X(x) \wedge Y(y) \wedge S(x,y)) \\
(X \subseteq Y)^* &= \forall x(X(x) \to Y(x)) \\
(X \subseteq Q_a)^* &= \forall x(X(x) \to Q_a(x)) \\
(\neg\varphi)^* &= \neg\varphi^* \\
(\varphi \vee \psi)^* &= \varphi^* \vee \psi^* \\
(\exists X\varphi)^* &= \exists X\varphi^*
\end{aligned}$$

Es importante notar que todas las variables nuevas de primer orden que introduce la traducción están acotadas por algún cuantificador. Además, la traducción, no introduce variables nuevas de segundo orden.

De nuevo, establecemos la relación que hay entre estas lógicas a través de sus modelos y los estados de las variables.

Sean $\mathcal{M} = \langle \mathbf{n}, \mathcal{I} \rangle$, $\widehat{\mathcal{M}} = \langle \mathbf{n}, \widehat{\mathcal{I}} \rangle$ dos interpretaciones para S1S y S1S0, respectivamente.

Las siguientes definiciones dependen de $\mathcal{M}$ y $\widehat{\mathcal{M}}$.

Definición 2.24. *Dado un estado de variables $\check{\sigma}$ en $\widehat{\mathcal{M}}$ definimos el estado de las variables $\widetilde{\sigma}$ en $\mathcal{M}$ como sigue:*

$$\widetilde{\sigma}(X) = \check{\sigma}(X)$$

Análogo al caso de $\bullet$, ahora podemos verificar la tradución de S1S0 a S1S. Primero veremos que $*$ traduce correctamente cada una de las fórmulas atómicas de S1S0.

Lema 2.8. *$\mathcal{M} \models_{\widetilde{\sigma}} S(X,Y)^*$ si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} S(X,Y)$.*

Demostración. $\mathcal{M} \models_{\widetilde{\sigma}} S(X,Y)^*$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ S(X,Y)^*\ ) = 1$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ Sing(X) \wedge Sing(Y) \wedge \exists x \exists y (X(x) \wedge Y(y) \wedge S(x,y))\ ) = 1$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ Sing(X)\ ) = 1$ y $\mathcal{I}_{\widetilde{\sigma}}(\ Sing(Y)\ ) = 1$ y $\mathcal{I}_{\widetilde{\sigma}}(\ \exists x \exists y (X(x) \wedge Y(y) \wedge S(x,y))\ ) = 1$

si y sólo si $\widetilde{\sigma}(X) = \{p\}$ y $\widetilde{\sigma}(Y) = \{m\}$ y $\mathcal{I}_{\widetilde{\sigma}[x,y/p,m]}(\ (X(x) \wedge Y(y)) \wedge S(x,y))\ ) = 1$ p.a. $p \in \mathbf{n}$ y p.a. $m \in \mathbf{n}$ Lema (2.1)

si y sólo si $\widetilde{\sigma}(X) = \{p\}$ y $\widetilde{\sigma}(Y) = \{m\}$ y $p \in \widetilde{\sigma}(X)$ y $m \in \widetilde{\sigma}(Y)$ y $m = p+1$ p.a. $p \in \mathbf{n}$ y p.a. $m \in \mathbf{n}$

si y sólo si $\check{\sigma}(X) = \{p\}$ y $\check{\sigma}(Y) = \{m\}$ y $m = p+1$ p.a. $p \in \mathbf{n}$ y p.a. $m \in \mathbf{n}$ definición de $\check{\sigma}$

si y sólo si $(\check{\sigma}(X), \check{\sigma}(Y)) \in S^{\widehat{\mathcal{I}}}$ definición de $S^{\widehat{\mathcal{I}}}$

si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} S(X,Y)$

□

Lema 2.9. *$\mathcal{M} \models_{\widetilde{\sigma}} (X \subseteq Y)^*$ si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} X \subseteq Y$.*

Demostración. $\mathcal{M} \models_{\widetilde{\sigma}} (X \subseteq Y)^*$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ (X \subseteq Y)^*\ ) = 1$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ \forall x (X(x) \to Y(x))\ ) = 1$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}[x/p]}(\ X(x) \to Y(x)\ ) = 1$ para toda $p \in \mathbf{n}$

si y sólo si $p \notin \widetilde{\sigma}(X)$ o $p \in \widetilde{\sigma}(Y)$ para toda $p \in \mathbf{n}$

si y sólo si $\begin{array}{l} p \notin \check{\sigma}(X) \text{ o} \\ p \in \check{\sigma}(Y) \end{array}$ para toda $p \in \mathbf{n}$ definición de $\widetilde{\sigma}$

si y sólo si $\check{\sigma}(X) \subseteq \check{\sigma}(Y)$

si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} X \subseteq Y$ definición de $\subseteq^{\widehat{\mathcal{I}}}$

□

Lema 2.10. *$\mathcal{M} \models_{\widetilde{\sigma}} (X \subseteq Q_a)^*$ si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} X \subseteq Q_a$.*

Demostración. $\mathcal{M} \models_{\widetilde{\sigma}} (X \subseteq Q_a)^*$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ (X \subseteq Q_a)^*\ ) = 1$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ \forall x(X(x) \to Q_a(x))\ ) = 1$

si y sólo si $\mathcal{I}_{\widetilde{\sigma}[x/p]}(\ X(x) \to Q_a(x)\ ) = 1$ para toda $p \in \mathbf{n}$

si y sólo si $\begin{array}{l} p \notin \widetilde{\sigma}(X) \text{ o} \\ p \in Q_a^{\mathcal{I}} \end{array}$ para toda $p \in \mathbf{n}$

si y sólo si $\begin{array}{l} p \notin \check{\sigma}(X) \text{ o} \\ p \in Q_a^{\widehat{\mathcal{I}}} \end{array}$ para toda $p \in \mathbf{n}$ $\begin{array}{l} Q_a^{\mathcal{I}} = Q_a^{\widehat{\mathcal{I}}} \text{ y} \\ \text{definición de } \widetilde{\sigma} \end{array}$

si y sólo si $\check{\sigma}(X) \subseteq Q_a^{\widehat{\mathcal{I}}}$

si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} X \subseteq Q_a$.

□

Análogo al caso de $\bullet$, los lemas (2.8), (2.9) y (2.10) sirven como caso base para la demostración por inducción sobre cualquier fórmula de S1S0 de la misma forma que lo hicimos en el teorema (2.1).

Teorema 2.2. *Sean $\varphi, \psi \in \Phi_{S1S0}$. Se cumple lo siguiente:*

1. *$\mathcal{M} \models_{\widetilde{\sigma}} (\neg\varphi)^*$ si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} \neg\varphi$.*
2. *$\mathcal{M} \models_{\widetilde{\sigma}} (\varphi \vee \psi)^*$ si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} \varphi \vee \psi$.*
3. *$\mathcal{M} \models_{\widetilde{\sigma}} (\exists X\varphi)^*$ si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} \exists X\varphi$.*

Demostración. Por hipótesis de inducción tenemos que:

- $\mathcal{M} \models_{\widetilde{\sigma}} \varphi^*$ si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} \varphi$, es decir, $\mathcal{I}_{\widetilde{\sigma}}(\varphi^*) = \widehat{\mathcal{I}}_{\check{\sigma}}(\varphi)$.
- $\mathcal{M} \models_{\widetilde{\sigma}} \psi^*$ si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} \psi$, es decir, $\mathcal{I}_{\widetilde{\sigma}}(\psi^*) = \widehat{\mathcal{I}}_{\check{\sigma}}(\psi)$.

Procedemos con el paso inductivo.

1. $\mathcal{M} \models_{\widetilde{\sigma}} (\neg\varphi)^*$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ (\neg\varphi)^*\ ) = 1$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ \neg\varphi^*\ ) = 1$ definición de $*$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ \varphi^*\ ) = 0$

   si y sólo si $\widehat{\mathcal{I}}_{\check{\sigma}}(\ \varphi\ ) = 0$ H.I.

   si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} \neg\varphi$.

2. $\mathcal{M} \models_{\widetilde{\sigma}} (\varphi \vee \psi)^*$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ (\varphi \vee \psi)^*\ ) = 1$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ \varphi^* \vee \psi^*\ ) = 1$ definición de $*$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ \varphi^*\ ) = 1$ o $\mathcal{I}_{\widetilde{\sigma}}(\ \psi^*\ ) = 1$

   si y sólo si $\widehat{\mathcal{I}}_{\check{\sigma}}(\ \varphi\ ) = 1$ o $\widehat{\mathcal{I}}_{\check{\sigma}}(\ \psi\ ) = 1$ H.I.

   si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} \varphi \vee \psi$

3. $\mathcal{M} \models_{\widetilde{\sigma}} (\exists X\varphi)^*$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ (\exists X\varphi)^*\ ) = 1$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}}(\ \exists X\varphi^*\ ) = 1$ definición de $*$

   si y sólo si $\mathcal{I}_{\widetilde{\sigma}[X/P]}(\ \varphi^*\ ) = 1$ para algún $P \subseteq \mathbf{n}$

   si y sólo si $\widehat{\mathcal{I}}_{\check{\sigma}[X/P]}(\ \varphi\ ) = 1$ para algún $P \subseteq \mathbf{n}$ H.I.

   si y sólo si $\widehat{\mathcal{M}} \models_{\check{\sigma}} \exists X\varphi$.

□

El siguiente corolario es inmediato

Corolario 2.2. *Si φ un enunciado en S1S0, entonces:*

$$\mathcal{M} \models \varphi^* \text{ si y sólo si } \widehat{\mathcal{M}} \models \varphi.$$

Así, cualquier fórmula de S1S0 tiene una fórmula equivalente en S1S.

De esta manera queda establecida la equivalencia entre S1S y S1S0, lo que nos lleva a verificar la equivalencia entre los lenguajes definibles por las mismas.

Es importante recordar que dada una palabra $w = a_0...a_{n-1}$ sobre Σ^*, los modelos de palabra $\hat{w}$ y $\check{w}$, para w, son tales que:

- $\hat{w} = \langle \mathbf{n}, \mathcal{I} \rangle$ es una interpretación para S1S.
- $\check{w} = \langle \mathbf{n}, \widehat{\mathcal{I}} \rangle$ es una interpretación para S1S0.

Lema 2.11. *Sea $\varphi \in \Phi_{S1S}$ un enunciado en S1S. Entonces, $L(\varphi) = L(\varphi^\bullet)$*

Demostración.

$w \in L(\varphi)$
si y sólo si $\hat{w} \models \varphi$
si y sólo si $\check{w} \models \varphi^\bullet$ corolario (2.1)
si y sólo si $w \in L(\varphi^\bullet)$

□

Lema 2.12. *Sea $\varphi \in \Phi_{S1S0}$ un enunciado en S1S0. Entonces, $L(\varphi) = L(\varphi^*)$*

Demostración.

$w \in L(\varphi)$
si y sólo si $\check{w} \models \varphi$
si y sólo si $\hat{w} \models \varphi^*$ corolario (2.2)
si y sólo si $w \in L(\varphi^*)$

□

Por lo que el siguiente resultado es inmediato.

Teorema 2.3. *Un lenguaje es S1S-definible si y sólo si es S1S0-definible.*

Demostración. Por los lemas (2.11) y (2.12) cualquier lenguaje que puede expresarse mediante un enunciado de S1S o S1S0, también puede expresarse con su traducción.

□

En este capítulo definimos la lógica monádica de segundo orden y dos signaturas particulares sobre la misma: la lógica S1S y la lógica S1S0. También definimos las funciones de traducción $\bullet$ y $*$ que mapean fórmulas de S1S a fórmulas de S1S0 y fórmulas de S1S0 a fórmulas de S1S, respectivamente. Demostramos que estas dos funciones preservan la satisfacibilidad de las fórmulas y por consiguiente la equivalencia de las lógicas S1S y S1S0, esto es, S1S y S1S0 tienen el mismo poder expresivo y por lo tanto cualquier lenguaje es S1S-definible si y sólo si es S1S0-definible.

Este resultado nos será de gran utilidad para demostrar la equivalencia entre los lenguajes S1S-definibles y los lenguajes regulares y ω-regulares.

Capítulo 3

Equivalencia entre autómatas y lógica

En 1960, Büchi [1] y Elgot [2] demostraron que un lenguaje de palabras finitas es reconocible por un autómata finito si y sólo si puede ser caracterizado por una fórmula de la lógica monádica de segundo orden. Desde entonces, distintos resultados análogos han sido obtenidos, por ejemplo, sobre palabras infinitas, árboles y gráficas.

En este capítulo demostraremos los teoremas que establecen la equivalencia entre las clases de los lenguajes S1S-definibles y las clases de lenguajes regulares y ω-regulares. Para estos últimos definiremos la noción de autómata de Büchi, el tipo de autómata que los acepta, y revisaremos a detalle sus propiedades de cerradura bajo las operaciones usuales de conjuntos.

En la demostración de ambos teoremas, seguiremos el mismo procedimiento. Primero veremos que si un lenguaje es reconocible por un autómata $\mathcal{A}$, entonces es S1S-definible. Para esto, debemos tomar en cuenta las siguientes características de una secuencia de aceptación de $\mathcal{A}$:

φ_1: La secuencia de estados de una ejecución de aceptación de $\mathcal{A}$.

φ_2: El inicio o paso por el estado inicial de dicha secuencia de estados.

φ_3: La relación de transición con respecto a dicha secuencia de estados.

φ_4: La condición de aceptación.

Siendo cada φ_i una fórmula de S1S y codificando la secuencia de aceptación

en un enunciado de la forma:

$$\varphi_{\mathcal{A}} \equiv_{def} \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4$$

Así, podremos tomar el modelo de cada palabra (finita o infinita) del lenguaje y verificar que satisface $\varphi_{\mathcal{A}}$.

Después veremos que si un lenguaje es S1S-definible, entonces podemos construir un autómata que lo acepte. Para esto utilizaremos la equivalencia entre S1S y S1S0. Construiremos, de manera inductiva, utilizando las propiedades de cerradura de los lenguajes regulares y ω-regulares, un autómata por cada fórmula de S1S0.

3.1. Autómatas finitos

Para ir de los autómatas finitos a la lógica S1S veremos que dado un autómata finito $\mathcal{A}$, podemos construir un enunciado φ en S1S tal que $L(\mathcal{A}) = L(\varphi)$.

3.1.1. De autómatas a lógica

En lo que sigue: $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ es un autómata finito con $Q = \{0, ..., k\}$, $q_0 = 0$, $w = a_0, ..., a_{n-1} \in L(\mathcal{A})$ y ρ_w es una ejecución de aceptación de $\mathcal{A}$ para w.

Ya que en S1S sólo podemos predicar acerca de posiciones y de conjuntos de posiciones de palabras, agruparemos las posiciones de w en un conjunto P_i por cada estado i visitado en ρ_w. Esta agrupación nos permite saber, a traveś de cada posición de w, que estado fue visitado en ρ_w. Los siguientes conjuntos definen dicha agrupación.

Definición 3.1. *Los conjuntos $P_0, ..., P_k \subseteq \mathbf{n}$ son tales que cada P_i contiene las posiciones de w donde el estado i es visitado en la ejecución de aceptación ρ_w. Es decir:*

$$P_i = \{j \mid i = \rho_j,\ j \in \mathbf{n}\}$$

Observese que P_i es vacío si no se pasó por el estado i.

Lema 3.1. *Si $i, j \in Q$ y $j \neq i$, entonces P_i, P_j son ajenos.*

Demostración. Demostramos por reducción al absurdo. Supongamos que P_i, P_j no son ajenos y que $j \neq i$, entonces existe p tal que $p \in P_i$ y $p \in P_j$ y por lo tanto $i = \rho_p = j$.

□

Con esto, construiremos un enunciado φ_i que codifique cada una de las condiciones de una secuencia de aceptación de $\mathcal{A}$.

Definición 3.2. *Sean $X_0, ..., X_k$ variables de S1S. Definimos las fórmulas $\varphi_1, \varphi_2, \varphi_3$ y φ_4 de la siguiente manera:*

$$\varphi_1 \equiv_{def} \bigwedge_{i \neq j} \forall x \neg(X_i(x) \wedge X_j(x))$$

$$\varphi_2 \equiv_{def} \forall x(\neg \exists y S(y, x) \rightarrow X_0(x))$$

$$\varphi_3 \equiv_{def} \forall x \forall y(S(x, y) \rightarrow (\bigvee_{(i,a,j) \in \Delta} (X_i(x) \wedge Q_a(x) \wedge X_j(y))))$$

$$\varphi_4 \equiv_{def} \forall x(\neg \exists y S(x, y) \rightarrow (\bigvee_{\substack{j \in F \\ (i,a,j) \in \Delta}} (X_i(x) \wedge Q_a(x))))$$

con $i, j \in Q$.

En adelante utilizaremos $\vec{P}$ para denotar $P_0, ..., P_k$ y $\vec{X}$ para $X_0, ..., X_k$.

A continuación demostramos la verdad de las fórmulas anteriores con respecto a los modelos de palabra e interpretando las variables con respecto a los conjuntos P_i de la definición (3.1).

Lema 3.2. $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_1$.

Demostración. Sin pérdida de generalidad, podemos tomar cualesquiera $i \neq j$ y demostrar sólo que $\hat{w} \models_{[\vec{P}/\vec{X}]} \forall x \neg(X_i(x) \wedge X_j(x))$. Sean $i, j \in Q$ con $i \neq j$.

$\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_1$

si y sólo si $\hat{w} \models_{[\vec{P}/\vec{X}]} \forall x \neg(X_i(x) \wedge X_j(x))$

si y sólo si $\hat{w} \models_{[\vec{P},p/\vec{X},x]} \neg(X_i(x) \wedge X_j(x))$ para toda $p \in \mathbf{n}$

si y sólo si $\hat{w} \models_{[\vec{P},p/\vec{X},x]} \neg X_i(x) \vee \neg X_j(x)$ para toda $p \in \mathbf{n}$

si y sólo si $\hat{w} \not\models_{[\vec{P},p/\vec{X},x]} X_i(x)$ o $\hat{w} \not\models_{[\vec{P},p/\vec{X},x]} X_j(x)$ para toda $p \in \mathbf{n}$

si y sólo si $p \notin P_i$ o $p \notin P_j$ para toda $p \in \mathbf{n}$

Ya que, por el lema (3.1), P_i y P_j son ajenos, se cumple que $p \notin P_i$ o $p \notin P_j$ para toda $p \in \mathbf{n}$.

□

Lema 3.3. $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_2$.

Demostración. $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_2$

si y sólo si $\hat{w} \models_{[\vec{P}/\vec{X}]} \forall x(\neg\exists y S(y,x) \to X_0(x))$

si y sólo si $\hat{w} \models_{[\vec{P},p/\vec{X},x]} \neg\exists y S(y,x) \to X_0(x)$ para toda $p \in \mathbf{n}$

si y sólo si $\hat{w} \models_{[\vec{P},p/\vec{X},x]} \exists y S(y,x)$ o $\hat{w} \models_{[\vec{P},p/\vec{X},x]} X_0(x)$ para toda $p \in \mathbf{n}$

si y sólo si $\hat{w} \models_{[\vec{P},p,q/\vec{X},x,y]} S(y,x)$ o $\hat{w} \models_{[\vec{P},p/\vec{X},x]} X_0(x)$ para alguna $q \in \mathbf{n}$ y para toda $p \in \mathbf{n}$

si y sólo si $p = q+1$ o $\rho_p = 0$ para toda $p \in \mathbf{n}$ y para alguna $q \in \mathbf{n}$

Sea $p \in \mathbf{n}$. Si $p = 0$, entonces $\rho_p = \rho_0 = 0$. Si $p \neq 0$ entonces $p = q+1$ para alguna $q \in \mathbf{n}$.

□

Lema 3.4. $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_3$.

Demostración. Sin pérdida de generalidad, podemos exhibir algunos $i, j \in Q$, $a \in \Sigma$, tales que $(i,a,j) \in \Delta$ y sólo tenemos que demostrar que $\hat{w} \models_{[\vec{P}/\vec{X}]} \forall x \forall y(S(x,y) \to (X_i(x) \wedge Q_a(x) \wedge X_j(y)))$. Sean $i, j \in Q$ y $a \in \Sigma$.

$\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_3$

si y sólo si $\hat{w} \models_{[\vec{P}/\vec{X}]} \forall x \forall y(S(x,y) \to (X_i(x) \wedge Q_a(x) \wedge X_j(y)))$

si y sólo si $\hat{w} \models_{[\vec{P},p,q/\vec{X},x,y]} S(x,y) \to (X_i(x) \wedge Q_a(x) \wedge X_j(y))$ para toda $p \in \mathbf{n}$ y para toda $q \in \mathbf{n}$

si y sólo si $\hat{w} \not\models_{[\vec{P},p,q/\vec{X},x,y]} S(x,y)$ o $\hat{w} \models_{[\vec{P},p,q/\vec{X},x,y]} X_i(x) \wedge Q_a(x) \wedge X_j(y)$ para toda $p \in \mathbf{n}$ y para toda $q \in \mathbf{n}$

si y sólo si $q \neq p+1$ o $p \in P_i$, $p \in Q_a^{\mathcal{I}}$, $q \in P_j$ para toda $p \in \mathbf{n}$ y para toda $q \in \mathbf{n}$

Sean $q, p \in \mathbf{n}$, en particular $a_p \in w$. Basta con ver que pasa cuando $q = p+1$. Si $q = p+1$ y ya que $a_p \in w$, entonces $(\rho_p, a_p, \rho_q) \in \Delta$. Tomemos $i = \rho_p, a = a_p, j = \rho_q$, por lo que $p \in P_i$, $p \in Q_a^{\mathcal{I}}$, $q \in P_j$.

□

Lema 3.5. $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_4$.

Demostración. Sin pérdida de generalidad, podemos exhibir algunos $i \in Q, j \in F$, $a \in \Sigma$ tales que $(i, a, j) \in \Delta$ y sólo tenemos que demostrar que $\hat{w} \models_{[\vec{P}/\vec{X}]} \forall x(\neg\exists y S(x, y) \to (X_i(x) \wedge Q_a(x)))$. Sean $i \in Q$, $j \in F$ y $a \in \Sigma$.

