

Introducción a la Lógica y la Computación

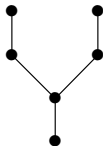
Mariana Badano Héctor Gramaglia
Pedro Sánchez Terraf Mauricio Tellechea
Guido Ivetta

FaMAF, 13 de octubre de 2021



Ejes de Contenidos

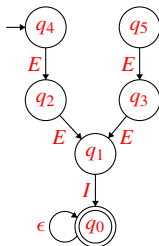
1 Estructuras Ordenadas



2 Lógica Proposicional

$$\frac{\frac{\frac{[\varphi \wedge \psi]_1 \wedge E}{\psi}}{\varphi} \wedge I}{\varphi \wedge \psi \rightarrow \psi \wedge \varphi} \rightarrow I_1$$

3 Lenguajes y Autómatas



Parte 3: Lenguajes y Autómatas

- Alejandro Tiraboschi y otros, *Introducción a la Lógica. Parte III: Lenguajes y Autómatas*.
- J. Hopcroft, R. Motwani y J. Ullman, *Introduction to Automata Theory, Languages and Computation* (2001; hay versión en castellano).

- 1** Lenguajes y Automatas
 - Ejemplo de autómata
 - Definición formal de lenguaje
 - Definición formal de autómata finito
 - Lenguajes aceptados y regulares
 - Caracterizando lenguajes aceptados
 - Autómatas finitos no deterministas

Los autómatas son un modelo matemático de dispositivos que **computan**.

Los autómatas son un modelo matemático de dispositivos que **computan**. En una primera aproximación teórica, podemos clasificarlos de acuerdo a qué tipo de memoria tienen.

Los autómatas son un modelo matemático de dispositivos que **computan**. En una primera aproximación teórica, podemos clasificarlos de acuerdo a qué tipo de memoria tienen.

Tipo de memoria

- Memoria sólo para el programa (finita, “registros”).
- Memoria tipo pila.
- Memoria tipo RAM.

Los autómatas son un modelo matemático de dispositivos que **computan**. En una primera aproximación teórica, podemos clasificarlos de acuerdo a qué tipo de memoria tienen.

Tipo de memoria

- Memoria sólo para el programa (finita, “registros”).
- Memoria tipo pila.
- Memoria tipo RAM.

En los dos últimos casos, se supone que no hay límites de recursos (pero se usan sólo finitos para cada computación).

Un **lenguaje** es una familia de strings (“cadenas”, “palabras”).

Un **lenguaje** es una familia de strings (“cadenas”, “palabras”).

¿Por qué hablamos de lenguajes?

Un problema puede *codificarse* como un lenguaje.

Un **lenguaje** es una familia de strings (“cadenas”, “palabras”).

¿Por qué hablamos de lenguajes?

Un problema puede *codificarse* como un lenguaje.

Ejemplo: Sea L_p el conjunto de sucesiones de números binarios que representan primos ($10 \in L_p$, $101 \in L_p$, $100 \notin L_p$).

Un **lenguaje** es una familia de strings (“cadenas”, “palabras”).

¿Por qué hablamos de lenguajes?

Un problema puede *codificarse* como un lenguaje.

Ejemplo: Sea L_p el conjunto de sucesiones de números binarios que representan primos ($10 \in L_p$, $101 \in L_p$, $100 \notin L_p$).

Decidir si n es primo se reduce a comprobar si su representación binaria está o no en L_p .

Funciones representadas con lenguajes

Ejemplo

Computar $f(x) := x^3$ usando un lenguaje.

Autómatas finitos

Ejemplo de autómeta

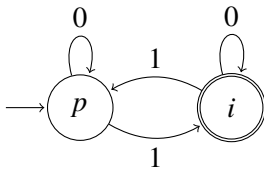


Diagrama de transición

Estados Los “nodos” en el diagrama.

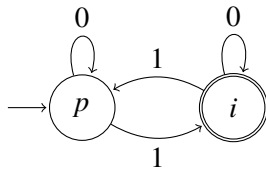
Transiciones Son las flechas, tienen un *símbolo* como etiqueta.

Estado inicial Se indica por la flecha que no tiene origen.

Estados finales Se indican por un doble círculo.

Símbolos Inputs que conoce el autómeta.

Ejemplo de autómata



Traza o ejecución

Dada una cadena, es la sucesión de estados por los que transita el autómata mientras la recorre, empezando con el estado inicial.

Aceptación

Una palabra es aceptada si su traza concluye en algún estado final.