$\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_4$

si y sólo si $\hat{w} \models_{[\vec{P}/\vec{X}]} \forall x(\neg\exists y S(x, y) \to (X_i(x) \wedge Q_a(x)))$

si y sólo si $\hat{w} \models_{[\vec{P},p/\vec{X},x]} \neg\exists y S(x, y) \to (X_i(x) \wedge Q_a(x))$ para toda $p \in \mathbf{n}$

si y sólo si $\hat{w} \models_{[\vec{P},p/\vec{X},x]} \exists y S(x, y)$ o $\hat{w} \models_{[\vec{P},p/\vec{X},x]} X_i(x) \wedge Q_a(x)$ para toda $p \in \mathbf{n}$

si y sólo si $\hat{w} \models_{[\vec{P},p,q/\vec{X},x,y]} S(x, y)$ o para alguna $q \in \mathbf{n}$; $\hat{w} \models_{[\vec{P},p/\vec{X},x]} X_i(x) \wedge Q_a(x)$ para toda $p \in \mathbf{n}$

si y sólo si $q = p + 1$ o para alguna $q \in \mathbf{n}$; $p \in P_i$, $p \in Q_a^{\mathcal{I}}$ para toda $p \in \mathbf{n}$

Sean $q, p \in \mathbf{n}$, en particular $a_p \in w$. Basta con ver que pasa cuando $q \neq p+1$. Si $q \neq p+1$ para toda $q \in \mathbf{n}$, entonces $p = n-1$ y $\rho_{p+1} \in F$, además $a_p \in w$, por lo que $(\rho_p, a_p, \rho_{p+1}) \in \Delta$. Tomemos $i = \rho_p, a = a_p, j = \rho_{p+1}$, entonces $p \in P_i$, $p \in Q_a^{\mathcal{I}}$.

□

Cada modelo de palabra de $L(\mathcal{A})$ satisface las condiciones descritas por cada fórmula φ_i. Ahora podemos definir un enunciado $\varphi_{\mathcal{A}}$ que describa a toda secuencia de aceptación de $\mathcal{A}$ y verificar que todas las palabras de $L(\mathcal{A})$ lo satisfacen.

Lema 3.6. *Si* $\varphi_{\mathcal{A}} \equiv_{def} \exists X_0 ... \exists X_k(\varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4)$, *entonces* $\hat{w} \models \varphi_{\mathcal{A}}$.

Demostración. Por los lemas (3.2), (3.3), (3.4) y (3.5) tenemos que:

$\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_1$ y $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_2$ y $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_3$ y $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_4$

si y sólo si $\hat{w} \models_{[\vec{P}/\vec{X}]} \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4$

si y sólo si $\hat{w} \models \exists X_0 ... \exists X_k(\varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4)$

si y sólo si $\hat{w} \models \varphi_{\mathcal{A}}$

□

El enunciado $\varphi_{\mathcal{A}}$ describe las condiciones necesarias de cualquier ejecución de aceptación del autómata $\mathcal{A}$ y por lo que toda palabra, aceptada por $\mathcal{A}$, cumple. Además, $\varphi_{\mathcal{A}}$ determina al lenguaje $L(\varphi_{\mathcal{A}})$.

Teorema 3.1. *Dado $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ un autómata finito. Existe φ en S1S tal que $w \in L(\mathcal{A})$ si y sólo si $w \in L(\varphi)$.*

Demostración. Como $w \in L(\mathcal{A})$, por el lema (3.6), podemos construir $\varphi_{\mathcal{A}}$ tal que $\hat{w} \models \varphi_{\mathcal{A}}$.

□

El teorema anterior nos asegura que todo lenguaje regular es S1S-definible. Lo siguiente es ver que el recíproco también se cumple.

3.1.2. De lógica a autómatas

Para ir de la lógica S1S a los autómatas finitos, veremos que dada $\varphi \in$ S1S, existe algún AF $\mathcal{A}$ tal que $L(\varphi) = L(\mathcal{A})$. Para esto, utilizaremos la equivalencia entre S1S y S1S0. Construiremos, de forma inductiva, un autómata $\mathcal{A}$ que acepte el lenguaje determinado por una fórmula $\psi \in$ S1S0 equivalente a una fórmula $\varphi \in$ S1S. La razón de hacerlo a través de las fórmulas de S1S0 es que simplifica considerablemente el proceso de la construcción de los autómatas.

Ya que procederemos de forma inductiva sobre la estructura de las fórmulas de S1S0, necesitamos tomar en cuenta las fórmulas con variables libres. Para esto, definiremos una función γ que codificará cierta información de los subconjuntos de $\mathbf{n}$ sobre los que se interpretarán dichas variables.

Las siguiente función nos será de utilidad.

Definición 3.3. *Sean $i \in \mathbf{n}$ y $P \subseteq \mathbf{n}$. La función $\mu : \mathbf{n} \times \mathcal{P}(\mathbf{n}) \to \{0, 1\}$ determina la pertenencia de un elemento en un subconjunto de $\mathbf{n}$ y se define como:*

$$\mu(i, P) = \begin{cases} 1 & i \in P \\ 0 & i \notin P \end{cases}$$

Extendemos μ para k subconjuntos de $\mathbf{n}$: Si $\vec{P} = (P_1, ..., P_k)$, entonces $\mu(i, \vec{P}) = (\mu(i, P_1), ..., \mu(i, P_k))$.

Definición 3.4. *Sean $k \geq 0$, $w = a_0...a_{n-1}$ en Σ^* y $\vec{P} = (P_1, ..., P_k), k$ subconjuntos de $\mathbf{n}$. $\gamma : \Sigma^* \times \mathcal{P}(\mathbf{n})^k \to \Sigma^*_{\times k}$ es una función (o familia de funciones) de codificación tal que:*

$$\gamma(w, \vec{P}) = ((a_0, \mu(0, \vec{P}))...(a_{n-1}, \mu(n-1, \vec{P})))$$

con $\Sigma_{\times k} = \Sigma \times \{0, 1\}^k$.

En particular, si $k = 0$, entonces $\gamma : \Sigma^* \to \Sigma^*$ y $\gamma(w) = w$.

La función de codificación γ toma una palabra de Σ^* y k subconjuntos de $\mathbf{n}$ y devuelve una palabra sobre el alfabeto $\Sigma_{\times k}$ de manera que, si $w' = a'_0...a'_{n-1} = \gamma(w, P_1, ..., P_k)$, entonces $a'_i = (a_i, b_1, ..., b_k)$, donde cada $b_j = 1$ si y sólo si $i \in P_j$. Así, la pertenencia de i en cada P_j está codificada en el $i-$ésimo símbolo de w'.

Ejemplo. *Si $\Sigma = \{a, b, c\}, w = abca, P_1 = \{0, 3\}, P_2 = \{1, 2\}, P_3 = \{0, 1, 3\}$, entonces:*

- $\gamma(w, P_1) = \gamma(abca, P_1) = (a, 1)(b, 0)(c, 0)(a, 1)$
- $\gamma(w, P_2) = \gamma(abca, P_2) = (a, 0)(b, 1)(c, 1)(a, 0)$
- $\gamma(w, P_3) = \gamma(abca, P_3) = (a, 1)(b, 1)(c, 0)(a, 1)$
- $\gamma(w, P_1, P_2) = \gamma(abca, P_1, P_2) = (a, 1, 0)(b, 0, 1)(c, 0, 1)(a, 1, 0)$
- $\gamma(w, P_2, P_3) = \gamma(abca, P_2, P_3) = (a, 0, 1)(b, 1, 1)(c, 1, 0)(a, 0, 1)$
- $\gamma(w, P_1, P_2, P_3) = \gamma(abca, P_1, P_2, P_3) =$
  $(a, 1, 0, 1)(b, 0, 1, 1)(c, 0, 1, 0)(a, 1, 0, 1)$
- $\gamma(w, P_2, P_3, P_1) = \gamma(abca, P_2, P_3, P_1) =$
  $(a, 0, 1, 1)(b, 1, 1, 0)(c, 1, 0, 0)(a, 0, 1, 1)$

Con esto, construiremos un autómata por cada fórmula φ de S1S0 y demostraremos que cada palabra es aceptada por $\mathcal{A}$ si y sólo si está en $L(\varphi)$. Primero los haremos para cada una de las fórmulas atómicas de S1S0, las cuales servirán de caso base para la demostración por inducción sobre cualquier fórmula.

Lema 3.7. *Sea $\varphi \equiv_{def} \top$. Existe un autómata $\mathcal{A}_\varphi$ tal que, para toda $w \in \Sigma^*$:*

$$\breve{w} \models \top \text{ si y sólo si } \gamma(w) \in L(\mathcal{A}_\varphi)$$

Demostración. Sean $w \in \Sigma^*$ y $\mathcal{A}_\varphi = (Q, \Sigma, q_0, \Delta, F)$ tales que:

$$Q = \{q_0\}$$
$$F = \{q_0\}$$

La relación Δ consta de las siguientes transiciones:

$$(q_0, \sigma, q_0) \in \Delta$$

con $\sigma \in \Sigma$.

Representado gráficamente:

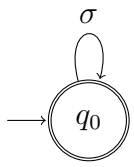


Figura: $\top$

Observese que $L(\mathcal{A}_\varphi) = \Sigma^*$, además $w = \gamma(w)$ por lo que $\gamma(w) \in L(\mathcal{A}_\varphi)$. Por otro lado, como $\models \top$, en particular $\breve{w} \models \top$.

Por lo tanto, siempre es el caso que $\gamma(w) \in L(\mathcal{A}_\varphi)$ y que $\breve{w} \models \top$.

□

Lema 3.8. *Sea $\varphi \equiv_{def} X_1 \subseteq X_2$, con X_1, X_2 libres. Existe un autómata $\mathcal{A}_\varphi$ tal que, para toda $w \in \Sigma^*$:*

$$\breve{w} \models_{[\vec{X}/\vec{P}]} X_1 \subseteq X_2 \text{ si y sólo si } \gamma(w, \vec{P}) \in L(\mathcal{A}_\varphi)$$

Demostración. Sean $P_1, P_2 \subseteq \mathbf{n}$ y $w' \in \Sigma_{\times 2}$, tales que $w' = \gamma(w, \vec{P})$ y sea $\mathcal{A}_\varphi = (Q, \Sigma_{\times 2}, q_0, \Delta, F)$ tal que:

$$Q = \{q_0, q_1\}$$
$$F = \{q_0\}$$

La relación Δ consta de las siguientes transiciones:

$$(q_0, (\sigma, 1, 1), q_0) \in \Delta$$
$$(q_0, (\sigma, 0, b_2), q_0) \in \Delta$$
$$(q_0, (\sigma, 1, 0), q_1) \in \Delta$$
$$(q_1, (\sigma, b_1, b_2), q_1) \in \Delta$$

con $\sigma \in \Sigma$.

Representado gráficamente:

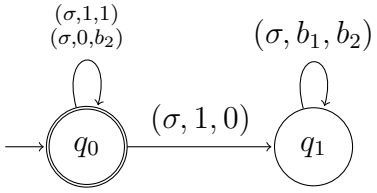


Figura: $X_1 \subseteq X_2$

Observemos que $\gamma(w, \vec{P}) \in L(\mathcal{A}_\varphi)$ si y sólo si $b_1 = b_2 = 1$ o $b_1 = 0$ para todo símbolo $a'_p = (a_p, b_1, b_2)$ de w'. Con lo que:

$p \in P_1, p \in P_2$ o $p \notin P_1$ para toda $p \in \mathbf{n}$

si y sólo si $P_1 \subseteq P_2$
si y sólo si $\breve{w} \models_{[\vec{X}/\vec{P}]} X_1 \subseteq X_2$

□

Lema 3.9. *Sea $\varphi \equiv_{def} X \subseteq Q_a$, con X libre. Existe un autómata $\mathcal{A}_\varphi$ tal que, para toda $w \in \Sigma^*$:*

$$\breve{w} \models_{[X/P]} X \subseteq Q_a \text{ si y sólo si } \gamma(w, P) \in L(\mathcal{A}_\varphi)$$

Demostración. Sean $P \subseteq \mathbf{n}$ y $w' \in \Sigma_{\times 1}$, tales que $w' = \gamma(w, P)$ y sea $\mathcal{A}_\varphi = (Q, \Sigma_{\times 1}, q_0, \Delta, F)$ tal que:

$$Q = \{q_0, q_1\}$$
$$F = \{q_0\}$$

La relación Δ consta de las siguientes transiciones:

$$(q_0, (\sigma, 0), q_0) \in \Delta$$
$$(q_0, (a, b_1), q_0) \in \Delta$$
$$(q_0, (\rho, 1), q_1) \in \Delta$$
$$(q_1, (\sigma, b_1), q_1) \in \Delta$$

con $\sigma, \rho \in \Sigma, \rho \neq a$.
Representado gráficamente:

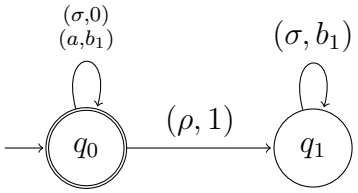


Figura: $X \subseteq Q_a$

Observemos que $\gamma(w, P) \in L(\mathcal{A}_\varphi)$ si y sólo si, para cualquier símbolo $a'_p = (a_p, b_1)$ de w', se tiene que $a_p = a$ o $b_1 = 0$. Con lo que:

$p \notin P$ o $p \in Q_a^{\mathcal{I}}$ para toda $p \in \mathbf{n}$

$$\begin{array}{ll} \text{si y sólo si} & P \subseteq Q_a^{\mathcal{I}} \\ \text{si y sólo si} & \breve{w} \models_{[X/P]} X \subseteq Q_a \end{array}$$

□

Lema 3.10. *Sea $\varphi \equiv_{def} S(X_1, X_2)$, con X_1, X_2 libres. Existe un autómata $\mathcal{A}_\varphi$ tal que, para toda $w \in \Sigma^*$:*

$$\breve{w} \models_{[\vec{X}/\vec{P}]} S(X_1, X_2) \text{ si y sólo si } \gamma(w, \vec{P}) \in L(\mathcal{A}_\varphi)$$

Demostración. Sean $P_1, P_2 \subseteq \mathbf{n}$ y $w' \in \Sigma_{\times 2}$, tales que $w' = \gamma(w, \vec{P})$ y sea $\mathcal{A}_\varphi = (Q, \Sigma_{\times 2}, q_0, \Delta, F)$ tal que:

$$\begin{array}{c} Q = \{q_0, q_1, q_2, q_3\} \\ F = \{q_2\} \end{array}$$

La relación Δ consta de las siguientes transiciones:

$$\begin{array}{ccc} (q_0, (\sigma, 0, 0), q_0) \in \Delta & (q_0, (\sigma, 1, 0), q_1) \in \Delta & (q_0, (\sigma, 0, 1), q_3) \in \Delta \\ (q_1, (\sigma, 0, 1), q_2) \in \Delta & (q_1, (\sigma, 0, 0), q_3) \in \Delta & (q_1, (\sigma, 1, b_2), q_3) \in \Delta \\ (q_2, (\sigma, 0, 0), q_2) \in \Delta & (q_2, (\sigma, 1, b_2), q_3) \in \Delta & (q_2, (\sigma, b_1, 1), q_3) \in \Delta \\ (q_3, (\sigma, b_1, b_2), q_3) \in \Delta & & \end{array}$$

con $\sigma \in \Sigma$.
Representado gráficamente:

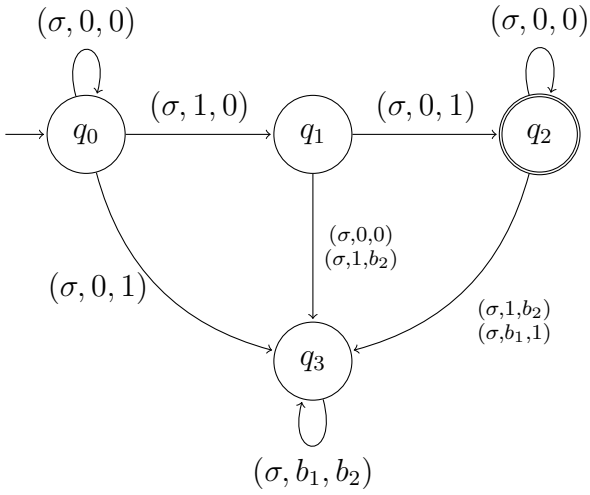


Figura: $S(X_1, X_2)$

Observemos que $\gamma(w, \vec{P}) \in L(\mathcal{A}_\varphi)$ si y sólo si, para algunos símbolos $a'_p = (a_p, b_1, b_2)$, $a'_q = (a_q, b_1, b_2)$ de w', se tiene que $q = p + 1$, $a'_p = (a_p, 1, 0)$, $a'_q = (a_q, 0, 1)$ y además $a'_j = (a_j, 0, 0)$ si $j \neq p, j \neq q$. Es decir, $p \in P_1, p+1 \in P_2$ y si $j \neq p, j \neq p+1$, entonces $j \notin P_1, j \notin P_2$, esto es si y sólo si $P_1 = \{p\}$ y $P_2 = \{p+1\}$ y esto pasa si y sólo si $\breve{w} \models_{[\vec{X}/\vec{P}]} \mathcal{S}(X_1, X_2)$.

□

Así, los lemas (3.8), (3.8), (3.9) y (3.10) sirven como caso base para la demostración por inducción sobre cualquier fórmula de S1S0.

Teorema 3.2. *Para cada fórmula $\varphi(X_1, ..., X_k) \in$ S1S0, con $X_1, ...X_k$ variables libres, existe un autómata $\mathcal{A}_\varphi$ tal que:*

$$\breve{w} \models_{[\vec{X}/\vec{P}]} \varphi(X_1, ..., X_k) \text{ si y sólo si } \gamma(w, \vec{P}) \in L(\mathcal{A}_\varphi)$$

.

Demostración. Procedemos por inducción sobre las fórmulas de S1S0.

1. Supongamos que $\varphi \equiv_{def} \neg\psi$. Por hipótesis de inducción, hay un AF $\mathcal{B}_\psi = (Q, \Sigma_{\times k}, q_0, \Delta, F)$ tal que $w \in L(\psi)$ si y sólo si $w \in L(\mathcal{B}_\psi)$.

   Ya que $L(\neg\psi) = \Sigma^* \setminus L(\psi)$ y el conjunto de lenguajes reconocibles por un AF es cerrado bajo complemento, entonces, hay un AF $\mathcal{A}_\varphi$ tal que $w \in L(\neg\psi)$ si y sólo si $w \in L(\mathcal{A}_\varphi)$.

   Nos basta con tomar $\mathcal{A}_\varphi = (Q, \Sigma_{\times k}, q_0, \Delta, F')$, con $F' = Q \setminus F$.

2. Supongamos que $\varphi \equiv_{def} \psi \vee \mu$. Por hipótesis de inducción, existen los autómatas finitos $\mathcal{C}_\psi = (Q_\mathcal{C}, \Sigma_{\times k}, q_\mathcal{C}, \Delta_\mathcal{C}, F_\mathcal{C})$ tal que $w \in L(\psi)$ si y sólo si $w \in L(\mathcal{C}_\psi)$ y $\mathcal{B}_\mu = (Q_\mathcal{B}, \Sigma_{\times j}, q_\mathcal{B}, \Delta_\mathcal{B}, F_\mathcal{B})$ tal que $w \in L(\mu)$ si y sólo si $w \in L(\mathcal{B}_\mu)$. Sin pérdida de generalidad, podemos suponer que $Q_\mathcal{C} \cap Q_\mathcal{B} = \emptyset$, con lo que $\Delta_\mathcal{C} \cap \Delta_\mathcal{B} = \emptyset$.

   Ya que $L(\psi \vee \mu) = L(\psi) \cup L(\mu)$ y el conjunto de lenguajes reconocibles por un AF es cerrado bajo unión, entonces, hay un AF $\mathcal{A}_\varphi$ tal que $w \in L(\psi \vee \mu)$ si y sólo si $w \in L(\mathcal{A}_\varphi)$.

   Nos basta con tomar $\mathcal{A}_\varphi = (Q, \Sigma, q_0, \Delta, F)$, donde:

   - $Q = Q_\mathcal{C} \cup Q_\mathcal{B} \cup \{q_0\}$
   - $\Sigma = \Sigma_{\times k} \cup \Sigma_{\times j}$
   - $F = F_\mathcal{C} \cup F_\mathcal{B}$
   - $\Delta = \Delta_\mathcal{C} \cup \Delta_\mathcal{B} \cup \bigcup_{q \in Q_\mathcal{C}} \{(q_0, a, q) \mid (q_\mathcal{C}, a, q) \in \Delta_\mathcal{C}\}) \cup \bigcup_{q \in Q_\mathcal{B}} \{(q_0, a, q) \mid (q_\mathcal{B}, a, q) \in \Delta_\mathcal{B}\})$

3. Supongamos que $\varphi(X_1, ..., X_k) \equiv_{def} \exists X_{k+1} \psi(X_1, ..., X_{k+1})$, con $X_1, ..., X_k$ variables libres en φ. Por hipótesis de inducción, existe un autómata $\mathcal{B}_\psi = (Q, \Sigma_{\times k+1}, q_0, \Delta_\mathcal{B}, F)$ tal que $\gamma(w, \vec{P}, P_{k+1}) \in L(\mathcal{B}_\psi)$ si y sólo si $\check{w} \models_{[\vec{X}, X_{k+1}\ \vec{P}, P_{k+1}]} \psi(X_1, ..., X_{k+1})$.

   Utilizamos $\mathcal{B}_\psi$ para construir $\mathcal{A}_\varphi$ tal que $\gamma(w, \vec{P}) \in L(\mathcal{A}_\varphi)$ si y sólo si $\check{w} \models_{[\vec{X}/\vec{P}]} \exists X_{k+1} \psi(X_1, ...X_{k+1})$.

   Nos basta con tomar $\mathcal{A}_\varphi = (Q, \Sigma_{\times k}, q_0, \Delta, F)$, donde:

$$(q_i, (a, b_1, ..., b_k), q_j) \in \Delta$$
$$\text{si y sólo si}$$
$$(q_i, (a, b_1, ..., b_k, b_{k+1}), q_j) \in \Delta_\mathcal{B} \text{ para algún } b_{k+1}$$

   Sean $w_k \in \Sigma_{\times k}$ y $w_{k+1} \in \Sigma_{\times k+1}$, tales que $w_k = \gamma(w, \vec{P})$ y $w_{k+1} = \gamma(w, \vec{P}, P_{k+1})$.

   Supongamos que $w_{k+1} \in L(\mathcal{B}_\psi)$, por lo tanto, existe una ejecución de aceptación $\rho = \rho_0, ..., \rho_n$ de $\mathcal{B}_\psi$ para w_{k+1} con $\rho_0 = q_0$, $(\rho_i, w_{k+1}(i), \rho_{i+1}) \in \Delta_\mathcal{B}$ y $\rho_n \in F$.

Por la construcción de $\mathcal{A}_\varphi$, $(\rho_i, w_{k+1}(i), \rho_{i+1}) \in \Delta_\mathcal{B}$ si y sólo si $(\rho_i, w_k(i), \rho_{i+1}) \in \Delta$, por lo que ρ es una ejecución de aceptación de $\mathcal{A}_\varphi$ para w_k.