Definición formal de lenguaje

Alfabeto Conjunto finito no vacío, habitualmente usamos Σ para referirnos a alfabetos. En el ejemplo anterior $\Sigma = \{0, 1\}$.

Definición formal de lenguaje

Alfabeto Conjunto finito no vacío, habitualmente usamos Σ para referirnos a alfabetos. En el ejemplo anterior $\Sigma = \{0, 1\}$.

Cadena Sucesión finita de símbolos de Σ .

A la sucesión de longitud 0 la denotamos con ϵ .

Dado una cadena α y un símbolo x $x\alpha$ es la cadena que tiene como primer símbolo x y luego α .

Definición formal de lenguaje

Alfabeto Conjunto finito no vacío, habitualmente usamos Σ para referirnos a alfabetos. En el ejemplo anterior $\Sigma = \{0, 1\}$.

Cadena Sucesión finita de símbolos de Σ .

A la sucesión de longitud 0 la denotamos con ϵ .

Dado una cadena α y un símbolo x $x\alpha$ es la cadena que tiene como primer símbolo x y luego α .

Potencias de Σ Definimos por recursión en $k \in \mathbb{N}$.

$$\Sigma^0 = \{\epsilon\}$$

$$\Sigma^{k+1} = \{x\alpha \mid \alpha \in \Sigma^k, x \in \Sigma\}$$

$$\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n$$

Si $\Sigma \neq \emptyset$, Σ^* es infinito.

$$\alpha \in \Sigma^k \iff |\alpha| = k.$$



UNC
Universidad
Nacional
de Córdoba



Lenguaje

Un **lenguaje** es un subconjunto de Σ^* .

Un **lenguaje** es un subconjunto de Σ^* .

Ejemplos

L_p sucesiones binarias que representan números primos.

L_{py} cadenas ASCII que son programas de Python que compilan.

L_I cadenas con una cantidad impar de unos.

$L_=$ cadenas con igual cantidad de ceros que de unos.



UNC
Universidad
Nacional
de Córdoba



1613-2013
400
AÑOS

Lenguaje

Un **lenguaje** es un subconjunto de Σ^* .

Ejemplos

L_p sucesiones binarias que representan números primos.

L_{py} cadenas ASCII que son programas de Python que compilan.

L_I cadenas con una cantidad impar de unos.

$L_=$ cadenas con igual cantidad de ceros que de unos.

Casos especiales

\emptyset el lenguaje vacío.

$\{\epsilon\}$ el lenguaje “unidad”.

Σ considerado a los símbolos como cadenas.

Σ^* todas las palabras del alfabeto Σ .

Concatenación pegar dos palabras del mismo alfabeto. Formalmente por recursión en una de las palabras:

$$\epsilon\beta := \beta$$

$$(x\alpha)\beta := x(\alpha\beta)$$

Subcadena α es *subcadena* de β si existen γ y γ' tales que $\beta = \gamma\alpha\gamma'$.

Prefijo α es *prefijo* de β si $\beta = \alpha\gamma'$.

Sufijo α es *sufijo* de β si $\beta = \gamma\alpha$.

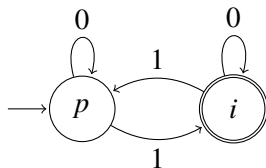
Autómatas finitos deterministas (DFA)

Un **autómata finito determinista** \mathbb{A} consta de los siguientes componentes:

- 1 Un conjunto finito de estados Q .
- 2 Un alfabeto Σ .
- 3 Una función de transición (que codifica las aristas de la representación gráfica) $\delta: Q \times \Sigma \rightarrow Q$.
- 4 Un estado inicial $q_0 \in Q$.
- 5 Un conjunto de estados finales $F \subseteq Q$.

Usaremos la notación $\mathbb{A} = (Q, \Sigma, \delta, q_0, F)$ para indicar los componentes del autómata \mathbb{A} .

Formalizando el autómata anterior



$$Q = \{p, i\}$$

$$\Sigma = \{0, 1\}$$

$$q_0 = p$$

$$F = \{i\}$$

δ	0	1
p	p	i
i	i	p

El tipo de la función de transición δ fuerza a que de cada estado salga una (y sólo una) arista por cada símbolo del alfabeto.

El tipo de la función de transición δ fuerza a que de cada estado salga una (y sólo una) arista por cada símbolo del alfabeto.

Por eso hablamos de autómatas **deterministas**: si el autómata está en un estado sabemos que dado un símbolo de entrada pasará a un único estado siguiente.