Por lo tanto:

$$\begin{array}{lll} \gamma(w, \vec{P}) \in L(\mathcal{A}_\varphi) & \text{si y sólo si} & \gamma(w, \vec{P}, P_{k+1}) \in L(\mathcal{B}_\psi) \\ & \text{si y sólo si} & \breve{w} \models_{[\vec{X}, X_{k+1}/\vec{P}, P_{k+1}]} \psi(X_1, ..., X_{k+1}) \quad \text{H.I.} \\ & \text{si y sólo si} & \breve{w} \models_{[\vec{X}/\vec{P}]} \exists X_{k+1} \psi(X_1, ..., X_{k+1}) \\ & \text{si y sólo si} & \breve{w} \models_{[\vec{X}/\vec{P}]} \varphi(X_1, ..., X_k) \end{array}$$

Con esto terminamos la inducción y la prueba del teorema (3.2).

□

Ahora consideramos los enunciados de S1S0.

Corolario. *Si φ es un enunciado de S1S0, entonces existe un AF $\mathcal{A}_\varphi$ tal que $L(\varphi) = L(\mathcal{A}_\varphi)$.*

Demostración. Por el teorema (3.2), existe un AF $\mathcal{A}_\varphi$ tal que $\breve{w} \models \varphi$ si y sólo si $\gamma(w, P_1, ..., P_k) \in L(\mathcal{A}_\varphi)$. Ahora, como φ es un enunciado y no tiene variables libres, entonces $k = 0$, por lo que $\gamma(w, P_1, ..., P_k) = \gamma(w) = w$ y por lo tanto, $\breve{w} \models \varphi$ si y sólo si $w \in L(\mathcal{A}_\varphi)$.

□

Ahora ya podemos establecer el vínculo entre la lógica S1S y los autómatas finitos.

Teorema 3.3. *Dada $\varphi \in$ S1S, existe un autómata finito $\mathcal{A}_\varphi$ tal que $w \in L(\varphi)$ si y sólo si $w \in L(\mathcal{A}_\varphi)$.*

Demostración. Sabemos que S1S0 es equivalente a S1S. Sean $\varphi \in$ S1S y $\psi \in$ S1S0 tales que $\psi = \varphi^\bullet$, por lo que $\breve{w} \models \psi$ si y sólo si $\hat{w} \models \varphi$. Por el teorema (3.2), existe un autómata $\mathcal{A}_\psi$ tal que $\breve{w} \models \psi$ si y sólo si $w \in L(\mathcal{A}_\psi)$. Basta con tomar $\mathcal{A}_\varphi = \mathcal{A}_\psi$, por lo que $w \in L(\varphi)$ si y sólo si $w \in L(\mathcal{A}_\varphi)$.

□

Ya tenemos todas las bases para poder enunciar y demostrar el teorema de Büchi/Elgot para lenguajes regulares.

Teorema 3.4 (**Teorema de Büchi/Elgot**). *Un lenguaje de palabras finitas es regular si y sólo si es S1S-definible.*

Demostración. ($\Rightarrow$). Del teorema (3.1) tenemos que: Dado $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ un autómata finito. Si $w \in L(\mathcal{A}) \subseteq \Sigma^*$, entonces existe $\varphi \in$ S1S tal que $w \in L(\varphi)$.

($\Leftarrow$). Del teorema (3.3) tenemos que: Dada $\varphi \in$ S1S, existe un autómata finito $\mathcal{A}_\varphi$ tal que $w \in L(\varphi)$ si y sólo si $w \in L(\mathcal{A}_\varphi)$.

□

Un resultado importante es la decidibilidad de S1S con interpretación sobre **n** que se sigue de la demostración del teorema (3.4).

Corolario 3.1. *La lógica S1S es decidible. Es decir, dada una fórmula φ en S1S, podemos determinar si existe o no $w \in \Sigma^*$ tal que $\hat{w} \models \varphi$.*

Demostración. Sea φ una fórmula en S1S. Por el teorema (3.1), podemos construir un AF $\mathcal{A}_\varphi$ tal que $w \in L(\mathcal{A}_\varphi)$ si y sólo si $w \in L(\varphi)$, para toda $w \in \Sigma^*$.

Por el lema (1.1), podemos determinar si $L(\mathcal{A}_\varphi) \neq \emptyset$, es decir, podemos saber si existe $w \in \Sigma^*$ tal que $w \in L(\mathcal{A}_\varphi)$.

□

Hemos establecido la equivalencia entre los lenguajes regulares y los lenguajes definibles en la lógica S1S interpretada sobre subconjuntos finitos de $\mathbb{N}$. Lo siguiente es ver que dicha equivalencia existe, de forma análoga, en el caso de una generalización sobre palabras infinitas, es decir, en el caso de los lenguajes ω-regulares y la lógica S1S interpretada sobre $\mathbb{N}$.

3.2. Autómatas de Büchi

Con los autómatas finitos podemos modelar problemas en los que se espera leer una entrada finita, hacer algún cálculo y terminar. Sin embargo, muchos problemas no son de ese tipo. Por ejemplo, consideremos:

- Un sistema operativo.
- Un sistema de control aéreo.
- Un sistema de control de procesos de una fábrica.

Idealmente, estos sistemas podrían procesar un número infinito de entradas y nunca terminar su ejecución. Para modelar estos sistemas necesitamos extender la definición usual de autómata finito.

Un autómata de Büchi es un autómata finito que recibe como entrada palabras infinitas. Una palabra infinita es aceptada si, al menos un estado final es visitado un número infinito de veces. Las siguientes definiciones formalizan dichos conceptos.

Definición 3.5. *Una palabra infinita u ω-palabra α sobre un alfabeto Σ, es una secuencia infinita de símbolos, $\alpha = a_0a_1...$, donde cada $a_i \in \Sigma$.*

Ejemplo. *Si $\Sigma = \{a, b, c\}$. Las siguientes son ω-palabras sobre Σ:*

- $aaaaaaaa\ldots$
- $abababab\ldots$
- $cabccbcccccccc\ldots$

Podemos representar a α como una función $\alpha : \mathbb{N} \to \Sigma$, con lo que $\alpha(i)$ denota el símbolo que está en la i-ésima posición: $\alpha(i) = a_i$.

Ejemplo. *Si $\alpha = cabccbcccccccc\ldots$, entonces, $\alpha(0) = c$, $\alpha(1) = a$, $\alpha(5) = b$.*

Definición 3.6. *Σ^ω es el conjunto de todas las palabras infinitas sobre el alfabeto Σ.*

Definición 3.7. *Sea $L \subseteq \Sigma^*$ un lenguaje de palabras finitas. El conjunto $L^\omega \subseteq \Sigma^\omega$ es el conjunto de palabras infinitas formadas a partir de la concatenación de un número infinito de palabras de L:*

$$L^\omega = \{\alpha \in \Sigma^\omega \mid \alpha = w_0w_1w_2\ldots,\ \text{con } w_i \in L\}$$

Decimos que L^ω es la ω-potencia de L.

Definición 3.8. *Si $\alpha \in \Sigma^\omega, i, j \in \mathbb{N}, i < j$, el segmento de α desde i hasta $j - 1$ lo definimos como:*

$$\alpha(i, j) = \alpha(i)\alpha(i + 1)...\alpha(j - 1)$$

Ejemplo. *Si $\alpha = cabccbcccccccc\ldots$, entonces, $\alpha(0, 1) = c$, $\alpha(0, 5) = cabcc$, $\alpha(5, 7) = bc$.*

Definición 3.9. *Sean $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ un autómata finito y $\alpha \in \Sigma^\omega$. Una ejecución ρ_α de $\mathcal{A}$ sobre α, es una secuencia infinita de estados $\rho_\alpha = \rho_0\rho_1...$ tal que $\rho_0 = q_0$ y $(\rho_i, a_i, \rho_{i+1}) \in \Delta$, con $0 \leq i$.*

Si $\mathcal{A}$ es un autómata finito y $\alpha \in \Sigma^\omega$, en cualquier ejecución ρ_α de $\mathcal{A}$ para α, algunos estados son visitados sólo un número finito de veces y otros un número infinito. Llamamos a estos conjuntos *fin*(ρ_α) e *infty*(ρ_α), respectivamente, y los definimos de la siguiente manera:

Definición 3.10. *El conjunto de estados que son visitados un número infinito de veces es:*

$$\mathit{infty}(\rho_\alpha) = \{q \in Q \mid \exists^\omega i \in \mathbb{N} : \rho_i = q\}$$

donde $\exists^\omega$ denota al cuantificador "existe una infinidad".

Definición 3.11. *El conjunto de estados que son visitados sólo un número finito de veces es:*

$$\mathit{fin}(\rho_\alpha) = \{q \in Q \mid \exists i \text{ tal que } \forall j > i : \rho_j \neq q\}$$

Observese que $Q \setminus \mathit{infty}(\rho_\alpha) = \mathit{fin}(\rho_\alpha)$.

Proposición 3.1. *Hay un sufijo infinito de la ejecución ρ_α en donde ninguno de los estados de fin(ρ_α) es visitado y sólo son visitados los estados de infty(ρ_α).*

Demostración. Como Q es finito, entonces *fin*(ρ_α) es finito. Por definición de *fin*(ρ_α), sabemos que $\forall q \in \mathit{fin}(\rho_\alpha)$, $\exists i$ tal que $\forall j > i$, $\rho_j \neq q$.

Sea k el máximo número de esas i's, entonces $\forall q \in \mathit{fin}(\rho_\alpha)$ y $\forall j > k$, $\rho_j \neq q$. Supongamos que existe $l > k$ tal que $\rho_l \in \mathit{fin}(\rho_\alpha)$, eso contradice la suposición de que $\forall j > k$, $\rho_j \neq q$.

Por lo tanto $\forall j > k$, $\rho_j \notin \mathit{fin}(\rho_\alpha)$ y además como ρ_α es una secuencia infinita, $\forall j > k$, $\rho_j \in \mathit{infty}(\rho_\alpha)$. Es decir, $\rho_{k+1}\rho_{k+2}\rho_{k+3}\ldots$ es un sufijo infinito con estados en *infty*(ρ_α).

□

Es razonable asumir que la clasificación de una ejecución, como de aceptación o de rechazo, recae en su comportamiento en *el límite* y por lo tanto debe depender sólo de *infty*(ρ_α). Büchi sugirió clasificar la ejecución como de aceptación si visita algún estado en F un número infinito de veces. Cómo sólo hay un número finito de estados en Q y F, esto es equivalente en pedir que la ejecución visite algún estado fijo en F un número infinito de veces.

Definición 3.12. *Un autómata de Büchi $\mathcal{A}$ es un autómata finito tal que $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ y el lenguaje aceptado por dicho autómata es:*

$$L(\mathcal{A}) = \{\alpha \mid \exists \rho_\alpha : \mathit{infty}(\rho_\alpha) \cap F \neq \emptyset\}$$

Observese que la única diferencia entre un autómata finito y un autómata de Büchi recae solamente en la condición de aceptación. También es importante notar que para cualquier autómata de Büchi $\mathcal{A}$, ninguna palabra finita es aceptada puesto que la definición de $L(\mathcal{A})$ depende de una ejecución infinita ρ_α de $\mathcal{A}$ y del conjunto $\mathit{infty}(\rho_\alpha)$.

Definición 3.13. *Una ejecución $\rho_\alpha = \rho_0\rho_1\rho_2\ldots$ de $\mathcal{A}$ para $\alpha \in \Sigma^\omega$, es una ejecución de aceptación si $\exists^\omega i \in \mathbb{N}$: $\rho_i \in F$. Decimos que α es aceptada por $\mathcal{A}$ si existe una ejecución de aceptación de $\mathcal{A}$ para α.*

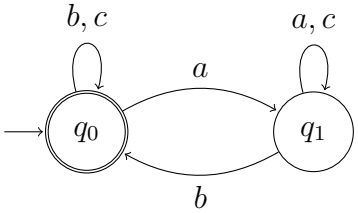
3.2.1. Lenguajes ω-regulares

Damos a continuación, a partir de un autómata de Büchi, la definición de los lenguajes ω-regulares.

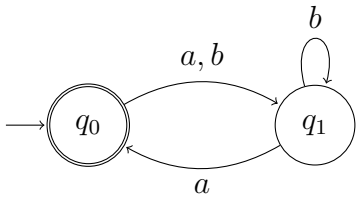
Definición 3.14. *Un lenguaje $L \subseteq \Sigma^\omega$ se dice que es ω-regular si es aceptado por un autómata de Büchi.*

Los lenguajes ω-regulares cumplen con las mismas propiedades de cerradura de los lenguajes regulares y también pueden ser definidos a partir de ellos, como demostraremos más adelante. Por esto, los lenguajes ω-regulares se pueden ver como una generalización, sobre palabras infinitas, de los lenguajes regulares.

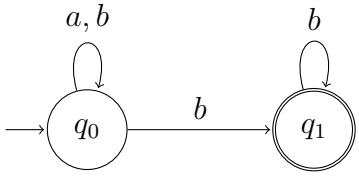
Ejemplo. *Los siguientes son autómatas de Büchi.*



Este autómata acepta todas las ω-palabras sobre $\{a, b, c\}$ en las cuales, no hay símbolo a o toda a tiene una b en algún lugar a su derecha.



Este autómata acepta todas las ω-palabras sobre $\{a, b\}$ que tienen un número infinito de símbolos a.



Este autómata acepta todas las ω-palabras sobre $\{a, b\}$ que tienen un número finito de símbolos a.

De los autómatas descritos anteriormente, sólo el último no es determinista. Llamaremos al lenguaje aceptado por ese autómata, Fin(a). Desafortunadamente, no podemos diseñar un autómata de Büchi determinista que acepte a Fin(a).

Proposición 3.2. *No hay un autómata de Büchi determinista que acepte únicamente al lenguaje Fin(a).*

Demostración. Supongamos que hay un autómata de Büchi determinista $\mathcal{A}$ que acepta únicamente a Fin(a). Este autómata debe de tener una ejecución de aceptación única sobre la palabra ab^ω (donde $b^\omega = bbbb...$). Supongamos que esta única ejecución entra a un estado en F después de ab^{n_1} para alguna $n_1 \geq 1$. Ahora, $ab^{n_1}ab^\omega$ también está en el lenguaje y hay una ejecución única para esta palabra, la cual extiende la ejecución para ab^{n_1} aceptándola y además, dicha ejecución, debe visitar un estado en F después de leer a $ab^{n_1}ab^{n_2}$ para alguna $n_2 \geq 1$. Repitiendo este argumento podemos construir una secuencia $ab^{n_1}ab^{n_2}...ab^{n_i}...$, en la que la ejecución única visita un estado en F una infinidad de veces y por lo tanto, la cadena con una infinidad de símbolos a es aceptada por $\mathcal{A}$. Esto contradice nuestra suposición de que $\mathcal{A}$ acepta únicamente el lenguaje Fin(a).

□

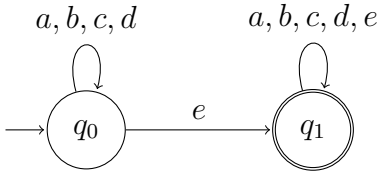
Por lo anterior, centraremos nuestra atención en los autómatas de Büchi no deterministas y nos referiremos a los mismos como ABN. Antes de ver las propiedades de cerradura de estos autómatas, veamos un par de aplicaciones interesantes.

Los sistemas concurrentes tienen un numeroso rango de aplicaciones entre las que se encuentran la inteligencia artificial, la física y el análisis del clima. Estos sistemas son construidos a partir de múltiples máquinas o procesadores y son utilizados en conjunto para resolver problemas especificos. Dichos problemas requieren un alto grado de complejidad computacional debibo a la naturaleza de los mismos. Al modelar sistemas concurrentes, se busca que estos cumplan correctamente con algun conjunto de requerimientos mediante programas llamados verificadores de modelos. Dos problemas principales que se presentan en este tipo de modelos son: las secuencias de eventos y la exclusión mutua.

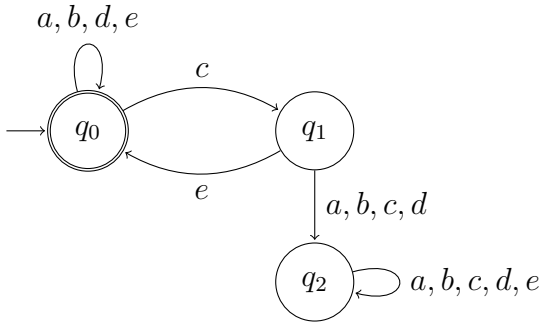
Es importante mencionar que en la programación concurrente, accesos concurrentes a recursos compartidos pueden llevar a comportamientos erróneos o inesperados, por lo que la parte del programa donde se accede a dichos recursos compartidos, está protegida. Esta sección protegida es conocida como *sección crítica* o *región crítica* y no puede ser ejecutada por más de un proceso a la vez. Típicamente, la región crítica accede a uno o más recursos compartidos como: una estructura de datos, un dispositivo periférico o una conexión de red, los cuales no permiten múltiples accesos concurrentes[19].

Secuencias de eventos: Supongamos que hay cinco tipos de eventos a, b, c, d, e, que pueden ocurrir en el sistema que queremos modelar. Sea $\Sigma = \{a, b, c, d, e\}$.

Primero consideremos el caso en el que requerimos que el evento e ocurra al menos una vez. El siguiente ABN acepta únicamente los elementos de Σ^ω que contienen almenos una ocurrencia de e:



Ahora supongamos que requerimos que todo evento c sea seguido imediatamente por un evento e. El siguiente ABN acepta únicamente los elementos de Σ^ω que satisfacen ese requerimiento:

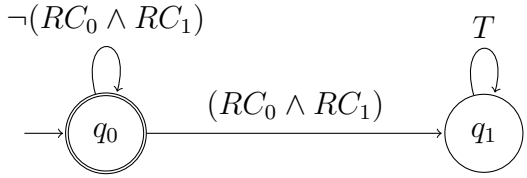


Exclusión mutua: Supongamos que queremos modelar un sistema concurrente con dos procesos y con la restricción de que nunca suceda que ambos procesos se encuentran en su región crítica al mismo tiempo. A esta restricción le llamamos exclusión mutua. Una forma de modelar el comportamiento de sistemas concurrentes complejos es permitir entradas que correspondan a expresiones booleanas que capturen las propiedades de interés. De esta manera, los mismos enunciados booleanos se pueden combinar para formar distintas expresiones en distintas máquinas que corresponderán a diferentes propiedades deseadas. Para capturar la restricción de exclusión mutua, usaremos los enunciados booleanos:

- RC_0, el cual tendrá el valor de verdad T (verdadero) si y sólo si el proceso p_0 está en su región crítica.
- RC_1, el cual tendrá el valor de verdad T (verdadero) si y sólo si el proceso p_1 está en su región crítica.

Con lo cual, el alfabeto Σ queda determinado por las expresiones booleanas: $RC_0 \wedge RC_1$, $\neg(RC_0 \wedge RC_1)$ y T. Es decir, $\Sigma = \{(RC_0 \wedge RC_1), \neg(RC_0 \wedge RC_1), T\}$.

El siguiente autómata de Büchi acepta únicamente las secuencias de entradas que satisfacen la propiedad de que RC_0 y RC_1 nunca ocurran:



A continuación veremos que los autómatas de Büchi también cumplen con las mismas propiedades de cerradura que cumplen los autómatas finitos.

3.2.2. Propiedades de cerradura

Probaremos las propiedades de cerradura de los lenguajes ω-regulares, y por consiguiente la de los autómatas de Büchi, con respecto a las operaciones usuales de conjuntos y a la concatenación. Lo haremos construyendo para cada propiedad, su correspondiente autómata.

Lema 3.11. *Si $V \subseteq \Sigma^*$ es regular, entonces V^ω es ω-regular.*

Demostración. Como V es regular, existe un AF $\mathcal{A} = (Q_\mathcal{A}, \Sigma, q_\mathcal{A}, \Delta_\mathcal{A}, F_\mathcal{A})$ que lo reconoce. Podemos suponer, sin pérdida de generalidad, que no existe alguna transición hacia $q_\mathcal{A}$ y que $\varepsilon \notin V$. Construimos el ABN $\mathcal{B}$ a partir de $\mathcal{A}$ de tal forma que $\mathcal{B} = (Q, \Sigma, q_\mathcal{A}, \Delta, F)$, con:

- $\Delta = \Delta_\mathcal{A} \cup \bigcup_{q_f \in F_\mathcal{A}} \{(q, a, q_\mathcal{A}) \mid (q, a, q_f) \in \Delta_\mathcal{A}\}$.
- $F = \{q_\mathcal{A}\}$.

Sea $\alpha \in V^\omega$, por lo que α es de la forma $\alpha = w_0 w_1 ...$, con $w_i \in V$. Sabemos que para toda $i \in \mathbb{N}$, $w_i \in V$ si y sólo si existe una ejecución de aceptación ρ_{w_i} de $\mathcal{A}$ sobre w_i tal que:

1. $\rho_{w_i} = \rho^i_0 ... \rho^i_{n_i}$.
2. $\rho^i_0 = q_\mathcal{A}$.
3. $\rho^i_{n_i} \in F_\mathcal{A}$.
4. $(\rho^i_{j_i}, w_i(j_i), \rho^i_{j_i+1}) \in \Delta_\mathcal{A}$, con $0 \leq j_i < n_i$.

Por definición de Δ, $(\rho^i_{n_i-1}, w_i(n_i - 1), q_\mathcal{A}) \in \Delta$. Por lo tanto, hay una ejecución ρ_α de $\mathcal{B}$ sobre α tal que $\rho_\alpha = \rho'_{w_0} \rho'_{w_1} \ldots$, siendo cada ρ'_{w_i} una subsecuencia de estados de la forma $\rho'_{w_i} = q_\mathcal{A} \rho^i_1 ... \rho^i_{n_i-1}$.

Notemos que después de la ejecución de cada w_i, se pasa por el estado $q_\mathcal{A} \in F$ y como esto sucede una infinidad de veces, $infty(\rho_\alpha) \cap F \neq \emptyset$. Por lo tanto, ρ_α es una secuencia de aceptación y $\alpha \in L(\mathcal{B})$.

Por otro lado, sea $\beta \in L(\mathcal{B})$, por lo que existe una ejecución de aceptación ρ_β de $\mathcal{B}$ para β. Por definición de F, el estado $q_\mathcal{A}$ es visitado un número infinito de veces en ρ_β, por lo cual, ρ_β es de la forma $\rho_\beta = \rho'_0 \rho'_1 \ldots$, siendo cada ρ'_i una subsecuencia de la forma $\rho'_i = \rho^i_0 \rho^i_1 \rho^i_2 \ldots \rho^i_{n_i}$, con $\rho^i_0 = q_\mathcal{A}$.

Por lo anterior, β es de la forma $\beta = w_0 w_1 \dots$ tal que para cada $i \in \mathbb{N}$, $(\rho^i_j, w_i(j), \rho^i_{j+1}) \in \Delta$, con $0 \leq j < n_i + 1$. Por definición de Δ, para cada transición $(\rho^i_{n_i}, w_i(n_i), q_{\mathcal{A}}) \in \Delta$, existe $q_i \in F_{\mathcal{A}}$ tal que $(\rho^i_{n_i}, w_i(n_i), q_i) \in \Delta_{\mathcal{A}}$, por lo cual, cada $w_i \in V$ y por lo tanto $\beta \in V^\omega$.

Por lo tanto $V^\omega = L(\mathcal{B})$ y V^ω es ω-regular.

□

Lema 3.12. *Si $U \subseteq \Sigma^*$ es regular y $L \subseteq \Sigma^\omega$ es ω-regular, entonces $U \cdot L$ es ω-regular.*

Demostración. Como U, L son regular y ω-regular, respectivamente, existe un autómata finito $\mathcal{A} = (Q_{\mathcal{A}}, \Sigma, q_{\mathcal{A}}, \Delta_{\mathcal{A}}, F_{\mathcal{A}})$ que reconoce a U y un autómata de Büchi $\mathcal{B} = (Q_{\mathcal{B}}, \Sigma, q_{\mathcal{B}}, \Delta_{\mathcal{B}}, F_{\mathcal{B}})$ que reconoce a L. Sin perder generalidad, podemos suponer que $Q_{\mathcal{A}} \cap Q_{\mathcal{B}} = \emptyset$.

Construimos el ABN $\mathcal{C}$ de tal forma que $\mathcal{C} = (Q_{\mathcal{A}} \cup Q_{\mathcal{B}}, \Sigma, q_{\mathcal{A}}, \Delta, F_{\mathcal{B}})$ con:

$$\Delta = \Delta_{\mathcal{A}} \cup \Delta_{\mathcal{B}} \ \cup \bigcup_{q_f \in F_{\mathcal{A}}} \{(q, a, q_{\mathcal{B}}) \mid (q, a, q_f) \in \Delta_{\mathcal{A}}\}$$

Sea $\alpha \in U \cdot L$, por lo que α es de la forma $\alpha = w\beta$, con $w \in U, \beta \in L$. Sabemos que $w \in U$ si y sólo si existe una ejecución de aceptación $\rho_w = \rho^w_0 ... \rho^w_n$, con $\rho^w_0 = q_{\mathcal{A}}$ y $\rho^w_n \in F_{\mathcal{A}}$. También sabemos que $\beta \in L$ si y sólo si existe una ejecución de aceptación $\rho_\beta = \rho^\beta_0 \rho^\beta_1 ...$, con $\rho^\beta_0 = q_{\mathcal{B}}$ tal que $\mathit{infty}(\rho_\beta) \cap F_{\mathcal{B}} \neq \emptyset$.

Sea ρ_α la ejecución de $\mathcal{C}$ sobre α. Por definición de Δ y como $\rho^w_n \in F_{\mathcal{A}}$, se tiene que $(\rho^w_{n-1}, w(n-1), q_{\mathcal{B}}) \in \Delta$. Con esto, la subsecuencia de estados ρ'_w para w es tal que $\rho'_w = q_{\mathcal{A}} \rho_1 ... \rho_{n-1}$, además, la subsecuencia de estados para β es ρ_β, ya que $\Delta_{\mathcal{B}} \subseteq \Delta$.

Por lo anterior, $\rho_\alpha = \rho'_w \rho_\beta$, además, es claro que $\mathit{infty}(\rho_\alpha) \cap F_{\mathcal{B}} \neq \emptyset$, pues $\mathit{infty}(\rho_\beta) \cap F_{\mathcal{B}} \neq \emptyset$, entonces ρ_α es una secuencia de aceptación y por lo tanto $\alpha \in L(\mathcal{C})$.

Por otro lado, sea $\beta \in L(\mathcal{C})$, entonces existe una secuencia de aceptación $\rho_\beta = \rho_0 \rho_1 \dots$ de $\mathcal{C}$ para β, con lo que $\mathit{infty}(\rho_\beta) \cap F_{\mathcal{B}} \neq \emptyset$. Por la definición de $\mathcal{C}$, $\rho_0 = q_{\mathcal{A}}$. Por la definición de Δ y dado que $Q_{\mathcal{A}} \cap Q_{\mathcal{B}} = \emptyset$, ρ_β es de la forma $\rho_\beta = \gamma_\beta \xi_\beta$, con $\rho_\beta = q_{\mathcal{A}} \gamma_1 \gamma_2 \dots \gamma_{n-1}$ y $\xi_\beta = q_{\mathcal{B}} \xi_1 \xi_2 \dots$.

Es claro que cada $\gamma_i \in Q_{\mathcal{A}}$, por lo que $(\gamma_i, \beta(i), \gamma_{i+1}) \in \Delta_{\mathcal{A}}$, con $0 \leq i < n$. De la misma manera, cada $\xi_j \in Q_{\mathcal{B}}$, por lo que $(\xi_j, \beta(n+j), \xi_{j+1}) \in \Delta_{\mathcal{B}}$.

Además, como $(\gamma_{n-i}, \beta(n-1), q_{\mathcal{B}}) \in \Delta$, existe $q \in F_{\mathcal{A}}$ tal que $(\gamma_{n-i}, \beta(n-1), q) \in \Delta_{\mathcal{A}}$.

Entonces, la secuencia $q_{\mathcal{A}}\gamma_1\gamma_2 \ldots \gamma_{n-1}q$ es una ejecución de aceptación de $\mathcal{A}$ para $\beta(0,n)$ y por lo tanto $\beta(0,n) \in U$. Por otro lado, dado que $infty(\rho_\beta) \cap F_{\mathcal{B}} \neq \emptyset$, entonces ξ_β es una secuencia de aceptación de $\mathcal{B}$ para $\beta' = \beta(n)\beta(n+1)\beta(n+2)\ldots$, por lo tanto, $\beta' \in L$. Ya que $\beta = \beta(0,n)\beta'$, entonces $\beta \in U \cdot L$

Por lo tanto $U \cdot L = L(\mathcal{C})$ y $U \cdot L$ es ω-regular.

□

Lema 3.13. *Si $L_1, L_2 \subseteq \Sigma^\omega$ son ω-regulares, entonces $L_1 \cup L_2$ es ω-regular.*

Demostración. Como L_1, L_2 son ω-regulares, existe un autómata $\mathcal{B} = (Q_{\mathcal{B}}, \Sigma, q_{\mathcal{B}}, \Delta_{\mathcal{B}}, F_{\mathcal{B}})$ que reconoce a L_1 y un autómata $\mathcal{C} = (Q_{\mathcal{C}}, \Sigma, q_{\mathcal{C}}, \Delta_{\mathcal{C}}, F_{\mathcal{C}})$ que reconoce a L_2. Y además podemos suponer que $Q_{\mathcal{B}} \cap Q_{\mathcal{C}} = \emptyset$ y por lo tanto $\Delta_{\mathcal{B}} \cap \Delta_{\mathcal{C}} = \emptyset$.

Construimos el ABN $\mathcal{A}$ de tal forma que $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$, con:

- $Q = Q_{\mathcal{B}} \cup Q_{\mathcal{C}} \cup \{q_0\}$.
- $\Delta = \Delta_{\mathcal{B}} \cup \Delta_{\mathcal{C}} \cup \bigcup_{q \in Q_{\mathcal{B}}} \{(q_0, a, q) \mid (q_{\mathcal{B}}, a, q) \in \Delta_{\mathcal{B}}\}) \cup \bigcup_{q \in Q_{\mathcal{C}}} \{(q_0, a, q) \mid (q_{\mathcal{C}}, a, q) \in \Delta_{\mathcal{C}}\})$.
- $F = F_{\mathcal{B}} \cup F_{\mathcal{C}}$.

Sea $\alpha \in L_1 \cup L_2$, si $\alpha \in L_1$, entonces existe una ejecución de aceptación ρ_α de $\mathcal{B}$ para α tal que $\rho_\alpha = q_{\mathcal{B}}\rho_1\rho_2...$ con $infty(\rho_\alpha) \cap F_{\mathcal{B}} \neq \emptyset$. Por la definición de Δ, la ejecución ξ_α de $\mathcal{A}$ sobre α es de la forma $\xi_\alpha = q_0\rho_1\rho_2...$, además $F_{\mathcal{B}} \subseteq F$, por lo que $infty(\xi_\alpha) \cap F \neq \emptyset$. Por lo tanto, ξ_α es una ejecución de aceptacion y por lo tanto $\alpha \in L(\mathcal{A})$. De la misma manera, si $\alpha \in L_2$, entonces $\alpha \in L(\mathcal{A})$.

Sea $\alpha \in L(\mathcal{A})$, entonces existe una ejecución de aceptación ρ_α de $\mathcal{A}$ sobre α tal que $\rho_\alpha = q_0\rho_1...$ y $infty(\rho_\alpha) \cap F \neq \emptyset$. Notemos que por la definición de Δ y dado que $Q_{\mathcal{B}} \cap Q_{\mathcal{C}} = \emptyset$ y $\Delta_{\mathcal{B}} \cap \Delta_{\mathcal{C}} = \emptyset$, si $\rho_1 \in Q_{\mathcal{B}}$, entonces $\forall i > 0, \rho_i \in Q_{\mathcal{B}}$ y además la ejecución ξ_α de $\mathcal{B}$ sobre α es de la forma $\xi_\alpha = q_{\mathcal{B}}\rho_1\rho_2...$, por lo que $infty(\xi_\alpha) \cap F_{\mathcal{B}} \neq \emptyset$. Entonces ξ_α es una ejecución de aceptación de $\mathcal{B}$ sobre α y por lo tanto $\alpha \in L_1$. De la misma forma, si $\rho_1 \in Q_{\mathcal{C}}$, entonces $\alpha \in L_2$ y por lo tanto $\alpha \in L_1 \cup L_2$.

Por lo tanto $L_1 \cup L_2 = L(\mathcal{B})$ y $L_1 \cup L_2$ es ω-regular.

□

La siguiente propiedad de cerradura es con respecto a la intersección. Observese que esta propiedad no se puede demostrar apelando a la unión y al complemento, como en el caso de autómatas finitos, puesto que la construcción de un autómata de Büchi para el complemento de un lenguaje es complicada y no se ha discutido aún. En su lugar, es fácil adaptar la construcción del autómata producto del caso finito a los autómatas de Büchi.

Lema 3.14. *Si $L_1, L_2 \subseteq \Sigma^\omega$ son ω-regulares, entonces $L_1 \cap L_2$ es ω-regular.*

Demostración. Como L_1, L_2 son ω-regulares, existe un autómata $\mathcal{B} = (Q_\mathcal{B}, \Sigma, q_\mathcal{B}, \Delta_\mathcal{B}, F_\mathcal{B})$ que reconoce a L_1 y un autómata $\mathcal{C} = (Q_\mathcal{C}, \Sigma, q_\mathcal{C}, \Delta_\mathcal{C}, F_\mathcal{C})$ que reconoce a L_2. Y además podemos suponer que $Q_\mathcal{B} \cap Q_\mathcal{C} = \emptyset$ y por lo tanto $\Delta_\mathcal{B} \cap \Delta_\mathcal{C} = \emptyset$.

Construimos el ABN $\mathcal{A}$ de tal forma que $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ con:

- $Q = Q_\mathcal{B} \times Q_\mathcal{C} \times \{b_0, b_1, b_2\}$, donde $b_0, b_1, b_2 \notin Q_\mathcal{B} \cup Q_\mathcal{C}$.
- $q_0 = (q_\mathcal{B}, q_\mathcal{C}, b_0)$.
- $F = Q_\mathcal{B} \times Q_\mathcal{C} \times \{b_2\}$.

Definimos Δ como sigue:
Si $(q, a, q') \in \Delta_\mathcal{B}$ y $(r, a, r') \in \Delta_\mathcal{C}$

- $((q, r, b_0), a, (q', r', b_1)) \in \Delta$ si $q' \in F_\mathcal{B}$
- $((q, r, b_1), a, (q', r', b_2)) \in \Delta$ si $r' \in F_\mathcal{C}$
- $((q, r, b_2), a, (q', r', b_0)) \in \Delta$
- $((q, r, b_0), a, (q', r', b_0)) \in \Delta$
- $((q, r, b_1), a, (q', r', b_1)) \in \Delta$

Sea $\alpha \in L_1 \cap L_2$, entonces existe una ejecución de aceptación ρ_α de $\mathcal{B}$ para α tal que $\rho_\alpha = q_\mathcal{B}\rho_1\rho_2...$ con $infty(\rho_\alpha) \cap F_\mathcal{B} \neq \emptyset$ y una ejecución de aceptación ξ_α de $\mathcal{C}$ para α tal que $\xi_\alpha = q_\mathcal{C}\xi_1\xi_2...$ con $infty(\xi_\alpha) \cap F_\mathcal{C} \neq \emptyset$. Por definición de Δ, $((q_\mathcal{B}, q_\mathcal{C}, b_0), \alpha(0), (\rho_1, \xi_1, b_0)) \in \Delta$.

Como ρ_α, ξ_α son secuencias infinitas podemos suponer, sin pérdida de generalidad, que existen $0 < j < k$ tales que $\rho_j \in \mathit{infty}(\rho_\alpha) \cap F_\mathcal{B}$, $\xi_k \in \mathit{infty}(\xi_\alpha) \cap F_\mathcal{C}$ y además $\exists^\omega l, m\ (l < m \wedge \rho_j = \rho_l \wedge \xi_k = \xi_m)$.

Con lo anterior, tenemos los siguientes casos:

1. Si $0 < i < j$, entonces $((\rho_i, \xi_i, b_0), \alpha(i), (\rho_{i+1}, \xi_{i+1}, b_0)) \in \Delta$.
2. Si $i = j - 1$, entonces $((\rho_i, \xi_i, b_0), \alpha(i), (\rho_j, \xi_j, b_1)) \in \Delta$.
3. Si $j \leq i < k$, entonces $((\rho_i, \xi_i, b_1), \alpha(i), (\rho_{i+1}, \xi_{i+1}, b_1)) \in \Delta$.
4. Si $i = k - 1$, entonces $((\rho_i, \xi_i, b_1), \alpha(i), (\rho_k, \xi_k, b_2)) \in \Delta$.
5. $((\rho_k, \xi_k, b_2), \alpha(k), (\rho_{k+1}, \xi_{k+1}, b_0)) \in \Delta$.

Notemos que a partir del caso 5, al llegar a ρ_{k+1} estamos en un estado de la forma (ρ_i, ξ_i, b_0) por lo que volvemos a estar en el caso 1. Por los casos 1 a 5 y dadas l, m de la forma antes descritas, pasamos por un número infinito de estados de la forma (ρ_l, ξ_l, b_1) y (ρ_m, ξ_m, b_2), además de que $(\rho_m, \xi_m, b_2) \in F$, por lo tanto tenemos que $(\rho_m, \xi_m, b_2) \in \mathit{infty}(\varrho_\alpha) \cap F$ y por tanto $\mathit{infty}(\varrho_\alpha) \cap F \neq \emptyset$.

Por lo anterior, la ejecución ϱ_α de $\mathcal{A}$ sobre α es tal que $\varrho_\alpha = (q_\mathcal{B}, q_\mathcal{C}, b_0)$ $(\rho_1, \xi_1, b_0) \ldots\ (\rho_j, \xi_j, b_1)(\rho_{j+1}, \xi_{j+1}, b_1) \ldots\ (\rho_k, \xi_k, b_2)(\rho_{k+1}, \xi_{k+1}, b_0) \ldots$. Dado que $\mathit{infty}(\varrho_\alpha) \cap F \neq \emptyset$, ϱ_α es una ejecución de aceptación, por lo que $\alpha \in L(\mathcal{A})$.

Por otro lado, sea $\alpha \in L(\mathcal{A})$, entonces existe una ejecución de aceptación $\rho_\alpha = \rho_0 \rho_1 \ldots$ de $\mathcal{A}$ para α, con cada ρ_i de la forma $\rho_i = (p, r, b_k)$, con $p \in Q_\mathcal{B}$, $r \in Q_\mathcal{C}$, $k \in \{0, 1, 2\}$. Por definición de Δ, para todas $p, p' \in Q_\mathcal{B}, r, r' \in Q_\mathcal{C}, i \in \mathbb{N}$, cada transición $((p, r, b_j), \alpha(i), (p', r', b_k)) \in \Delta$ si y sólo si $(p, \alpha(i), p') \in \Delta_\mathcal{B}$ y $(r, \alpha(i), r') \in \Delta_\mathcal{C}$.

Sean $\rho_\mathcal{B} = \rho_0^\mathcal{B} \rho_1^\mathcal{B} \ldots$ y $\rho_\mathcal{C} = \rho_0^\mathcal{C} \rho_1^\mathcal{C} \ldots$ las ejecuciones de $\mathcal{B}$ y $\mathcal{C}$ para α, respectivamente, de tal forma que $\rho_i \in \Delta$ si y sólo si $\rho_i^\mathcal{B} \in \Delta_\mathcal{B}$ y $\rho_i^\mathcal{C} \in \Delta_\mathcal{C}$.

Por definición de F, existe un estado $q_f \in F$ de la forma $q_f = (p, r, b_2)$ que es visitado una infinidad de veces en ρ_α. Por definición de Δ, sólo podemos llegar a q_f si $r \in F_\mathcal{C}$ y se visitó un estado q_{b_1} de la forma $q_{b_1} = (p', r', b_1)$. Por la construcción de $\rho_\mathcal{C}$ y como q_f es visitado una infinidad de veces, por lo tanto, $\mathit{infty}(\rho_\mathcal{C}) \cap F_\mathcal{C} \neq \emptyset$, es decir, $\rho_\mathcal{C}$ es una ejecución de aceptación de $\mathcal{C}$ para α y por lo tanto, $\alpha \in L_2$. De la misma manera, sólo podemos llegar a q_{b_1} si $p' \in F_\mathcal{B}$ y se visitó un estado q_{b_0} de la forma $q_{b_0} = (p'', r'', b_0)$. Por

la construcción de $\rho_{\mathcal{B}}$ y como q_{b_1} es visitado una infinidad de veces, por lo tanto, $infty(\rho_{\mathcal{B}}) \cap F_{\mathcal{B}} \neq \emptyset$, es decir, $\rho_{\mathcal{B}}$ es una ejecución de aceptación de $\mathcal{B}$ para α y por lo tanto, $\alpha \in L_1$. Por lo anterior, $\alpha \in L_1 \cap L_2$.

Por lo tanto $L_1 \cap L_2 = L(\mathcal{A})$ y $L_1 \cap L_2$ es ω-regular.

□

Como ya se mencionó, la propiedad de cerradura bajo complemento requiere de un mayor análisis debido a la complejidad que involucra el trabajar con palabras infinitas. La siguiente sección está dedicada a la demostración de dicha propiedad.

3.3. Cerradura bajo complemento

Hasta ahora, demostrar cada una las propiedades de cerradura se han reducido a construir el correspondiente autómata de Büchi. Sin embargo, la cerradura bajo complemento no puede ser dada en los mismos términos y es aquí precisamente donde radica la dificultad de la demostración del teorema de Büchi sobre ω-palabras. Para demostrar esta propiedad necesitamos profundizar un poco más en el comportamiento de una ejecución en un autómata de Büchi.

En lo que sigue, $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ es un ABN, $w \in \Sigma^*$ y $\alpha \in \Sigma^\omega$.

Definición 3.15. *Denotamos con $q \xrightarrow{w} q'$ al hecho de que exista una secuencia $p_0, ..., p_n \in Q$ tal que $q = p_0, q' = p_n$ y $(p_i, w(i), p_{i+1}) \in \Delta$ para $0 \leq i < n$.*

Observese que $q \xrightarrow{w} q'$ sucede si y sólo si la ejecución de la palabra finita w, a partir de q, termina en q'.

Definición 3.16. *Denotamos con $W_{\mathcal{A}(qq')}$ al conjunto de palabras finitas dadas por las secuencias de estados que van de q a q', es decir:*

$$W_{\mathcal{A}(qq')} = \{w \in \Sigma^* \mid q \xrightarrow{w} q'\}$$

Es claro que $W_{\mathcal{A}(qq')}$ es regular. La siguiente propiedad de este conjunto será de utilidad más adelante.

Lema 3.15. *Para todo estado $q \in Q$, $W_{\mathcal{A}(qq)} \cdot W_{\mathcal{A}(qq)} \subseteq W_{\mathcal{A}(qq)}$.*

Demostración. Si $w \in W_{\mathcal{A}(qq)} \cdot W_{\mathcal{A}(qq)}$, entonces $w = xy$ con $x, y \in W_{\mathcal{A}(qq)}$, por lo que $q \xrightarrow{x} q \xrightarrow{y} q$, lo cual significa que $q \xrightarrow{w} q$ y por lo tanto $w \in W_{\mathcal{A}(qq)}$. □

Las siguientes dos definiciones son análogas a las anteriores agregando la condición de que se visite un estado final.

Definición 3.17. *Denotamos con* $q \xrightarrow[F]{w} q'$ *si hay una secuencia* $p_0, ..., p_n \in Q$ *tal que* $q = p_0, q' = p_n$ *y* $(p_i, w(i), p_{i+1}) \in \Delta$ *para* $0 \leq i < n$ *y además* $p_i \in F$ *para alguna* i.

Definición 3.18. *Denotamos con* $W^F_{\mathcal{A}(qq')}$, *al conjunto de palabras finitas dadas por las secuencias de estados que van de* q *a* q' *pasando por algún estado final, es decir:*

$$W^F_{\mathcal{A}(qq')} = \{w \in \Sigma^* \mid q \xrightarrow[F]{w} q'\}$$

Nuevamente, es claro que $W^F_{\mathcal{A}(qq')}$ es regular.

Con esto, podemos caracterizar a la clase de los lenguajes ω-regulares como uniones finitas de lenguajes que son concatenación de un regular con la ω-potencia de otro regular.

Teorema 3.5. *Un lenguaje* $L \subseteq \Sigma^\omega$ *es* ω*-regular si y sólo si* L *es la unión finita de lenguajes* $U_i \cdot V_i^\omega$ *donde* $U_i, V_i \subseteq \Sigma^*$ *son regulares, es decir:*

$$L = \bigcup_{i=0}^{n} U_i \cdot (V_i)^\omega$$

más aún, $V_i \cdot V_i \subseteq V_i$.

Demostración.
($\Rightarrow$) Como L es ω-regular, existe un autómata $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ de Büchi que reconoce a L. Observese que por la definición de aceptación de un ABN, el ω-lenguaje reconocido por $\mathcal{A}$ es:

$$L = L(\mathcal{A}) = \bigcup_{q \in F} W_{\mathcal{A}(q_0 q)} \cdot (W_{\mathcal{A}(qq)})^\omega$$

Además, por el lema (3.15), $W_{\mathcal{A}(qq)} \cdot W_{\mathcal{A}(qq)} \subseteq W_{\mathcal{A}(qq)}$.