El tipo de la función de transición δ fuerza a que de cada estado salga una (y sólo una) arista por cada símbolo del alfabeto.

Por eso hablamos de autómatas **deterministas**: si el autómata está en un estado sabemos que dado un símbolo de entrada pasará a un único estado siguiente.

Generalizando: dada una palabra existe un único camino desde el estado inicial que consume toda la palabra.

Lenguaje de un autómata y lenguajes regulares

La función de transición δ puede ser extendida a $\hat{\delta}: Q \times \Sigma^* \rightarrow Q$ para que acepte palabras:

$$\hat{\delta}(q, \epsilon) := q$$

$$\hat{\delta}(q, x\alpha) := \hat{\delta}(\delta(q, x), \alpha)$$

Lenguaje de un autómata y lenguajes regulares

La función de transición δ puede ser extendida a $\hat{\delta}: Q \times \Sigma^* \rightarrow Q$ para que acepte palabras:

$$\hat{\delta}(q, \epsilon) := q$$

$$\hat{\delta}(q, x\alpha) := \hat{\delta}(\delta(q, x), \alpha)$$

$$\hat{\delta}(q, \epsilon) := q$$

$$\hat{\delta}(q, \beta x) := \delta(\hat{\delta}(q, \beta), x)$$

Lenguaje de un autómata y lenguajes regulares

La función de transición δ puede ser extendida a $\hat{\delta}: Q \times \Sigma^* \rightarrow Q$ para que acepte palabras:

$$\begin{aligned}\hat{\delta}(q, \epsilon) &:= q & \hat{\delta}(q, \epsilon) &:= q \\ \hat{\delta}(q, x\alpha) &:= \hat{\delta}(\delta(q, x), \alpha) & \hat{\delta}(q, \beta x) &:= \delta(\hat{\delta}(q, \beta), x)\end{aligned}$$

El **lenguaje del autómata** \mathbb{A} son las palabras que comenzando en el estado inicial terminan (según $\hat{\delta}$) en uno final:

$$L_{\mathbb{A}} := \{\alpha \in \Sigma^* \mid \hat{\delta}(q_0, \alpha) \in F\}$$

Lenguaje de un autómata y lenguajes regulares

La función de transición δ puede ser extendida a $\hat{\delta}: Q \times \Sigma^* \rightarrow Q$ para que acepte palabras:

$$\begin{aligned}\hat{\delta}(q, \epsilon) &:= q & \hat{\delta}(q, \epsilon) &:= q \\ \hat{\delta}(q, x\alpha) &:= \hat{\delta}(\delta(q, x), \alpha) & \hat{\delta}(q, \beta x) &:= \delta(\hat{\delta}(q, \beta), x)\end{aligned}$$

El **lenguaje del autómata** \mathbb{A} son las palabras que comenzando en el estado inicial terminan (según $\hat{\delta}$) en uno final:

$$L_{\mathbb{A}} := \{\alpha \in \Sigma^* \mid \hat{\delta}(q_0, \alpha) \in F\}$$

Dado un lenguaje cualquiera L , diremos que L es **regular** si existe un autómata finito determinista \mathbb{A} tal que $L = L_{\mathbb{A}}$.

Definición (Lenguaje de un estado)

$$L_q := \{\alpha \in \Sigma^* \mid \hat{\delta}(q_0, \alpha) = q\}$$

Probamos $\alpha \in L_q$ si y sólo si α cumple cierta propiedad, que está determinada por la información que representa cada estado.

Definición (Lenguaje de un estado)

$$L_q := \{\alpha \in \Sigma^* \mid \hat{\delta}(q_0, \alpha) = q\}$$

Probamos $\alpha \in L_q$ si y sólo si α cumple cierta propiedad, que está determinada por la información que representa cada estado.

Probamos simultáneamente todos los casos por inducción en el largo de la palabra.

Definición (Lenguaje de un estado)

$$L_q := \{\alpha \in \Sigma^* \mid \hat{\delta}(q_0, \alpha) = q\}$$

Probamos $\alpha \in L_q$ si y sólo si α cumple cierta propiedad, que está determinada por la información que representa cada estado.

Probamos simultáneamente todos los casos por inducción en el largo de la palabra.

Volviendo al ejemplo

$\alpha \in L_p \iff \alpha$ tiene una cantidad par de unos.

$\alpha \in L_i \iff \alpha$ tiene una cantidad impar de unos.