($\Leftarrow$) Sean $U_i, V_i \subseteq \Sigma^*$ lenguajes regulares con $i \leq n$ para alguna $n \in \mathbb{N}$.

Por los lemas (3.13), (3.12) y (3.11), $L = \bigcup_{i=0}^{n} U_i \cdot (V_i)^\omega$ es ω-regular.

□

El siguiente corolario es de gran importancia.

Corolario 3.2. *Dado un ABN $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$, podemos determinar si $L(\mathcal{A}) \neq \emptyset$.*

Demostración. Puesto que el problema de verificar si un lenguaje regular es vacío, es decidible y dado que, por el teorema anterior, $L(\mathcal{A}) = \bigcup_{i=0}^{n} U_i \cdot (V_i)^\omega$, con U_i y V_i regulares, el problema se reduce a verificar si V_i^ω es no vacío. Esto es inmediato puesto que V_i es regular y $V_i = \emptyset$ si y sólo si $V_i^\omega = \emptyset$.

□

Lo que haremos a continuación será caracterizar a los ω-lenguajes a través de clases de congruencia y utilizar algunas propiedades que resulten de esta caracterización. Esto cobra sentido si pensamos en el teorema de Myhill-Nerode y los lenguajes regulares.

Definición 3.19. *Sean $u, v \in \Sigma^*$. Definimos la relación $\equiv_\mathcal{A}$ sobre Σ^* como:*

$$u \equiv_\mathcal{A} v \text{ si y sólo si}$$
$$\forall p, q \in Q(u \in W_{\mathcal{A}(pq)} \Leftrightarrow v \in W_{\mathcal{A}(pq)} \wedge u \in W^F_{\mathcal{A}(pq)} \Leftrightarrow v \in W^F_{\mathcal{A}(pq)})$$

es decir, para cualesquiera p, q estados en Q, u nos lleva de p a q si y sólo si v nos lleva de p a q y además con u pasamos por un estado final si y sólo si también lo hacemos con v.

La siguiente propiedad de la relación de equivalencia $\equiv_\mathcal{A}$ es relevante.

Lema 3.16. *La relación $\equiv_\mathcal{A}$ es una congruencia de índice finito sobre Σ^*.*

Demostración. Es claro que la relación $\equiv_\mathcal{A}$ es de equivalencia ya que está definida directamente en términos de equivalencias lógicas y es de índice finito porque Q es finito y cada una de sus clases está determinada por un par de elementos de Q. Veamos ahora que $\equiv_\mathcal{A}$ es una congruencia.

Sean $u, v, x \in \Sigma^*$ tales que $u \equiv_\mathcal{A} v$. Supongamos que $u, v \in W_{\mathcal{A}(pq)}$ para algunos $p, q \in Q$. Si $\exists r \in Q\ (x \in W_{\mathcal{A}(qr)})$, entonces $ux \in W_{\mathcal{A}(pr)} \wedge vx \in W_{\mathcal{A}(pr)}$.

Si $\forall r \in Q\,(x \notin W_{\mathcal{A}(qr)})$, entonces $ux \notin W_{\mathcal{A}(pr)} \wedge vx \notin W_{\mathcal{A}(pr)}$. Por lo que ambos casos permiten concluir que $\forall x \in \Sigma^*\ (ux \in W_{\mathcal{A}(pr)} \Leftrightarrow vx \in W_{\mathcal{A}(pr)})$. De manera similar podemos mostrar que $\forall x \in \Sigma^*\ (ux \in W^F_{\mathcal{A}(pr)} \Leftrightarrow vx \in W^F_{\mathcal{A}(pr)})$. Por lo tanto $ux \equiv_{\mathcal{A}} vx$, es decir, $\equiv_{\mathcal{A}}$ es invariante por la derecha.

Siguiendo el mismo razonamiento podemos ver que $xu \equiv_{\mathcal{A}} xv$ y por lo tanto $\equiv_{\mathcal{A}}$ es invariante por la izquierda.

□

El siguiente concepto es de primordial importancia para demostrar la cerradura de los lenguajes ω-regulares bajo complemento.

Definición 3.20. *Sea $\equiv$ una congruencia sobre Σ^*. Decimos que $\equiv$ satura a un ω-lenguaje $L \subseteq \Sigma^\omega$ si, para todas las clases U, V de $\equiv$, $U \cdot V^\omega \cap L \neq \emptyset$ implica que $U \cdot V^\omega \subseteq L$.*

A continuación mostramos que la recién definida congruencia $\equiv_{\mathcal{A}}$, satura tanto a $L(\mathcal{A})$ como a su complemento.

Lema 3.17. *La congruencia $\equiv_{\mathcal{A}}$ satura a $L(\mathcal{A})$.*

Demostración. Sea $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ un ABN, sean U, V clases de congruencia de $\equiv_{\mathcal{A}}$ tales que $U \cdot V^\omega \cap L(\mathcal{A}) \neq \emptyset$ y sea $\alpha \in U \cdot V^\omega \cap L(\mathcal{A})$. Como $\alpha \in U \cdot V^\omega$, $\alpha = uv_1v_2...$ con $u \in U$, $v_i \in V$. Como $\alpha \in L(\mathcal{A})$, existe una ejecución de aceptación ρ_α de $\mathcal{A}$ para α, a partir de la cual podemos obtener estados $q_0q_1q_2...$ de modo que $q_0 \xrightarrow{u} q_1 \xrightarrow{v_1} q_2 \xrightarrow{v_2} ...$ y además $q_i \xrightarrow[F]{v_i} q_{i+1}$ para una infinidad de índices i.

Sea $\beta \in U \cdot V^\omega$ tal que $\beta = u'v'_1v'_2...$ con $u' \in U$, $v'_i \in V$, $i > 0$. Como U, V son clases de $\equiv_{\mathcal{A}}$, entonces $u \equiv_{\mathcal{A}} u'$ y para $i > 0$ $v_i \equiv_{\mathcal{A}} v'_i$ por lo que $q_0 \xrightarrow{u'} q_1 \xrightarrow{v'_1} q_2 \xrightarrow{v'_2} ...$ y además $q_i \xrightarrow[F]{v'_i} q_{i+1}$ para una infinidad de índices i. Esta secuencia de estados atestigua que $\beta \in L(\mathcal{A})$ y como $\beta \in U \cdot V^\omega$ es arbitraria, entonces $U \cdot V^\omega \subseteq L(\mathcal{A})$ y por lo tanto $\equiv_{\mathcal{A}}$ satura a $L(\mathcal{A})$.

□

Lema 3.18. *La congruencia $\equiv_{\mathcal{A}}$ satura a $\Sigma^\omega \setminus L(\mathcal{A})$.*

Demostración. Sea $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ un ABN, sean U, V clases de congruencia de $\equiv_{\mathcal{A}}$ tales que $U \cdot V^\omega \cap (\Sigma^\omega \setminus L(\mathcal{A})) \neq \emptyset$ y sea $\alpha \in U \cdot V^\omega \cap (\Sigma^\omega \setminus L(\mathcal{A}))$.

Como $U \cdot V^\omega \cap (\Sigma^\omega \setminus L(\mathcal{A})) \neq \emptyset$, entonces $U \cdot V^\omega \not\subseteq L(\mathcal{A})$ y por el lema (3.17), $U \cdot V^\omega \cap L(\mathcal{A}) = \emptyset$, por lo que $U \cdot V^\omega \subseteq \Sigma^\omega \setminus L(\mathcal{A})$ y por lo tanto $\equiv_{\mathcal{A}}$ satura a $\Sigma^\omega \setminus L(\mathcal{A})$.

□

La propiedad de saturación nos asegura que un ω-lenguaje L de la forma $U \cdot V^\omega$, con U, V clases de congruencia inducida por un ABN $\mathcal{A}$, está totalmente contenido en $L(\mathcal{A})$ o en su complemento $\Sigma^\omega \setminus L(\mathcal{A})$.

Hemos podido establecer la forma de las ω-palabras reconocidas por un ABN en términos de las clases de una congruencia inducida por el mismo. Ahora necesitamos hacerlo en términos de las clases de una congruencia arbitraria de índice finito definida sobre Σ^*. Para esto, tendremos que analizar con más detalle la estructura de las palabras.

En lo siguiente se considera que $\equiv$ es una congruencia de índice finito sobre Σ^*, $\alpha \in \Sigma^\omega$ y $k, k', m \in \mathbb{N}$, con $m > k, m > k'$.

Definición 3.21. *Decimos que dos posiciones k, k' de α se unen en la posición m si $\alpha(k, m) \equiv \alpha(k', m)$ y lo denotamos con $k \overset{\alpha}{\underset{m}{\sim}} k'$.*

Lema 3.19. $\overset{\alpha}{\underset{m}{\sim}}$ *es una relación de equivalencia de índice finito sobre* $\mathbb{N}$.

Demostración. Es claro que $\overset{\alpha}{\underset{m}{\sim}}$ es una relación de equivalencia de índice finito sobre $\mathbb{N}$ porque se define directamente de $\equiv$, que es una congruencia de índice finito.

□

Lema 3.20. *Si $k \overset{\alpha}{\underset{m}{\sim}} k'$ y $m' > m$, entonces $k \overset{\alpha}{\underset{m'}{\sim}} k'$.*

Demostración. Como $k \overset{\alpha}{\underset{m}{\sim}} k'$, entonces $\alpha(k, m) \equiv \alpha(k', m)$ y como $\equiv$ es una congruencia, $\alpha(k, m)\alpha(m, m') \equiv \alpha(k', m)\alpha(m, m')$, de donde se sigue que $\alpha(k, m') \equiv \alpha(k', m')$, es decir, $k \overset{\alpha}{\underset{m'}{\sim}} k'$. □

Definición 3.22. *La relación $\overset{\alpha}{\sim}$ sobre $\mathbb{N}$ se define como:*

$$k \overset{\alpha}{\sim} k' \text{ si } \exists m \in \mathbb{N},\ k \overset{\alpha}{\underset{m}{\sim}} k'$$

Lema 3.21. $\overset{\alpha}{\sim}$ *es una relación de equivalencia de índice finito.*

Demostración. Es claro que la relación $\overset{\alpha}{\sim}$ es reflexiva y simétrica. Veamos que es transitiva, sean $k, k', k'' \in \mathbb{N}$ tales que $k \overset{\alpha}{\sim} k'$ y $k' \overset{\alpha}{\sim} k''$, esto es $k \underset{m}{\overset{\alpha}{\sim}} k'$ y $k' \underset{n}{\overset{\alpha}{\sim}} k''$ para algunas $m, n \in \mathbb{N}$. Tomamos $r = max(m, n)$, de donde, por el lema (3.20), se sigue que $k \underset{r}{\overset{\alpha}{\sim}} k'$ y $k' \underset{r}{\overset{\alpha}{\sim}} k''$ lo cual implica que $k \underset{r}{\overset{\alpha}{\sim}} k''$ y finalmente $k \overset{\alpha}{\sim} k''$.

Notemos que cada clase $[k]_{\overset{\alpha}{\sim}}$ coincide con una clase de equivalencia $[k]_{\underset{m}{\overset{\alpha}{\sim}}}$, por lo que $\overset{\alpha}{\sim}$ también es de índice finito.

□

A continuación demostramos unas propiedades importantes de la relación $\overset{\alpha}{\sim}$.

Lema 3.22. *Dada $\overset{\alpha}{\sim}$ sobre $\mathbb{N}$, existen $k_0, k_1, k_2...$ posiciones tales que:*

1. $k_0 < k_1 < k_2....$
2. $k_0 > 0$.
3. *Existe una clase de $\overset{\alpha}{\sim}$ a la que pretenecen todos los k_i.*
4. *Para toda $i > 0$ los segmentos $\alpha(k_0, k_i)$ pertenecen a una misma clase de $\equiv$.*
5. $\forall i > 0$ $(k_0 \underset{k_{i+1}}{\overset{\alpha}{\sim}} k_i)$, *es decir,* $\alpha(k_0, k_{i+1}) \equiv \alpha(k_i, k_{i+1})$.

Demostración. Tómese una secuencia P arbitraria creciente de naturales sin el cero (cumpliendo 1 y 2). Como $\overset{\alpha}{\sim}$ es de índice finito y P es infinita, podemos construir una subsecuencia infinita $P' = k_0, k_1, k_2...$ que cumpla (3).

Por ser P' infinita, podemos construir una subsecuencia P'' de P', con $k_0 \in P''$, de manera que para cada $i > 0$, si k_i cumple (4) la agregamos a P'' y si no, la descartamos, renombrando los elementos de P'' obtenemos la subsecuencia infinita $P'' = k_0, k_1, k_2...$ tal que cada k_i cumple (4).

Como $k_0, k_1, ...$ pertenecen a la misma clase de $\overset{\alpha}{\sim}$ (por 3), entonces para cada $i > 0$ $\exists m_i > k_i$ tal que $k_0...k_i$ se unen en m_i. Volvemos a construir una subsecuencia P''' de P'', con $k_0 \in P'''$, de la siguiente manera:

Para cada $k_i \in P''$ con $i > 0$, tomamos la primera $k_j \in P''$ tal que $j > i$ y $k_j \geq m_i$, por el lema (3.20) tenemos que $k_0...k_i$ se unen en k_j.

Descartamos todas las k_s con $i < s < j$ (si hay alguna) y renombramos k_j a k_{i+1} agregandola a P'''.

Por la construcción de P''', para $i > 0$ y $k_i \in P'''$ se tiene que $k_0...k_i$ se unen en k_{i+1} cumpliendo (5).

□

El lema anterior nos permitirá demostrar que cualquier ω-palabra está determinada por un par de clases de alguna congruencia sobre Σ^*, en el sentido del siguiente:

Lema 3.23. *Sea $\equiv$ una congruencia de índice finito sobre Σ^*. Para cualquier palabra $\alpha \in \Sigma^\omega$, existen U, V clases de congruencia de $\equiv$, tales que $\alpha \in U \cdot V^\omega$ y además $V \cdot V \subseteq V$.*

Demostración. Sea $k_0, k_1, k_2...$ una secuencia infinita de posiciones que cumplen las condiciones del lema (3.22).

Considerese la descomposición $\alpha = \alpha(0, k_0)\alpha(k_0, k_1)\alpha(k_1, k_2)...$ y sean $U = [\alpha(0, k_0)]_\equiv$ y V la clase de $\equiv$ a la que pertenecen los segmentos $\alpha(k_0, k_i)$. De esta manera se tiene que $\forall i > 0\ \alpha(k_i, k_{i+1}) \in V$ y por lo tanto $\alpha \in U \cdot V^\omega$.

Por otra parte, para toda $i > 0\ \alpha(k_0, k_{i+1}) = \alpha(k_0, k_i)\alpha(k_i, k_{i+1})$ con $\alpha(k_0, k_{i+1}) \in V$ y $\alpha(k_0, k_i)\alpha(k_i, k_{i+1}) \in V \cdot V$, por lo que $V \cdot V \cap V \neq \emptyset$.

Ya que $V \cdot V \cap V \neq \emptyset$ tomemos $w \in V \cdot V \cap V$, como $w \in V \cdot V$ entonces $w = w_1 \cdot w_2$ con $w_1, w_2 \in V$.

Veamos ahora que $V \cdot V \subseteq V$. Sea $x \in V \cdot V$, por lo que $x = x_1 \cdot x_2$ con $x_1, x_2 \in V$. Como V es una clase de congruencia y $x_1, x_2, w_1, w_2 \in V$, entonces $x_1 \cdot x_2 \equiv w_1 \cdot w_2$, es decir, $x \equiv w$ y además $w \in V$, por lo tanto $x \in V$, lo cual demuestra que $V \cdot V \subseteq V$.

□

El siguiente lema muestra como una congruencia que satura a un lenguaje dado permite descomponerlo mediante las clases de dicha congruencia.

Lema 3.24. *Si $\equiv$ es una congruencia de índice finito sobre Σ^* y además satura a $L \subseteq \Sigma^\omega$, entonces L es ω-regular y*

$$L = \bigcup \{U \cdot V^\omega \mid U, V \text{ clases de } \equiv, U \cdot V^\omega \cap L \neq \emptyset\}$$

Demostración. Sean $\equiv$ una congruencia de índice finito sobre Σ^* y $L \subseteq \Sigma^\omega$ tal que $\equiv$ satura a L.

($\supseteq$) Es la definición de la propiedad de saturación.

($\subseteq$) Si $\alpha \in L$, por el lema (3.23), existen U, V clases de congruencia de $\equiv$ tales que $\alpha \in U \cdot V^\omega$.

Como $\equiv$ es una congruencia de índice finito, por el lema (1.4), cada clase de $\equiv$ es regular y como L es la unión finita de algunas de ellas, por el teorema (3.5), L es ω-regular.

□

Con todo lo anterior tenemos la base para poder demostrar la cerradura bajo complemento de los lenguajes ω-regulares y por lo tanto de los ABN.

Teorema 3.6. *Si $L \subseteq \Sigma^\omega$ es ω-regular, entonces $\Sigma^\omega \setminus L$ es ω-regular. Más aún, se puede construir un ABN que reconozca a $\Sigma^\omega \setminus L$.*

Demostración. Sea $L \subseteq \Sigma^\omega$ un lenguaje ω-regular. Por lo que existe un ABN $\mathcal{A}$ que reconoce a L.

Por el lema (3.18), $\equiv_{\mathcal{A}}$ satura a $\Sigma^\omega \setminus L$. Por el lema (3.16), sabemos que $\equiv_{\mathcal{A}}$ es una congruencia de índice finito y por el lema (3.24), $\Sigma^\omega \setminus L$ es ω-regular.

Más aún, por el teorema (3.5), $\Sigma^\omega \setminus L = \bigcup_{i=0}^{n} U_i \cdot (V_i)^\omega$ donde $U_i, V_i \subseteq \Sigma^*$ son regulares y por lo tanto, de la misma forma que demostramos los lemas (3.13), (3.12) y (3.11), podemos construir un ABN que reconozca $\Sigma^\omega \setminus L$.

□

Con esto, podemos ver que los autómatas de Büchi y los lenguajes ω-regulares son una generalización, sobre palabras infinitas, de los autómatas finitos y los lenguajes regulares, respectivamente.

La complejidad que se deriva del procesamiento de palabras infinitas no permitió que la cerradura se demostrara de manera análoga a los autómatas finitos, lo que nos obligó a profundizar en el análisis con algunos conceptos algebraicos como las relaciones de equivalencia sobre Σ^*. Así, pudimos caracterizar a cada ω-lenguaje y su complemento, a través de las clases de equivalencia de dichas relaciones y verificar que si un lenguaje es ω-regular, también lo es su complemento.

Cabe destacar que la propiedad de saturación de una relación de equivalencia de índice finito ha resultado de gran importancia en la demostración de la cerradura bajo complemento de los lenguajes ω-regulares.

Lo siguiente es verificar, análogamente al caso finito, que los lenguajes son ω-regulares si y sólo si son S1S-definibles.

3.4. Teorema de Büchi

Ya hemos demostrado la equivalencia entre los lenguajes regulares y los lenguajes definibles en la lógica S1S. Ahora lo haremos para una generalización sobre palabras infinitas siguiendo el mismo procedimiento. En resumen:

1. De ABN a lógica. Construiremos un enunciado $\varphi_{\mathcal{A}}$ en S1S a partir de un ABN $\mathcal{A}$ dado y verificaremos que $L(\mathcal{A}) = L(\varphi_{\mathcal{A}})$.

2. De lógica a ABN. Construiremos un ABN $\mathcal{A}_\psi$ a partir de un enunciado ψ en S1S0 equivalente a un enunciado φ en S1S y verificaremos que $L(\varphi) = L(\mathcal{A}_\psi)$.

Necesitamos ajustar algunas definiciones para S1S pensando en que ahora nuestras interpretaciones serán sobre $\mathbb{N}$.

Antes definiremos un par de fórmulas auxiliares sobre S1S que nos permitirán expresar la propiedad de orden.

$$\begin{aligned} ClS(X) &\equiv_{def} \quad \forall z \forall v((X(z) \wedge S(z,v)) \rightarrow X(v)) \\ x < y &\equiv_{def} \quad \exists X(ClS(X) \wedge \neg X(x) \wedge X(y)) \end{aligned}$$

donde $x, y, z, v \in V_1$ y $X \in V_2$.

Con la fórmula $ClS(X)$ expresamos que un conjunto X es cerrado bajo la relación sucesor.

Con la fórmula $x < y$ expresamos que existe un conjunto X cerrado bajo sucesor que contiene a y y no contiene a x. Esto es, X se interpreta como un intervalo de la forma $[k, \infty]$ que contiene a $\sigma(y)$ y no contiene a $\sigma(x)$, para algún estado de variables σ y alguna $k \in \mathbb{N}$.

La fórmula $x < y$, así definida, realmente captura la noción de orden entre dos números, debido a que las interpretaciones tendrán como universo a $\mathbb{N}$.

Veamos como se define nuestra interpretación sobre palabras infinitas. Las siguientes definiciones son análogas a las utilizadas para las lógicas S0S y S1S.

Definición 3.23. *Sean Σ un alfabeto y $\alpha = a_0a_1...$ en Σ^ω. El modelo $\hat{\alpha} = \langle \mathbb{N}, \mathcal{I} \rangle$ se conoce como el modelo de palabra para α y éste es una interpretación para S1S, donde $\mathcal{I}$ es la función de interpretación definida como:*

- $S^{\mathcal{I}} = \{(i, i+1) \mid 0 \leq i\}$ *es la relación sucesor en* $\mathbb{N}$.
- $Q_a^{\mathcal{I}} = \{i \in \mathbb{N} \mid \alpha(i) = a\}$ *son predicados unarios que determinan el conjunto de posiciones de símbolos de α que son iguales al símbolo a.*

Definición 3.24. *Sea $\alpha \in \Sigma^\omega$. Decimos que α satisface un enunciado φ en S1S si $\hat{\alpha} \models \varphi$.*

Con lo anterior ya podemos definir un lenguaje de palabras infinitas a través de S1S.

Definición 3.25. *El lenguaje de palabras infinitas definido por el enunciado φ en S1S es:*

$$L(\varphi) = \{\alpha \in \Sigma^\omega \mid \hat{\alpha} \models \varphi\}$$

Definición 3.26. *Decimos que un lenguaje de palabras infinitas $L \subseteq \Sigma^\omega$ es S1S-definible si existe un enunciado ψ en S1S tal que $L = L(\psi)$.*

El siguiente lema permitirá simplificar las demostraciones que se deriven de la fórmulas auxiliares $Sing(X)$ y $x < y$ bajo esta nueva interpretación.

Lema 3.25. *Sea $M = \langle \mathbb{N}, \mathcal{I} \rangle$ una interpretación para S1S y sea σ un estado de las variables. Entonces:*

$$\begin{aligned} \mathcal{I}_\sigma(Sing(X)) = 1 &\quad \textit{si y sólo si } \sigma(X) = \{m\} \textit{ para algún } m \in \mathbb{N} \\ \mathcal{I}_\sigma(x < y) = 1 &\quad \textit{si y sólo si } \sigma(x) < \sigma(y) \end{aligned}$$

Demostración.