Definición (Lenguaje de un estado)

$$L_q := \{\alpha \in \Sigma^* \mid \hat{\delta}(q_0, \alpha) = q\}$$

Probamos $\alpha \in L_q$ si y sólo si α cumple cierta propiedad, que está determinada por la información que representa cada estado.

Probamos simultáneamente todos los casos por inducción en el largo de la palabra.

Volviendo al ejemplo

$\alpha \in L_p \iff \alpha$ tiene una cantidad par de unos.

$\alpha \in L_i \iff \alpha$ tiene una cantidad impar de unos.

La caracterización del lenguaje del autómata se deduce de la caracterización de los estados finales.

- $\alpha \in L_p \iff \alpha$ tiene una cantidad par de unos $\iff |\alpha|_1$ es par.
- $\alpha \in L_i \iff \alpha$ tiene una cantidad impar de unos $\iff |\alpha|_1$ es impar.

Definición

Dos autómatas \mathbb{A} , \mathbb{A}' son **equivalentes** si $L_{\mathbb{A}} = L_{\mathbb{A}'}$.

Definición

Dos autómatas \mathbb{A} , \mathbb{A}' son **equivalentes** si $L_{\mathbb{A}} = L_{\mathbb{A}'}$.

A continuación veremos otra clase de autómatas tales que todo DFA es equivalente a uno de nuestra clase (y viceversa).

Autómatas finitos no deterministas (NFA)

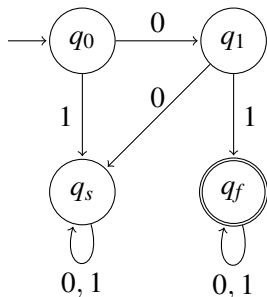
- Los autómatas no deterministas permiten que para estado haya cero, una o más transiciones por cada símbolo.

Autómatas finitos no deterministas (NFA)

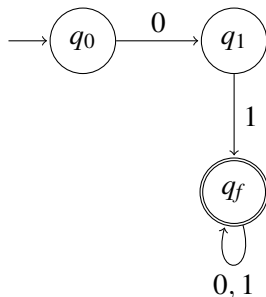
- Los autómatas no deterministas permiten que para estado haya cero, una o más transiciones por cada símbolo.
- Una palabra es aceptada por un NFA si hay **al menos** un camino que la consume y termina en un estado final.

Ejemplo: palabras que empiezan con 01

Determinista



No determinista



Un **autómata finito no determinista** \mathbb{A} consta de los siguientes componentes:

- 1 Un conjunto finito de estados Q .
- 2 Un alfabeto Σ .
- 3 Una función de transición $\delta: Q \times \Sigma \rightarrow \mathcal{P}(Q)$.
- 4 Un estado inicial $q_0 \in Q$.
- 5 Un conjunto de estados finales $F \subseteq Q$.

Un **autómata finito no determinista** \mathbb{A} consta de los siguientes componentes:

- 1 Un conjunto finito de estados Q .
- 2 Un alfabeto Σ .
- 3 Una función de transición $\delta: Q \times \Sigma \rightarrow \mathcal{P}(Q)$.
- 4 Un estado inicial $q_0 \in Q$.
- 5 Un conjunto de estados finales $F \subseteq Q$.

El único cambio está en la función de transición.

Ejemplo de NFA (1/2)

Aceptación para NFA

Ahora $\delta(p, x)$ es un conjunto: no se puede usar la composición de funciones directamente.

$$\hat{\delta}: Q \times \Sigma^* \rightarrow \mathcal{P}(Q)$$

$$\hat{\delta}(q, \epsilon) := \{q\}$$

$$\hat{\delta}(q, x\alpha) := \bigcup_{q' \in \delta(q, x)} \hat{\delta}(q', \alpha)$$

Aceptación para NFA

Ahora $\delta(p, x)$ es un conjunto: no se puede usar la composición de funciones directamente.

$$\hat{\delta}: Q \times \Sigma^* \rightarrow \mathcal{P}(Q)$$

$$\hat{\delta}(q, \epsilon) := \{q\}$$

$$\hat{\delta}(q, x\alpha) := \bigcup_{q' \in \delta(q, x)} \hat{\delta}(q', \alpha)$$

Definición

El **lenguaje del autómata** \mathbb{A} son las palabras que comenzando en el estado inicial terminan (según $\hat{\delta}$) en uno final:

$$L_{\mathbb{A}} := \{\alpha \in \Sigma^* \mid \hat{\delta}(q_0, \alpha) \cap F \neq \emptyset\}.$$

Ejemplo de NFA (2/2)