- $Sing(X)$. La demostración es la misma que para el lema (2.1).
- $x < y$.

  $\mathcal{I}_\sigma(x < y) = 1$

  si y sólo si $\mathcal{I}_\sigma(\ \exists X(ClS(X) \wedge \neg X(x) \wedge X(y))\ ) = 1$

  si y sólo si $\mathcal{I}_{\sigma[X/R]}(\ \forall z \forall v((X(z) \wedge S(z,v)) \rightarrow X(v))\ ) = 1$ y $\mathcal{I}_{\sigma[X/R]}(\ \neg X(x) \wedge X(y)\ ) = 1$ para algún $R \subseteq \mathbb{N}$

$$
\begin{array}{lll}
 & \mathcal{I}_{\sigma[X,z,v/R,q,p]}(\ \neg X(z)\ ) = 1 \text{ o} & \\
 & \mathcal{I}_{\sigma[X,z,v/R,q,p]}(\ \neg S(z,v)\ ) = 1 \text{ o} & \text{para algún } R \subseteq \mathbb{N} \text{ y} \\
\text{si y sólo si} & \mathcal{I}_{\sigma[X,z,v/R,q,p]}(\ X(v)\ ) = 1 \text{ y} & \text{para toda } p \in \mathbb{N} \text{ y} \\
 & \mathcal{I}_{\sigma[X/R]}(\ \neg X(x)\ ) = 1 \text{ y} & \text{para toda } q \in \mathbb{N} \\
 & \mathcal{I}_{\sigma[X/R]}(\ X(y)\ ) = 1 &
\end{array}
$$

$$
\begin{array}{lll}
 & \mathcal{I}_{\sigma[X,z,v/R,q,p]}(\ X(z)\ ) = 0 \text{ o} & \\
 & \mathcal{I}_{\sigma[X,z,v/R,q,p]}(\ S(z,v)\ ) = 0 \text{ o} & \text{para algún } R \subseteq \mathbb{N} \text{ y} \\
\text{si y sólo si} & \mathcal{I}_{\sigma[X,z,v/R,q,p]}(\ X(v)\ ) = 1 \text{ y} & \text{para toda } p \in \mathbb{N} \text{ y} \\
 & \mathcal{I}_{\sigma[X/R]}(\ X(x)\ ) = 0 \text{ y} & \text{para toda } q \in \mathbb{N} \\
 & \mathcal{I}_{\sigma[X/R]}(\ X(y)\ ) = 1 &
\end{array}
$$

$$
\begin{array}{lll}
 & 1.\ q \notin R \text{ o } p \neq q+1 \text{ o } p \in R \text{ y} & \text{para algún } R \subseteq \mathbb{N} \text{ y} \\
\text{si y sólo si} & 2.\ \sigma(x) \notin R \text{ y} & \text{para toda } p \in \mathbb{N} \text{ y} \\
 & 3.\ \sigma(y) \in R & \text{para toda } q \in \mathbb{N}
\end{array}
$$

Basta mostrar que las últimas tres condiciones son equivalentes a que $\sigma(x) < \sigma(y)$. Si estas condiciones se cumplen, entonces no puede suceder que $\sigma(x) \geq \sigma(y)$, puesto que la condición (1) implica que R es cerrado bajo sucesor y como $\sigma(y) \in R$, necesariamente se tendría que $\sigma(x) \in R$, lo cual contradice la condición (2).

En la otra dirección, suponemos que $\sigma(x) < \sigma(y)$ y basta con exhibir $R \subseteq \mathbb{N}$ tal que se cumplan (1), (2) y (3).

Tomemos $R = [\sigma(y), \infty] \subset \mathbb{N}$. Por definición de R, se cumple (3) y puesto que $\sigma(x) < \sigma(y)$, también se cumple (2).

Para probar (1) analizamos dos casos, si $q < \sigma(y)$, entonces $q \notin R$ y se cumple (1).

En otro caso, $q \geq \sigma(y)$ y ahora analizamos otros dos casos, a saber $p = q + 1$ y $p \neq q + 1$. Si $p = q + 1$, $p \in R$, puesto que $p \geq q$ y por lo tanto se cumple (1). Finalmente, si $p \neq q + 1$, se cumple (1).

Notemos que para todo $q \geq \sigma(y)$, $q \in R$, por lo que, si $p = q + 1$, entonces $p \in R$ y se cumple (1). Si $p \neq q+1$, se cumple (1). Si $q < \sigma(y)$, entonces $q \notin R$ y se cumple (1). Por lo tanto (1) se cumple para todo $p, q \in R$.

□

De la misma manera, ajustamos las definiciones, análogas al caso finito, para la lógica S1S0.

Definición 3.27. *Sean Σ un alfabeto y $\alpha = a_0a_1...$ en Σ^ω. El modelo de palabra $\check{\alpha} = \langle \mathbf{n}, \mathcal{I} \rangle$ para α es una interpretación para S1S0, donde $\mathcal{I}$ es la función de interpretación definida como:*

- $S^{\mathcal{I}} = \{(\{i\}, \{i+1\}) \mid 0 \leq i < n-1\}$ *es la relación sucesor para subconjuntos de un sólo elemento en* $\mathbb{N}$.
- $Q_a^{\mathcal{I}} = \{i \in \mathbb{N} \mid \alpha(i) = a\}$ *son conjuntos de posiciones de símbolos de α que son iguales al símbolo a.*
- $\subseteq^{\mathcal{I}} = \{(N, K) \mid N, K \subseteq \mathbb{N},\ N \subseteq K\}$ *es la relación usual de contención entre conjuntos.*

Observese, nuevamente, que la única diferencia, entre el modelo de palabra $\check{\alpha}$ para S1S0 y el modelo $\hat{\alpha}$ para S1S (3.23), está en la función de interpretación.

Definición 3.28. *Sea $\alpha \in \Sigma^\omega$. Decimos que α satisface un enunciado φ en S1S0 si $\check{\alpha} \models \varphi$.*

Con lo anterior ya podemos definir un lenguaje de palabras infinitas a través de S1S0.

Definición 3.29. *El lenguaje definido por el enunciado φ en S1S0 es*

$$L(\varphi) = \{\alpha \in \Sigma^\omega \mid \check{\alpha} \models \varphi\}$$

Definición 3.30. *Decimos que un lenguaje $L \subseteq \Sigma^\omega$ es S1S0-definible si existe un enunciado ψ en S1S0 tal que $L = L(\psi)$.*

Verifiquemos que todo lenguaje ω-regular es S1S-definible.

3.4.1. De ABN a lógica

En lo que sigue: $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ es un ABN con $Q = \{0, ..., k\}$, $q_0 = 0$, $\alpha = a_0a_1... \in L(\mathcal{A})$ y ρ_α es una ejecución de aceptación de $\mathcal{A}$ para α.

Ls siguiente definición es analóga a la del caso con autómatas finitos.

Definición 3.31. *Los conjuntos $P_0, ..., P_k \subseteq \mathbb{N}$ son tales que cada P_i contiene las posiciones de α donde el estado i es visitado en la ejecución de aceptación ρ_α. Es decir:*

$$P_i = \{j \mid i = \rho_j,\ j \in \mathbb{N}\}$$

Observese que P_i puede ser vacío si no se pasó por el estado i.

De la misma manera que lo hicimos para autómatas finitos, necesitamos construir un enunciado φ_i para cada una de las cuatro condiciones necesarias de una secuencia de aceptación de $\mathcal{A}$ para cada palabra de $L(\mathcal{A})$.

Definición 3.32. *Sean $X_0, ..., X_k$ variables de S1S. Definimos las fórmulas $\varphi_1, \varphi_2, \varphi_3$ y φ_4 de la siguiente manera:*

$$\varphi_1 \equiv_{def} \bigwedge_{i \neq j} \forall x \neg(X_i(x) \wedge X_j(x))$$

$$\varphi_2 \equiv_{def} \forall x(\neg \exists y S(y, x) \rightarrow X_0(x))$$

$$\varphi_3 \equiv_{def} \forall x \forall y(S(x, y) \rightarrow (\bigvee_{(i,a,j) \in \Delta} (X_i(x) \wedge Q_a(x) \wedge X_j(y))))$$

$$\varphi_4 \equiv_{def} \forall x \exists y(x < y \wedge (\bigvee_{\substack{j \in F \\ (i,a,j) \in \Delta}} (X_i(y) \wedge Q_a(y))))$$

con $i, j \in Q$.

Aquí es importante notar que φ_1, φ_2 y φ_3 son las mismas fórmulas que en la definición (3.2) para autómatas finitos y que φ_4 es la fórmula que codifica la condición de aceptación del ABN.

A continuación demostramos la verdad de las fórmulas anteriores con respecto a los modelos de palabra e interpretando las variables con respecto a los conjuntos P_i de la definición (3.31).

Lema 3.26. $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_1$.

Demostración. La demostración es igual a la del lema (3.2). □

Lema 3.27. $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_2$.

Demostración. La demostración es igual a la del lema (3.3). □

Lema 3.28. $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_3$.

Demostración. La demostración es igual a la del lema (3.4). □

Lema 3.29. $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_4$.

Demostración. Sea $j \in F$. Sin pérdida de generalidad, podemos suponer que $j \in infty(\rho_\alpha)$ con lo que se tiene que $\exists^\omega i \in \mathbb{N}$ tal que $(i, \alpha(i), j) \in \Delta$. Basta demostrar que $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \forall x \exists y(x < y \wedge X_i(y) \wedge Q_a(y))$, para algunas i, a tales que $(i, a, j) \in \Delta$.

$\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \forall x \exists y(x < y \wedge X_i(y) \wedge Q_a(y))$

si y sólo si $\hat{\alpha} \models_{[\vec{P},p/\vec{X},x]} \exists y(x < y \wedge X_i(y) \wedge Q_a(y))$ para toda $p \in \mathbb{N}$

si y sólo si $\hat{\alpha} \models_{[\vec{P},p,q/\vec{X},x,y]} x < y \wedge X_i(y) \wedge Q_a(y)$ para toda $p \in \mathbb{N}$ y para alguna $q \in \mathbb{N}$

si y sólo si $p < q$, $q \in P_i$, $q \in Q_a^{\mathcal{I}}$ para toda $p \in \mathbb{N}$ y para alguna $q \in \mathbb{N}$

Sea $p \in \mathbb{N}$. Ya que ρ_α es infinita y además j aparece una infinidad de veces en ρ_α, podemos elegir q tal que $p < q$ y $(\rho_q, \alpha(q), j) \in \Delta$. Tomemos $i = \rho_q$ y $a = \alpha(q)$, entonces $p < q$, $q \in P_i$, $q \in Q_a^{\mathcal{I}}$.

□

De manera análoga al caso finito, cada modelo de palabra de $L(\mathcal{A})$ satisface las condiciones descritas por cada fórmula φ_i. Ahora podemos definir un enunciado $\varphi_{\mathcal{A}}$ que describa a toda secuencia de aceptación de $\mathcal{A}$ y verificar todas las palabras de $L(\mathcal{A})$ lo satisfacen.

Lema 3.30. *Si* $\varphi_{\mathcal{A}} \equiv_{def} \exists X_0 ... \exists X_k(\varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4)$, *entonces* $\hat{\alpha} \models \varphi_{\mathcal{A}}$.

Demostración. Por los lemas (3.26), (3.27), (3.28) y (3.29) tenemos que:

$\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_1$ y $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_2$ y $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_3$ y $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_4$

si y sólo si $\hat{\alpha} \models_{[\vec{P}/\vec{X}]} \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4$

si y sólo si $\hat{\alpha} \models \exists X_0 ... \exists X_k(\varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4)$,

si y sólo si $\hat{\alpha} \models \varphi_{\mathcal{A}}$.

□

El enunciado $\varphi_{\mathcal{A}}$ describe las condiciones necesarias de cualquier ejecución de aceptación del autómata $\mathcal{A}$ y por lo que toda palabra, aceptada por $\mathcal{A}$, cumple. Además, $\varphi_{\mathcal{A}}$ determina a un lenguaje $L(\varphi_{\mathcal{A}})$.

Teorema 3.7. *Dado $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ un ABN. Existe φ en S1S tal que $\alpha \in L(\mathcal{A})$ si y sólo si $\alpha \in L(\varphi)$.*

Demostración. Como $\alpha \in L(\mathcal{A})$, por el lema (3.30), podemos construir $\varphi_{\mathcal{A}}$ tal que $\alpha \in L(\varphi_{\mathcal{A}})$.

□

El teorema anterior nos asegura que todo lenguaje ω-regular es S1S-definible. Lo siguiente es ver que el recíproco también se cumple.

3.4.2. De lógica a ABN

Para ir de la lógica S1S a los autómatas de Büchi, veremos que dada $\varphi \in$ S1S, existe un ABN $\mathcal{A}$ tal que $L(\varphi) = L(\mathcal{A})$. el procedimiento será, una vez más, de la misma manera que para los autómatas finitos.

Empecemos puntualizando las definiciones análogas a aquellas utilizadas en el caso de autómatas finitos de la sección (3.1.2).

Definición 3.33. *Sean $i \in \mathbb{N}$ y $P \subseteq \mathbb{N}$. La función $\mu : \mathbb{N} \times \mathcal{P}(\mathbb{N}) \to \{0, 1\}$ determina la pertenencia de un elemento en un subconjunto de $\mathbb{N}$ y se define como:*

$$\mu(i, P) = \begin{cases} 1 & i \in P \\ 0 & i \notin P \end{cases}$$

Extendemos μ para k subconjuntos de $\mathbb{N}$: Si $\vec{P} = (P_1, ..., P_k)$, entonces $\mu(i, \vec{P}) = (\mu(i, P_1), ..., \mu(i, P_k))$.

Definición 3.34. *Sean $k \geq 0$, $\alpha = a_0 a_1 ...$ en Σ^ω y $\vec{P} = (P_1, ..., P_k), k$ subconjuntos de $\mathbb{N}$. $\gamma : \Sigma^\omega \times \mathcal{P}(\mathbb{N})^k \to \Sigma^\omega_{\times k}$ es una función (o familia de funciones) de codificación tal que:*

$$\gamma(\alpha, \vec{P}) = (a_0, \mu(0, \vec{P}))(a_1, \mu(1, \vec{P}))...$$

con $\Sigma_{\times k} = \Sigma \times \{0, 1\}^k$.

En particular si $k = 0$, entonces $\gamma : \Sigma^\omega \to \Sigma^\omega$ con $\gamma(\alpha) = \alpha$.

La función de codificación γ toma una palabra de Σ^ω y k subconjuntos de $\mathbb{N}$ y devuelve una palabra sobre el alfabeto $\Sigma_{\times k}$, de manera que, si $\alpha' = a'_0 a'_1 ... = \gamma(\alpha, P_1, ..., P_k)$, entonces $a'_i = (a_i, b_1, ..., b_k)$, donde $b_j = 1$ si y sólo si $i \in P_j$. Así, la pertenencia de i en cada P_j está codificada en el $i-$ésimo símbolo de α'.

Ejemplo. *Si* $\Sigma = \{a, b, c\}, \alpha = abca\ldots, P_1 = \{0, 3\}, P_2 = \{1, 2\}, P_3 = \{0, 1, 3\}$*, entonces:*

- $\gamma(\alpha, P_1) = \gamma(abca\ldots, P_1) = (a, 1)(b, 0)(c, 0)(a, 1)\ldots$
- $\gamma(\alpha, P_2, P_3) = \gamma(abca\ldots, P_2, P_3) = (a, 0, 1)(b, 1, 1)(c, 1, 0)(a, 0, 1)\ldots$

El siguiente teorema muestra la importancia de la función γ.

Teorema 3.8. *Para cada fórmula* $\varphi(X_1, ..., X_k) \in$ *S1S0, con* $X_1, ...X_k$ *variables libres, existe un ABN* $\mathcal{A}_\varphi$ *tal que:*

$$\breve{\alpha} \models_{[\vec{X}/\vec{P}]} \varphi(X_1, ..., X_k) \text{ si y sólo si } \gamma(\alpha, \vec{P}) \in L(\mathcal{A}_\varphi)$$

Demostración. La demostración de este teorema es la misma que la del teorema para palabras finitas (3.2) con la única diferencia en el paso inductivo para la negación:

Supongamos que $\varphi \equiv_{def} \neg\psi$. Por hipótesis de inducción hay un ABN $\mathcal{A}_\psi$ tal que $\alpha \in L(\psi)$ si y sólo si $\alpha \in L(\mathcal{A}_\psi)$, por lo que $L(\psi)$ es ω-regular.

Como $L(\varphi) = L(\neg\psi) = \Sigma^\omega \setminus L(\psi)$, por el teorema (3.6) $L(\varphi)$ es ω-regular y además podemos construir un ABN $\mathcal{A}_\varphi$ que lo reconozca.

□

Aquí es importante notar que no podemos dar una regla general para la construcción de un ABN que acepte el complemento de un lenguaje, a partir del ABN que reconoce dicho lenguaje.

Corolario. *Si* φ *es un enunciado de S1S0, existe un ABN* $\mathcal{A}_\varphi$ *tal que* $L(\varphi) = L(\mathcal{A}_\varphi)$.

Demostración. Por el teorema (3.8), existe un ABN $\mathcal{A}_\varphi$ tal que $\breve{\alpha} \models \varphi$ si y sólo si $\gamma(\alpha, (P_1, ..., P_k)) \in L(\mathcal{A}_\varphi)$. Ahora, como φ es un enunciado y no tiene variables libres, entonces $k = 0$, por lo que $\gamma(\alpha, (P_1, ..., P_k)) = \gamma(\alpha) = \alpha$ y por lo tanto, $\breve{\alpha} \models \varphi$ si y sólo si $\alpha \in L(\mathcal{A}_\varphi)$.

□

Ahora ya podemos establecer el vínculo entre la lógica S1S y los autómatas de Büchi.

Teorema 3.9. *Dada* $\varphi \in$ *S1S, existe un ABN* $\mathcal{A}_\varphi$ *tal que* $\alpha \in L(\varphi)$ *si y sólo si* $\alpha \in L(\mathcal{A}_\varphi)$.

Demostración. Sabemos que S1S0 es equivalente a S1S. Sean $\varphi \in$ S1S y $\psi \in$ S1S0 tales que $\psi = \varphi^{\bullet}$, por lo que $\check{\alpha} \models \psi$ si y sólo si $\hat{\alpha} \models \varphi$. Por el teorema (3.8), existe un ABN $\mathcal{A}_\psi$ tal que $\check{\alpha} \models \psi$ si y sólo si $\alpha \in L(\mathcal{A}_\psi)$. Basta con tomar $\mathcal{A}_\varphi = \mathcal{A}_\psi$, por lo que $\alpha \in L(\varphi)$ si y sólo si $\alpha \in L(\mathcal{A}_\varphi)$.

□

Finalmente tenemos todo lo necesario para poder enunciar y demostrar el teorema de Büchi para lenguajes ω-regulares.

Teorema 3.10 (**Teorema de Büchi).** *Un lenguaje de palabras infinitas es ω-regular si y sólo si es S1S-definible.*

Demostración. ($\Rightarrow$). Del teorema (3.7) tenemos que: Dado $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$ un ABN. Si $\alpha \in L(\mathcal{A}) \subseteq \Sigma^\omega$, entonces existe $\varphi \in$ S1S tal que $\alpha \in L(\varphi)$.

($\Leftarrow$). Del teorema (3.9) tenemos que: Dada $\varphi \in$ S1S, existe un ABN $\mathcal{A}_\varphi$ tal que $\alpha \in L(\varphi)$ si y sólo si $\alpha \in L(\mathcal{A}_\varphi)$.

□

Un resultado importante es la decidibilidad de la lógica S1S con interpretación sobre $\mathbb{N}$ que se sigue de la demostración del teorema (3.10).

Corolario 3.3. *La lógica S1S es decidible. Es decir, dada una fórmula φ en S1S, podemos determinar si existe o no $\alpha \in \Sigma^\omega$ tal que $\hat{\alpha} \models \varphi$.*

Demostración. Sea φ una fórmula en S1S. Por el teorema (3.9), podemos construir un ABN $\mathcal{A}_\varphi$ tal que $\alpha \in L(\mathcal{A}_\varphi)$ si y sólo si $\alpha \in L(\varphi)$, para toda $\alpha \in \Sigma^*$.

Por el corolario (3.2), podemos determinar si $L(\mathcal{A}_\varphi) \neq \emptyset$, es decir, podemos saber si existe $\alpha \in \Sigma^\omega$ tal que $\alpha \in L(\mathcal{A}_\varphi)$.

□

En este capítulo demostramos los teoremas centrales de este trabajo: El teorema de Büchi/Elgot para palabras finitas y el teorema de Büchi para ω-palabras. Para esto, definimos las nociones de autómata de Büchi y de lenguaje ω-regular y verificamos que cumplen con las mismas propiedades de cerradura que cumplen los autómatas finitos y los lenguajes regulares.

Debido a la complejidad que se deriva de trabajar con palabras infinitas, para demostrar la cerradura bajo complemento, recurrimos a un análisis algebraico que nos permitió determinar que cualquier lenguaje es ω-regular si

y sólo si es la unión finita de lenguajes de la forma $U \cdot V^{\omega}$, siendo U y V clases de alguna congruencia de índice finito. Por último, utilizando la propiedad de saturación de la congruencia $\equiv_{\mathcal{A}}$ determinada por un ABN $\mathcal{A}$ que reconoce a un lenguaje ω-regular L, se demostró que el complemento de L también es ω-regular.

Lo anterior nos permite ver a los lenguajes ω-regulares y los autómatas de Büchi como una generalización, sobre palabras infinitas, de los lenguajes regulares y los autómatas finitos, respectivamente. Este hecho permitió que en las demostraciones de ambos teoremas se procediera de la misma manera:

- Primero verificamos que cualquier lenguaje L regular/ω-regular es S1S-definible. Para esto, construimos un enunciado φ que describe las condiciones necesarias que cumple toda secuencia de aceptación de un autómata que acepta L para después verificar que $L(\varphi) = L$.

- Después verificamos que cualquier lenguaje S1S-definible es regular/ω-regular. Para esto, utilizamos las propiedades de cerradura de los autómatas y la equivalencia entre S1S y S1S0, con lo que construimos un autómata $\mathcal{A}$ que aceptara un lenguaje S1S-definible L de tal forma que $L(\mathcal{A}) = L$.

Por último, verificamos un resultado importante y que se sigue de ambos teoremas: la decidibilidad de S1S bajo las correspondientes interpretaciones. Resultado que utilizaremos para demostrar la decidibilidad de la aritmética de Presburguer.

Capítulo 4

Aritmética de Presburger

La aritmética de Presburger es la teoría de primer orden de los números naturales con la suma, nombrada así en honor a Mojżesz Presburger, quien la introdujo en 1929. La signatura de la aritmética de Presburger contiene sólo la adición como operación y un conjunto de axiomas que incluyen un esquema de inducción. La diferencia que guarda con la aritmética usual de Peano es que esta última incluye la operación multiplicación.

Como una aplicación interesante de los resultados desarrollados hasta ahora en este trabajo, mostramos, mediante una codificación adecuada, la decidibilidad de la aritmética de Presburger, la cual, será una consecuencia directa de la decidibilidad de la lógica de segundo orden S1S demostrada anteriormente.

Para la codificación, la idea es representar a los números como conjuntos cuyos elementos son las posiciones, de la representación binaria invertida, en donde se tiene un dígito igual a 1. Es conveniente utilizar la representación binaria invertida para que el bit b_i coincida con la posición i de la misma. Por ejemplo, el número 25, con representación binaria invertida 10011, es codificado en el conjunto $\{0, 3, 4\}$.

Así, definiremos una fórmula $+(X_1, X_2, X_3)$ que expresa que los conjuntos finitos X_1, X_2, X_3, representan a los números x_1, x_2, x_3, tales que $x_1 + x_2 = x_3$. Esta fórmula describe el algoritmo de la suma binaria procediendo dígito por dígito, usando el sucesor y la existencia de un conjunto auxiliar para el acarreo. De esta manera, cualquier fórmula $\varphi(x_1, ..., x_n)$ en la signatura de Presburger se transforma inductivamente en una fórmula correspondiente $\varphi'(X_1, ..., X_n)$ en S1S, usando cuantificadores de segundo orden en vez de cuantificadores sobre números. La decidibilidad de la aritmética de Presbur-

ger se sigue de aplicar esta traducción e invocando la decidibilidad de S1S.

Procedemos primero definiendo formalmente la aritmética de Presburger como una teoría de primer orden (*Pres*).

Definición 4.1. *La sintaxis del lenguaje Pres extiende la sintaxis de la parte común a todo lenguaje lógico de primer orden y definimos su signatura como:*

- *El conjunto de símbolos de predicado:* $\mathcal{P} = \emptyset$
- *El conjunto de símbolos de función:* $\mathcal{F} = \{S^{(1)}, +^{(2)}\}$
- *El conjunto de símbolos de constante:* $\mathcal{C} = \{0, 1\}$

Tenemos que las fórmulas atómicas de *Pres* son de la forma $t_1 = t_2$, con $t_1, t_2 \in T$ (términos), siendo estos alguna de las siguientes: una variable, la aplicación de la función S, la aplicación de la función $+$, una de las constantes 0 y 1.

Definición 4.2. *Sea* $\mathcal{M} = \langle \mathbb{N}, \mathcal{I} \rangle$ *una interpretación para Pres, donde* $\mathcal{I}$ *se define como:*

$$\begin{aligned} 0^{\mathcal{I}} &= 0 \\ 1^{\mathcal{I}} &= 1 \\ S^{\mathcal{I}} : \mathbb{N} \to \mathbb{N} &\quad \textit{es la función sucesor en } \mathbb{N}. \\ +^{\mathcal{I}} : \mathbb{N}^2 \to \mathbb{N} &\quad \textit{es la función suma en } \mathbb{N}. \end{aligned}$$

A continuación, definiremos una función recursiva de normalización sobre las fórmulas atómicas de *Pres* ($t_1 = t_2$) para obtener un conjunto base de fórmulas atómicas *normalizadas*. Esta función de normalización es una aportación nuestra que nos facilitará la definición de una función de traducción de *Pres* a S1S.

Definición 4.3. *Un término es simple si y sólo si es de alguna de las siguientes formas:*

$$0, 1, x, S(x), x + y$$

Nuestro objetivo es transformar cada fórmula de *Pres* de manera que todas sus subfórmulas atómicas sean de la forma $t = x$, donde t es un término simple. A esta clase de fórmulas las llamaremos normalizadas y esto lo logramos mediante la siguiente definición.

Definición 4.4. *Sean x, y, z, símbolos de variable; t, t_1, t_2, términos que no son variables y ψ, φ, fórmulas en Pres. La función de normalización $\circ$ se define como:*

$$(0 = x)^\circ = 0 = x \tag{4.1}$$
$$(1 = x)^\circ = 1 = x \tag{4.2}$$
$$(y = x)^\circ = y = x \tag{4.3}$$
$$(S(y) = x)^\circ = S(y) = x \tag{4.4}$$
$$(z + y = x)^\circ = z + y = x \tag{4.5}$$
$$(S(t) = z)^\circ = \exists y(S(y) = z \wedge (t = y)^\circ) \tag{4.6}$$
$$(t + y = z)^\circ = \exists x(x + y = z \wedge (t = x)^\circ) \tag{4.7}$$
$$(x + t = z)^\circ = \exists y(x + y = z \wedge (t = y)^\circ) \tag{4.8}$$
$$(t_1 + t_2 = z)^\circ = \exists x \exists y(x + y = z \wedge (t_1 = x)^\circ \wedge (t_2 = y)^\circ) \tag{4.9}$$
$$(x = t)^\circ = (t = x)^\circ \tag{4.10}$$
$$(t_1 = t_2)^\circ = \exists x((t_1 = x)^\circ \wedge (t_2 = x)^\circ) \tag{4.11}$$
$$(\psi \vee \varphi)^\circ = (\psi)^\circ \vee (\varphi)^\circ \tag{4.12}$$
$$(\neg\varphi)^\circ = \neg(\varphi)^\circ \tag{4.13}$$
$$(\exists x\varphi)^\circ = \exists x(\varphi)^\circ \tag{4.14}$$

Ya que el conjunto de símbolos de variable es infinito, suponemos que siempre que $\circ$ introduce una variable en alguna fórmula φ°, este símbolo de variable es nuevo, es decir, el símbolo no figuraba en φ. Esto, en particular, es importante para las ecuaciones (4.11), (4.6), (4.7), (4.8) y (4.9).

Es importante notar que $\circ$ es una función recursiva, que los primeros cinco casos son los casos base de la recursión y corresponden a la clase de ecuaciones que deseamos, las de la forma $t = x$, con t simple; los siguientes cuatro casos corresponden a la forma $t = z$, donde t no es simple; la ecuación (4.10) corresponde al caso en que el término a la derecha de una igualdad no es una variable pero el izquierdo sí; finalmente la ecuación (4.11) resuelve el caso en el que ninguno de los términos en la ecuación son variables. Observese también que los casos para fórmulas no atómicas (4.12, 4.13, 4.14) son simplemente homomórficos.

Ejemplos.

1. $(S(x+v)=z)^\circ = \exists y(S(y)=z \wedge x+v=y)$
2. $(x+y+z=w)^\circ = \exists v(v+z=w \wedge x+y=v)$
3. $(u+v+S(w)=z)^\circ = \exists x \exists y(x+y=z \wedge u+v=x \wedge S(w)=y)$

4.1. La suma en S1S

Antes de definir la traducción de *Pres* a S1S, veamos la codificación que utilizaremos para representar cada número como un conjunto finito de posiciones que será interpretado en S1S.

Debemos recordar que S1S es una lógica que nos permite hablar sobre cadenas (finita o infinitas) sobre un alfabeto Σ. En su signatura contamos sólo con predicados que nos dicen si alguna posición es sucesora de otra o si en ella se encuentra algún símbolo de Σ, es decir, sólo podemos predicar sobre posiciones o conjuntos de posiciones de palabras. Por lo que tiene sentido *transformar* cada número en alguna palabra que nos permita hacer esto. Un buen candidato es la representación binaria.

Para cualquier natural $k \geq 0$ obtendremos, mediante la función *bin*, la representación binaria invertida (el primer símbolo de la cadena es el bit menos significativo), algo de la forma $bin(k) = b_0 b_1 ... b_n$. Después, usando la función pos_1 obtenemos, de esta cadena, el conjunto P de posiciones i tales que $b_i = 1$. Con lo que $pos_1(bin(k)) = P$.

Definición 4.5. *Sea $k \in \mathbb{N}$. Definimos la función $bin : \mathbb{N} \to \{0,1\}^*$ para obtener la representación binaria invertida como: $bin(k) = b_0 b_1 ... b_{n-1}$ si y sólo si $b_{n-1} = 1$ y $k = \sum_{i=0}^{n-1} b_i 2^i$.*

En adelante haremos referencia a $bin(k)$ como la representación binaria de k. Observemos que la condición $b_{n-1} = 1$ evita que existan colas de ceros a la derecha, lo cual ocasiona que la representación sea única.

Definición 4.6. *Sea $b = b_0 ... b_{n-1}$ una representación binaria. Denotamos la longitud de b con $|b|$, es decir, $|b| = n$.*

Definición 4.7. *Sea $k \in \mathbb{N}$ y $bin(k) = b_0 b_1 ... b_{n-1}$ su representación binaria. Definimos la función $pos_1 : \{0,1\}^* \to \mathcal{P}(\mathbb{N})$ que regresa el conjunto de posiciones iguales a 1 en una representación binaria como:*

$$pos_1(b_0 b_1 ... b_{n-1}) = \{i \mid b_i = 1, 0 \leq i < n\}$$

Es importante notar que podemos agregar cualquier cantidad de dígitos 0 a la derecha de una representación binaria sin alterar el resultado de la función pos_1.

Con todo lo anterior, definiremos, por simplicidad, la función de codificación *enc* que será tan sólo la composición de las funciones *bin* y pos_1.

Definición 4.8. *Definimos la función de codificación* $enc : \mathbb{N} \to \mathcal{P}(\mathbb{N})$ *que mapea cada número* $k \in \mathbb{N}$ *en un conjunto finito* $P \subset \mathbb{N}$ *como:* $enc = pos_1 \circ bin$

Ejemplos.

- *El número 0 se codifica como:* $bin(0) = 0$, $pos_1(0) = \emptyset$. *Por lo que* $enc(0) = \emptyset$.
- *El número 1 se codifica como:* $bin(1) = 1$, $pos_1(1) = \{0\}$. *Por lo que* $enc(1) = \{0\}$.
- *Veamos como codificar el número 25. Primero obtenemos la representación binaria* $bin(25) = 10011$. *Después obtenemos las posiciones donde hay* 1, $pos_1(10011) = \{0, 3, 4\}$, *con lo que* $enc(25) = \{0, 3, 4\}$.

Como ya lo mencionamos, S1S nos permite hablar sobre posiciones de cadena y *Pres* es la lógica de la aritmética de los naturales y la suma. Para poder establecer un vínculo entre ellas, necesitamos expresar la suma de alguna manera en S1S. Para ello vamos a definir una fórmula $+(X, Y, Z)$ que exprese que para todo x, y, z en $\mathbb{N}$, si $enc(x) = X^{\mathcal{I}}$, $enc(y) = Y^{\mathcal{I}}$, entonces $enc(z) = Z^{\mathcal{I}}$, donde $x + y = z$. Esta fórmula describe el algoritmo de la suma binaria procediendo dígito por dígito (en términos de las posiciones de palabras binarias) utilizando una variable de conjunto auxiliar V que interpretaremos como la codificación del acarreo.

Definiremos un conjunto de fórmulas las cuales codificarán la relación de pertenencia de la variable de primer orden libre (denotadas con φ) o de su sucesor (denotadas con ψ) en las variables de segundo orden que figuren libres en la fórmula.

Por ejemplo, la fórmula $\psi_1(x, V) \equiv_{def} \exists y(S(x, y) \wedge V(y))$ definida de esta manera será interpretada como: *El sucesor de* x *pertenece a* V.

En las siguientes definiciones se considera que $V, X, Y, Z \in V_2$, $x, y \in V_1$, son variables en S1S. Definimos las siguientes fórmulas:

Definición 4.9. *La relación de pertenencia del sucesor de x en V y de x en Z se codifica con las siguientes fórmulas:*

$$\begin{aligned}
\psi_1(x,V) &\equiv_{def} \exists y(S(x,y) \wedge V(y)) \\
\psi_{11}(x,V,Z) &\equiv_{def} \psi_1(x,V) \wedge Z(x) \\
\psi_{01}(x,V,Z) &\equiv_{def} \neg\psi_1(x,V) \wedge Z(x) \\
\psi_{10}(x,V,Z) &\equiv_{def} \psi_1(x,V) \wedge \neg Z(x) \\
\psi_{00}(x,V,Z) &\equiv_{def} \neg\psi_1(x,V) \wedge \neg Z(x)
\end{aligned}$$

Si tenemos la fórmula $\psi_{d_0d_1}(x,V,Z)$. El dígito d_0 codifica la pertenencia del sucesor de x en V, y d_1 la de x en Z, utilizando el 1 para la afirmación y el 0 para la negación. Por ejemplo, si $d_0d_1 = 01$ la fórmula $\psi_{01}(x,V,Z)$ será satisfecha si $\sigma(x)+1 \notin \sigma(V)$ y $\sigma(x) \in \sigma(Z)$ para un estado de variables σ.

De manera análoga, definimos una relación de pertenencia de una variable de primer orden a tres variables de segundo orden.

Definición 4.10. *La relación de pertenencia de x en V, X, Y, se codifica con las siguientes fórmulas:*

$$\begin{aligned}
\varphi_{111}(x,V,X,Y) &\equiv_{def} V(x) \wedge X(x) \wedge Y(x) \\
\varphi_{011}(x,V,X,Y) &\equiv_{def} \neg V(x) \wedge X(x) \wedge Y(x) \\
\varphi_{101}(x,V,X,Y) &\equiv_{def} V(x) \wedge \neg X(x) \wedge Y(x) \\
\varphi_{110}(x,V,X,Y) &\equiv_{def} V(x) \wedge X(x) \wedge \neg Y(x) \\
\varphi_{001}(x,V,X,Y) &\equiv_{def} \neg V(x) \wedge \neg X(x) \wedge Y(x) \\
\varphi_{010}(x,V,X,Y) &\equiv_{def} \neg V(x) \wedge X(x) \wedge \neg Y(x) \\
\varphi_{100}(x,V,X,Y) &\equiv_{def} V(x) \wedge \neg X(x) \wedge \neg Y(x) \\
\varphi_{000}(x,V,X,Y) &\equiv_{def} \neg V(x) \wedge \neg X(x) \wedge \neg Y(x)
\end{aligned}$$

De manera similar, en la fórmula $\varphi_{b_0b_1b_2}(x,V,X,Y)$, los dígitos b_0, b_1, b_2 codifican la pertenencia de la variable x en V, X, Y. Por ejemplo, si $b_0b_1b_2 = 010$ la fórmula $\varphi_{010}(x,V,X,Y)$ será satisfecha si $\sigma(x) \notin \sigma(V)$ y $\sigma(x) \in \sigma(X)$ y $\sigma(x) \notin \sigma(Y)$ para un estado de variables σ.

El siguiente grupo de fórmulas es una abreviación de la implicación entre el segundo y el primer grupo de fórmulas y codifica cada paso del algoritmo de la suma.

Definición 4.11. *Dados los dígitos $d_0 \ldots d_4$, definimos la fórmula $\varphi_{d_0d_1d_2\to d_3d_4}(x,V,X,Y,Z)$ como sigue:*

$$\begin{aligned}
\varphi_{111\to 11}(x,V,X,Y,Z) &\equiv_{def} \varphi_{111}(x,V,X,Y) \to \psi_{11}(x,V,Z)\\
\varphi_{011\to 10}(x,V,X,Y,Z) &\equiv_{def} \varphi_{011}(x,V,X,Y) \to \psi_{10}(x,V,Z)\\
\varphi_{101\to 10}(x,V,X,Y,Z) &\equiv_{def} \varphi_{101}(x,V,X,Y) \to \psi_{10}(x,V,Z)\\
\varphi_{110\to 10}(x,V,X,Y,Z) &\equiv_{def} \varphi_{110}(x,V,X,Y) \to \psi_{10}(x,V,Z)\\
\varphi_{001\to 01}(x,V,X,Y,Z) &\equiv_{def} \varphi_{001}(x,V,X,Y) \to \psi_{01}(x,V,Z)\\
\varphi_{010\to 01}(x,V,X,Y,Z) &\equiv_{def} \varphi_{010}(x,V,X,Y) \to \psi_{01}(x,V,Z)\\
\varphi_{100\to 01}(x,V,X,Y,Z) &\equiv_{def} \varphi_{100}(x,V,X,Y) \to \psi_{01}(x,V,Z)\\
\varphi_{000\to 00}(x,V,X,Y,Z) &\equiv_{def} \varphi_{000}(x,V,X,Y) \to \psi_{00}(x,V,Z)
\end{aligned}$$

La fórmula $\varphi_{b_0b_1b_2\to d_0d_1}(x,V,X,Y,Z)$ codifica el algoritmo de la suma binaria en la posición $\sigma(x)$. Es decir, supongamos que $\sigma(x) = i$, $\sigma(V) = enc(c_0...c_n)$ corresponde al acarreo, $\sigma(X) = enc(a_0...a_n)$, $\sigma(Y) = enc(e_0...e_n)$ y $\sigma(Z) = enc(s_0...s_n)$. Ahora fijémonos en en el i-ésimo símbolo de cada representación binaria al aplicar el algoritmo de la suma.

$$\begin{array}{cccc}
 & i & i+1 & \\
 & \downarrow & \downarrow & \\
c_0... & c_i & c_{i+1} & ...c_n \\
a_0... & a_i & a_{i+1} & ...a_n \\
e_0... & e_i & e_{i+1} & ...e_n \quad + \\
\hline
s_0... & s_i & s_{i+1} & ...s_n
\end{array}$$

Si suponemos que $b_0b_1b_2 = 011$ y $d_0d_1 = 10$, la fórmula $\varphi_{011\to 10}(x,V,X,Y,Z)$ será satisfecha si y sólo si $c_i = 0$, $a_i = 1$ $e_i = 1$, $c_{i+1} = 1$ y $s_i = 0$.

$$\begin{array}{cccc}
 & i & i+1 & \\
 & \downarrow & \downarrow & \\
c_0... & 0 & 1 & ...c_n \\
a_0... & 1 & a_{i+1} & ...a_n \\
e_0... & 1 & e_{i+1} & ...e_n \quad + \\
\hline
s_0... & 0 & s_{i+1} & ...s_n
\end{array}$$

Que es precisamente, aplicar el algoritmo en la i-ésima posición.

Definición 4.12. *La siguiente fórmula conjunta todos los casos del algoritmo de la suma.*

$$\begin{aligned}
\varphi_+(x,V,X,Y,Z) \equiv_{def} \varphi_{111\to 11}(x,V,X,Y,Z) \ \vee \varphi_{011\to 10}(x,V,X,Y,Z) \vee\\
\varphi_{101\to 10}(x,V,X,Y,Z) \ \vee \varphi_{110\to 10}(x,V,X,Y,Z) \vee\\
\varphi_{001\to 01}(x,V,X,Y,Z) \ \vee \varphi_{010\to 01}(x,V,X,Y,Z) \vee\\
\varphi_{100\to 01}(x,V,X,Y,Z) \ \vee \varphi_{000\to 00}(x,V,X,Y,Z)
\end{aligned}$$

Recordemos que al aplicar el algoritmo, cada dígito del acarreo es generado a partir de los valores de los operandos, además, el algoritmo se aplica sobre todas las posiciones de cada representación. Entonces podemos acotar, por algún cuantificador, las variables que serán interpretadas de esta manera.

Definición 4.13. *La siguiente fórmula permite expresar la ecuación de la forma $x + y = z$, con $x, y, z \in \mathbb{N}$.*

$$+(X, Y, Z) \equiv_{def} \forall x \exists V \varphi_+(x, V, X, Y, Z)$$

Es importante recalcar que cualquier representación binaria de longitud m puede extenderse a cualquier longitud $l > m$ agregando ceros a la derecha. Esto es indispensable para poder aplicar el algoritmo de la suma correctamente. Por ejemplo, si tenemos $r_a, r_e, r_s, r_c \in \{0,1\}^*$ las representaciones binarias de $a, e, s, c \in \mathbb{N}$ y además $a + e = s$, $|r_a| = p$ y $|r_e| = q$. Al aplicar el algoritmo de la suma en r_a y r_e tenemos:

$$\begin{array}{r|l} r_c: & c_0 c_1 ... c_n \\ r_a: & a_0 a_1 ... a_n \\ r_e: & e_0 e_1 ... e_n \quad + \\ \hline r_s: & s_0 s_1 ... s_n \end{array}$$

con n tal que:

$$n = \begin{cases} p & a_p = 1, e_p = c_p = 0 \\ q & e_q = 0, a_q = c_q = 0 \\ p+1 & a_p = c_p = 1 \\ q+1 & e_q = c_q = 1 \\ p+1 & \begin{array}{l} p = q,\ a_p = c_p = 1 \text{ o } e_p = c_p = 1 \text{ o} \\ e_p = a_p = 1 \text{ o } e_p = a_p = c_p = 1 \end{array} \end{cases}$$

y extendiendo con 0 a la derecha r_a o r_e de ser necesario. Es decir, las representaciones binarias de a, e, s y el acarreo que obtenemos al aplicar el algoritmo de la suma, tienen longitud n.

En lo que sigue, consideremos a $\mathcal{M} = \langle \mathbb{N}, \mathcal{I} \rangle$ una interpretación para S1S. Verifiquemos que la fórmula $+(X, Y, Z)$ está bien definida.

Lema 4.1. *Sean $a, e, s, n \in \mathbb{N}$, $r_a, r_e, r_s, r_c \in \{0,1\}^*$ y $P_a, P_e, P_s \subseteq \mathbb{N}$ tales que: $a + e = s$, las palabras r_a, r_e, r_s, son las representaciones binarias de a, e, s, respectivamente. r_c es la representación binaria del acarreo que se obtiene al aplicar el algoritmo de la suma para r_a, r_e. Las longitudes de las representaciones binarias son $|r_a| = |r_e| = |r_s| = |r_c| = n$. Los conjuntos P_a, P_e, P_s, corresponden a las codificaciones de a, e, s, es decir $P_a = enc(a)$, $P_e = enc(e)$, $P_s = enc(s)$. Entonces:*

$$\mathcal{M} \models_{[P_a,P_e,P_s/X,Y,Z]} +(X,Y,Z)$$

Demostración. Por la definición de $+(X,Y,Z)$ tenemos que:

$$\mathcal{M} \models_{[P_x,P_y,P_z/X,Y,Z]} +(X,Y,Z)$$
si y sólo si
$$\mathcal{M} \models_{[P_x,P_y,P_z,C,p/X,Y,Z,V,x]} \varphi_+(x,V,X,Y,Z)$$

para alguna $C \subseteq \mathbb{N}$ y para toda $p \in \mathbb{N}$. Sin importar que p tomemos, basta con encontrar C para satisfacer la fórmula $\varphi_+(x,V,X,Y,Z)$.

Sea $C = pos_1(r_c)$. Aquí es importante notar que $pos_1(r_a) = P_a$, $pos_1(r_e) = P_e$ y $pos_1(r_s) = P_s$.

Para toda posición $p \in \mathbb{N}$, al aplicar el algoritmo de la suma, tenemos los siguientes ocho casos:

000. Si $c_p = a_p = e_p = 0$, sumamos $0 + 0 + 0$ por lo que el resultado en la posición p y el acarreo para la posición $p+1$ es 0, esto es, $c_{p+1} = s_p = 0$. Por lo tanto, si $p \notin C \wedge p \notin P_x \wedge p \notin P_y$, entonces, $p+1 \notin C \wedge p \notin P_z$. De manera que la fórmula $\varphi_{000\to00}(x,V,X,Y,Z)$ es satisfecha.

Siguiendo el mismo razonamiento:

001. Si $c_p = a_p = 0, e_p = 1$, entonces, $c_{p+1} = 0, s_p = 1$. Por lo tanto, la fórmula $\varphi_{001\to01}(x,V,X,Y,Z)$ es satisfecha.

010. Si $c_p = e_p = 0, a_p = 1$, entonces, $c_{p+1} = 0, s_p = 1$. Por lo tanto, la fórmula $\varphi_{010\to01}(x,V,X,Y,Z)$ es satisfecha.

011. Si $c_p = 0, a_p = e_p = 1$, entonces, $c_{p+1} = 1, s_p = 0$. Por lo tanto, la fórmula $\varphi_{011\to10}(x,V,X,Y,Z)$ es satisfecha.

100. Si $c_p = 1, a_p = e_p = 0$, entonces, $c_{p+1} = 0, s_p = 1$. Por lo tanto, la fórmula $\varphi_{100\to01}(x,V,X,Y,Z)$ es satisfecha.

101. Si $c_p = 1, a_p = 0, e_p = 1$, entonces, $c_{p+1} = 1, s_p = 0$. Por lo tanto, la fórmula $\varphi_{101\to 10}(x, V, X, Y, Z)$ es satisfecha.

110. Si $c_p = a_p = 1, e_p = 0$, entonces, $c_{p+1} = 1, s_p = 0$. Por lo tanto, la fórmula $\varphi_{110\to 10}(x, V, X, Y, Z)$ es satisfecha.

111. Si $c_p = a_p = e_p = 1$, entonces, $c_{p+1} = s_p = 1$. Por lo tanto, la fórmula $\varphi_{111\to 11}(x, V, X, Y, Z)$ es satisfecha.

Notemos que los ocho casos anteriores corresponden al total de las fórmulas que definen a $\varphi_+(x, V, X, Y, Z)$.

□

Con esto ya contamos con todos los elementos para poder definir una traducción de *Pres* a S1S.

4.2. Traducción de *Pres* a S1S

Necesitamos establecer una relación entre la interpretaciones para S1S y *Pres* a partir de los estados de las variables. Ya que en *Pres* sólo hay variables de primer orden y como en S1S las variables de primer orden representan posiciones de palabras, cada variable de *Pres* será codificada con una variable de segundo orden en S1S. Para esto, podemos asumir que tenemos una función inyectiva que a cada variable de primer orden en *Pres* le asigna un símbolo de variable de segundo orden en S1S como sigue:

$$x \longmapsto X$$

Antes de definir la función de traducción y para facilitar la notación, vamos a introducir en S1S dos nuevas constantes: 0 y 1.

Definición 4.14. *Definimos el conjunto de símbolos de constante $\mathcal{C}$ de S1S como:*

$$\mathcal{C} = \{0, 1\}$$

y extendemos la función de interpretación con:

$$0^{\widehat{\mathcal{I}}} = \emptyset \ y \ 1^{\widehat{\mathcal{I}}} = \{0\}$$

Esta extensión permitirá definir limpiamente la función de traducción aunque queremos recalcar que cada una de las constantes recién introducidas es definible en S1S de la siguiente manera:

- $0 \equiv_{def} \exists X \forall x \neg X(x)$
- $1 \equiv_{def} \exists Y \forall x (\neg \exists y (y < x) \rightarrow Y(x))$

Es fácil ver que al interpretar las constantes así definidas tenemos que:

$$0^{\widehat{\mathcal{I}}} = 1 \text{ si y sólo si } \hat{\sigma}(X) = \emptyset \text{ y } 1^{\widehat{\mathcal{I}}} = 1 \text{ si y sólo si } \hat{\sigma}(Y) = \{0\}$$

Las cuales coinciden con la codificación de 0 y 1:

$$enc(0) = \emptyset \text{ y } enc(1) = \{0\}$$

Con esto podemos definir la función de traducción.

Definición 4.15. *La función de traducción* $* : \Phi_{Pres} \rightarrow \Phi_{S1S}$ *mapea fórmulas de Pres en fórmulas de S1S como sigue:*

$$\begin{aligned}
(0 = x)^* &= 0 = X \\
(1 = x)^* &= 1 = X \\
(y = x)^* &= Y = X \\
(S(x) = z)^* &= +(X, Y, Z) \wedge Y = 1 \\
(x + y = z)^* &= +(X, Y, Z) \\
(\neg\varphi)^* &= \neg\varphi^* \\
(\varphi \vee \psi)^* &= \varphi^* \vee \psi^* \\
(\exists x \varphi)^* &= \exists X \varphi^*
\end{aligned}$$

Nuestro objetivo ahora, es mostrar que esta traducción es fiel, en el sentido de que preserva satifacibilidad.

Sean $\mathcal{M} = \langle \mathbb{N}, \mathcal{I} \rangle$, $\widehat{\mathcal{M}} = \langle \mathbb{N}, \widehat{\mathcal{I}} \rangle$ interpretaciones para *Pres* y S1S, respectivamente.

Las siguientes definiciones dependen de $\mathcal{M}$ y $\widehat{\mathcal{M}}$.

Definición 4.16. *Dado un estado de las variables* σ *en* $\mathcal{M}$*, definimos el estado de las variables* $\hat{\sigma}$ *en* $\widehat{\mathcal{M}}$*, tal que:*

$$\hat{\sigma}(X) = enc(\sigma(x))$$

Veamos que los estados modificados se corresponden con esta definición.

Lema 4.2. *Sean σ un estado de las variables en Pres, x una variable de primer orden en Pres y $p \in \mathbb{N}$. Entonces:*

$$\widehat{\sigma}[X/enc(p)](X) = \widehat{\sigma[x/p]}(X)$$

Demostración.

$$\begin{array}{lll} \widehat{\sigma[x/p]}(X) & = enc(\sigma[x/p](x)) & \text{definición de } \widehat{\sigma} \\ & = enc(p) & \text{definición de estado modificado} \\ & = \widehat{\sigma}[X/enc(p)](X) & \text{definición de estado modificado} \end{array}$$

□

Ahora podemos verificar, procediendo por inducción, la traducción de *Pres* a S1S.

Lema 4.3. *La interpretación de cada fórmula φ en Pres coincide con la interpretación de su traducción φ^* en S1S. Es decir:*

$$\mathcal{I}_\sigma(\varphi) = \widehat{\mathcal{I}}_{\hat{\sigma}}(\varphi^*)$$

Demostración. Verificamos primero los casos base.

1. $(0 = x)^* = 0 = X$.

$$\begin{array}{lll} \mathcal{I}_\sigma(0 = x) = 1 & \text{si y sólo si} & \sigma(x) = 0 \\ & \text{si y sólo si} & enc(\sigma(x)) = enc(0) \\ & \text{si y sólo si} & enc(\sigma(x)) = \emptyset \\ & \text{si y sólo si} & \hat{\sigma}(X) = \emptyset \qquad \text{definición de } \hat{\sigma} \\ & \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(0 = X) = 1 \end{array}$$

2. $(1 = x)^* = 1 = X$.

$$\begin{array}{lll} \mathcal{I}_\sigma(1 = x) = 1 & \text{si y sólo si} & \sigma(x) = 1 \\ & \text{si y sólo si} & enc(\sigma(x)) = enc(1) \\ & \text{si y sólo si} & enc(\sigma(x)) = \{0\} \\ & \text{si y sólo si} & \hat{\sigma}(X) = \{0\} \qquad \text{definición de } \hat{\sigma} \\ & \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(1 = X) = 1 \end{array}$$

3. $(y = x)^* = Y = X$.

$$\begin{array}{lll} \mathcal{I}_\sigma(y = x) = 1 & \text{si y sólo si} & \sigma(y) = \sigma(x) \\ & \text{si y sólo si} & enc(\sigma(y)) = enc(\sigma(x)) \\ & \text{si y sólo si} & \hat{\sigma}(Y) = \hat{\sigma}(X) \qquad \text{definición de } \hat{\sigma} \\ & \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(Y = X) = 1 \end{array}$$

4. $(S(x) = z)^* = +(X, Y, Z) \wedge Y = 1$.

Notemos que $\widehat{\mathcal{I}}_{\hat{\sigma}}(Y = 1) = 1$ si y sólo si $\hat{\sigma}(Y) = 1^{\widehat{\mathcal{I}}} = \{0\} = enc(1)$. Además, por definición de $\hat{\sigma}$, $\hat{\sigma}(X) = enc(\sigma(x))$ y $\hat{\sigma}(Z) = enc(\sigma(z))$. Con lo que:

$$\begin{array}{lll} \mathcal{I}_\sigma(S(x) = z) = 1 & \text{si y sólo si} & S^{\mathcal{I}}(\sigma(x)) = \sigma(z) \\ & \text{si y sólo si} & \sigma(x) + 1 = \sigma(z) \\ & & \hat{\sigma}(X) = enc(\sigma(x)), \\ & \text{si y sólo si} & \hat{\sigma}(Z) = enc(\sigma(z)) \text{ y} \quad \text{lema (4.1)} \\ & & \hat{\sigma}(Y) = enc(1) \\ & \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(+(X, Y, Z) \wedge Y = 1) = 1 \end{array}$$

5. $(x + y = z)^* = +(X, Y, Z)$.

Por definición de $\hat{\sigma}$, $\hat{\sigma}(X) = enc(\sigma(x))$, $\hat{\sigma}(Z) = enc(\sigma(z))$ y $\hat{\sigma}(Y) = enc(\sigma(y))$. Con lo que:

$\mathcal{I}_\sigma(x + y = z) = 1$

$$\begin{array}{lll} \text{si y sólo si} & \sigma(x) + \sigma(y) = \sigma(z) \\ & \hat{\sigma}(X) = enc(\sigma(x)), \\ \text{si y sólo si} & \hat{\sigma}(Z) = enc(\sigma(z)) \text{ y} \quad \text{lema (4.1)} \\ & \hat{\sigma}(Y) = enc(\sigma(y)) \\ \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(+(X, Y, Z)) = 1 \end{array}$$

Procedemos ahora con el paso inductivo. Suponemos que para cualquier fórmula φ en *Pres* se tiene que $\mathcal{I}_\sigma(\varphi) = \widehat{\mathcal{I}}_{\hat{\sigma}}(\varphi^*)$.

1. $(\neg\varphi)^* = \neg\varphi^*$.

$$\begin{array}{lll} \mathcal{I}_\sigma(\neg\varphi) = 1 & \text{si y sólo si} & \mathcal{I}_\sigma(\varphi) = 0 \\ & \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(\varphi^*) = 0 \qquad \text{H.I.} \\ & \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(\neg\varphi^*) = 1 \end{array}$$

2. $(\varphi \vee \psi)^* = \varphi^* \vee \psi^*$.

$$\begin{array}{lll} \mathcal{I}_\sigma(\varphi \vee \psi) = 1 & \text{si y sólo si} & \mathcal{I}_\sigma(\varphi) = 1 \text{ o } \mathcal{I}_\sigma(\psi) = 1 \\ & \text{si y sólo si} & \mathcal{I}_\sigma(\varphi^*) = 1 \text{ o } \mathcal{I}_\sigma(\psi^*) = 1 \quad \text{H.I.} \\ & \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(\varphi^* \vee \psi^*) = 1 \end{array}$$

3. $(\exists x\varphi)^* = \exists X\varphi^*$.

$\mathcal{I}_\sigma(\exists x\varphi) = 1$

$$\begin{array}{lll} \text{si y sólo si} & \mathcal{I}_{\sigma[x/p]}(\varphi) = 1 & \text{para algún } p \in \mathbb{N} \\ \text{si y sólo si} & \widehat{\mathcal{I}}_{\widehat{\sigma[x/p]}}(\varphi^*) = 1 & \text{H.I.} \\ \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}[X/enc(p)]}(\varphi^*) = 1 & \text{lema (4.2)} \\ \text{si y sólo si} & \widehat{\mathcal{I}}_{\hat{\sigma}}(\exists X\varphi^*) = 1 & \end{array}$$

□

El siguiente corolario es inmediato

Corolario 4.1. *Si φ es un enunciado en Pres, entonces:*

$$\mathcal{M} \models \varphi \text{ si y sólo si } \widehat{\mathcal{M}} \models \varphi^*$$

Finalmente podemos demostrar nuestro objetivo acerca de la decidibilidad de la aritmética de Presburger.

Teorema 4.1. *La aritmética de Presburger es decidible. Es decir, dada una fórmula φ en Pres, podemos determinar si existe o no un estado σ de variables tal que $\mathcal{M} \models_\sigma \varphi$.*

Demostración. Dadas φ y σ podemos decidir si $\widehat{\mathcal{M}} \models \varphi^*$, puesto que la lógica S1S es decidible, pero esto es equivalente, por el corolario anterior, a que $\mathcal{M} \models \varphi$.

Por lo tanto, *Pres* es decidible.

□

En este capítulo demostramos la decidibilidad de la aritmética de Presburger que es la teoría de primer orden de los números naturales con la suma. Definimos la signatura *Pres* en la lógica de primer orden. Con esto, definimos una función $\circ$ de normalización que transforma cualquier fórmula de *Pres* en una fórmula cuyas subfórmulas atómicas son de la forma $t = x$, donde t es

un término simple. Después, definimos una fórmula $+(X, Y, Z)$ en S1S que describe el algoritmo de la suma binaria.

La función $\circ$ para fórmulas de *Pres* y la fórmula $+(X, Y, Z)$ en S1S, facilitaron la definición de una función $*$ de traducción de fórmulas de *Pres* a fórmulas de S1S. Por último, demostramos que la función $*$ preserva la satisfacibilidad de las fórmulas y dado que S1S es decidible, entonces *Pres* es decidible.

Conclusiones

A lo largo de este trabajo buscamos caracterizar a los lenguajes regulares y ω-regulares a través de la lógica. Para eso, definimos una signatura del lenguaje lógico de manera que nos permitiera describir que un conjunto de palabras sobre un alfabeto finito fuera reconocible por un autómata finito o un autómata de Büchi.

En un primer intento recurrimos a la lógica S0S de primer orden, sólo para verificar que, debido a que la noción de un camino finito de longitud arbitraria no puede ser definida en un lenguaje de primer orden, el poder expresivo no alcanza para poder describir un lenguaje regular, y por lo tanto, tampoco es posible describir un lenguaje ω-regular.

Por lo anterior, definimos la lógica S1S, con un mayor poder expresivo y que es la extensión de S0S sobre la lógica monádica de segundo orden que es, a su vez, la extensión de la lógica de primer orden que permite la cuantificación sobre predicados monádicos.

Definimos el autómata de Büchi a partir de la definición de un autómata finito y por tanto, como una extensión del mismo sobre palabras infinitas. Demostramos que, a diferencia de los autómatas finitos, los autómatas de Büchi no deterministas tienen un mayor poder expresivo que los deterministas. También verificamos que los autómatas de Büchi cumplen con las mismas propiedades de cerradura que los autómats finitos. Demostrar las propiedades de cerradura se redujo a construir los correspondientes autómatas con la excepción del complemento, propiedad que requirió de un mayor análisis y se sirvió de técnicas algebraicas presentadas a detalle aquí.

Definimos los lenguajes ω-regulares como los lenguajes aceptados por los autómatas de Büchi y que por lo tanto, cumplen con las propiedades de cerradura. Demostramos que todo lenguaje ω-regular se puede expresar como una unión finita de lenguajes de la forma $U\dot{V}^{\omega}$, donde U y V son regulares.

Con todo lo anterior, se establece una analogía entre los autómatas finitos

y los autómatas de Büchi, los lenguajes regulares y los ω-regulares. Todo esto nos permite preservar esa relación a través de las demostraciones de los teoremas de Büchi, el primero sobre lenguajes regulares y el segundo sobre ω-regulares. Los teoremas de Büchi establecen la equivalencia entre las clases de los lenguajes S1S-definibles y las clases de lenguajes regulares y ω-regulares.

Para ambos teoremas seguimos el mismo procedimiento. Primero demostramos que si un lenguaje es reconocible por un autómata, entonces es S1S-definible. Después demostramos que si un lenguaje es S1S-definible, entonces podemos construir un autómata que lo acepte.

Por último mostramos, como una aplicación interesante, la decidibilidad de la aritmética de Presburger a partir de la decidibilad de la lógica monádica de segundo orden S1S, la cual se sigue de los teoremas de Büchi. Para esto, construimos una traducción de la aritmética de Presburger a la lógica S1S, la cual se sirve de una función de normalización que es aportación nuestra.

Como extensión de este trabajo es natural preguntarse en que otras estructuras, además de las palabras, puede darse un resultado análogo a los teoremas de Büchi.

Una generalización a considerar es sobre árboles binarios propios [3], en donde cada nodo es una hoja o tiene dos sucesores y es representado como una palabra finita sobre el alfabeto $\{0, 1\}$, donde 0 denota al *hijo izquierdo* y 1 al *hijo derecho*. Para cada árbol t, el modelo de palabra tendría como universo un subconjunto cerrado bajo prefijo $dom(t)$ del conjunto $\{0, 1\}^*$, es decir, para cada palabra $w \in dom(t)$, ambas o ninguna de las palabras $w0, w1$ están en $dom(t)$. Así, un árbol sobre el alfabeto Σ se puede ver como una función $t : dom(t) \to \Sigma$. La signatura de S1S tendría que adaptarse definiendo dos relaciones sucesor S_0, S_1, sucesor izquierdo y sucesor derecho, respectivamente, con $(w, w0) \in S_0^{\mathcal{I}}$ y $(w, w0) \in S_1^{\mathcal{I}}$, para $w, w0, w1 \in dom(t)$.

Otra generalización son la gráficas dirigidas [3], en donde las etiquetas de los vértices pertenecen a un alfabeto Σ y además contamos con un alfabeto E para etiquetar a las aristas. Así, los nodos n, n' conectados por la arista b son representados por la palabra nbn'. Nuevamente, la signatura de S1S se adaptaría considerando el nuevo alfabeto incluyendo una relación *arista* E_b por cada símbolo $b \in E$. Teniendo al conjunto V como universo de una interpretación, cada $E_b^{\mathcal{I}}$ corresponde a un subconjunto disjunto de $V \times V$. Cabe mencionar que con las gráficas acíclicas, los modelos de palabras y los de árboles binarios se pueden ver como un caso particular de éstas, en los que las relaciones sucesor de cada interpretación corresponden a una relación

arista.

Es importante notar que en el marco de la lógica, no existe gran diferencia ni dificultad en la adaptación de las nociones de definibilidad desde el dominio de las palabras a los dominios extendidos de árboles y gráficas, además, la transición de modelos finitos a modelos infinitos no representa ningun problema conceptual, sólamente es necesario adaptar la signatura bajo consideración y cambiar la clase de modelos aceptados.

Bibliografía

[1] J.R. Büchi. (1960). *Weak second-order arithmetic and finite automata*. Z. Math. Logik Grundl. Math. 6, 66-92.

[2] C.C. Elgot. (1961). *Decision problems of finite automata design and related arithmetics*. Trans. Amer. Math. Soc. 98, 21-52.

[3] W. Thomas. (1996). *Languages, Automata, and Logic*. Technical Report 9607, Institut für Informatik und Praktische Mathematik, Christian-Albrechts-Universität zu Kiel, Germany.

[4] W. Thomas. (1990). *Automata on Infinite Objects*. In J. van Leeuwen, editor, Handbook of Theoretical Computer Science, volume B: Formal Models and Semantics, pages 133-192. Elsevier Science Publishers, Amsterdam.

[5] R. Dustin Wehr. (2007). *Monadic Second Order Logic and Automata on Infinite Words: Büchi's Theorem*.

[6] M. Huth. M. Ryan. (2004). *Logic in computer science. Modelling and Reasoning about Systems*. Cambridge University Press.

[7] Arnold L. Rosenberg. (2010). *The Pillars of Computation Theory. State, Encoding, Nondeterminism*. Springer.

[8] Favio Ezequiel Miranda Perea. Araceli Liliana Reyes Cabello. Lourdes Del Carmen González Huesca. (2013) *Lógica de Primer Orden: Introducción y Sintaxis*. Lógica Computacional 2013-2, Notas de clase 4. Facultad de Ciencias. UNAM.

[9] Favio Ezequiel Miranda Perea. Araceli Liliana Reyes Cabello. Lourdes Del Carmen González Huesca. (2013) *Semántica de la Lógica de Primer Orden*. Lógica Computacional 2013-2, Notas de clase 6. Facultad de Ciencias. UNAM.

[10] W. Thomas. (2010). *On monadic theories on monadic predicates*. Fields of Logic and Computation Lecture Notes in Computer Science Volume 6300, pp 615-626.

[11] Ganesh Gopalakrishnan. (2006). *Computation Engineering: Applied Automata Theory and Logic*. Springer.

[12] Jean-Eric Pin. (1996). *Logic, semigroups and automata on words*. Annals of Mathematics and Artificial Intelligence 16, pp. 343-384. J.C. Baltzer AG, Science Publishers.

[13] John C. Martin. (2004). *Lenguajes formales y teoría de la computación, 3a ed*. McGraw-Hill Interamericana.

[14] Hossein Hojjat. (2010). *Monadic Second Order Logic*. LARA.

[15] Volker Diekert, Paul Gastin. (2007). *First-order definable languages*. Institut für Formale Methoden der Informatik. Universität Stuttgart. Laboratoire Spécification et Vérification. École Normale Supérieure de Cachan.

[16] O. Matz, N. Schweikardt. (2008). *Expressive power of monadic logics on words, trees, pictures, and graphs*. Logic and Automata: History and Perspectives, pp. 531-552. Amsterdam University Press.

[17] Nir Piterman. (2007). *From nondeterministic Büchi and Streett automata to deterministic parity automata*. Logical Methods in Computer Science, Vol. 3 (3:5) 2007, pp. 1-21.

[18] C. C. Chang, H. J. Keisler. (1990). *Model Theory*. Studies in logic and the foundations of mathematics. Volume 73. Elsevier Science Publishers B.Y.

[19] Raynal, Michel (2012). *Concurrent Programming: Algorithms, Principles, and Foundations*. Springer Science & Business Media. p. 9.