



Universidad Nacional Autónoma de México
Facultad de Ciencias
TEORÍA DE LA COMPUTACIÓN
NOTAS DE CLASE
Profesor: Enrique F. Soto-Astorga, MA/MSc.
Ayudante: Laura Itzel Rodríguez Dimayuga



1 0 1 1 0 0 0 1 1 0 0 1 0 1 1 1 0 0 0 1 0 1 0 1 0 1 1 0 1 1 1 0 1 0 0 1 1 0 0 0 1 }

Índice

1. Historia e Introducción

- 1.1. Turing 1936
- 1.2. Hilbert y el Entscheidungsproblem
- 1.3. Gödel y la incompletitud
- 1.4. Tesis de Church-Turing-Post
- 1.5. Automatas, Computabilidad y Complejidad
- 1.6. Incomputabilidad

2. Cadenas y alfabetos

3. Operaciones sobre alfabetos, lenguajes y cadenas

- 3.1. Operaciones sobre cadenas
- 3.2. Lenguajes y operaciones sobre lenguajes
 - 3.2.1. Muchas cosas se pueden resolver con lenguajes y autómatas, y muchas cosas que no .

4. Expresiones Regulares

1. Historia e Introducción

1.1. Turing 1936

1.2. Hilbert y el Entscheidungsproblem

1.3. Gödel y la incompletitud

1.4. Tesis de Church-Turing-Post

1.5. Automatas, Computabilidad y Complejidad

1.6. Incomputabilidad

2. Cadenas y alfabetos

Definición 1

Un *alfabeto* es un conjunto finito no vacío. Para referirnos al alfabeto utilizaremos Σ

Ejemplo:

- $\Sigma = \{0, 1\}$ es un alfabeto binario.
- $\Sigma = \{a, b, c, \dots, z\}$ es un alfabeto de letras.
- $\Sigma = \{\text{El conjunto de todos los símbolos ASCII}\}$

Definición 2

Los *símbolos* o *letras* son elementos de Σ , es decir, de nuestro alfabeto.

Ejemplo:

- Si $\Sigma = \{0, 1\}$, entonces 0 y 1 son símbolos de Σ .
- Si $\Sigma = \{a, b, c, \dots, z\}$, entonces a, b, c, \dots, z son símbolos de Σ .
- Si $\Sigma = \{\text{El conjunto de todos los símbolos ASCII}\}$, entonces cada símbolo ASCII es un símbolo de Σ .

Definición 3

Una *cadena* (string) es una secuencia finita de elementos de Σ .

Ejemplo:

- Si $\Sigma = \{0, 1\}$, entonces 0011 es una cadena sobre Σ .
- Si $\Sigma = \{a, b, c, \dots, z\}$, entonces abc es una cadena sobre Σ .
- Si $\Sigma = \{\text{El conjunto de todos los símbolos ASCII}\}$, entonces $Hello\$ \# @$ es una cadena sobre Σ .

Definición 4

El *tamaño* de una cadena es la cantidad de símbolos que la componen.

Ejemplo:

- Si $\Sigma = \{0, 1\}$, entonces $|0011| = 4$.

- Si $\Sigma = \{a, b, c, \dots, z\}$, entonces $|abc| = 3$.
- Si $\Sigma = \{\text{El conjunto de todos los símbolos ASCII}\}$, entonces $|\text{Hello}\$ \# @| = 8$.

Definición 5

A la *cadena vacía* la llamamos ϵ o λ , de manera de que $|\lambda| = 0$

La cadena vacía es diferente al conjunto vacío $\lambda \neq \emptyset$. Más adelante veremos que aparece en conjunto de todas las cadenas (Σ^*) sobre un alfabeto.

Definición 6

La *reversa* de una cadena w se denota como w^R y se obtiene invirtiendo el orden de los símbolos en w .

Ejemplo:

Si $w = 0011$, entonces $w^R = 1100$. Podemos definir al conjunto de las palabras palíndromas¹ como:

$$P = \{w \in \Sigma^* \mid w = w^R\}$$

Definición 7

Decimos que x es *prefijo* de y , si existe una cadena z tal que $y = xz$. Por ejemplo, 00 es prefijo de 0011, ya que existe $z = 11$ tal que $0011 = 00 \cdot 11$. Análogamente *sufijo* de y si existe una cadena z tal que $y = zx$. Por ejemplo, 11 es sufijo de 0011, ya que existe $z = 00$ tal que $0011 = 00 \cdot 11$.

Ejemplos:

- Si $w = 0011$ y $x = 00$, entonces x es prefijo de w , ya que existe $z = 11$ tal que $0011 = 00 \cdot 11$.
- Si $w = 0011$ y $y = 11$, entonces y es sufijo de w , ya que existe $z = 00$ tal que $0011 = 00 \cdot 11$.

3. Operaciones sobre alfabetos, lenguajes y cadenas

Definición 8

La *concatenación* de dos cadenas w e z se denota como $w \cdot z$ ó wz y es la cadena que resulta de unir w e z .

Nota: La concatenación aplica para **símbolos**, **cadenas** y **lenguajes**.

Ejemplo:

- Si $\Sigma = \{0, 1\}$, entonces 01 es una concatenación sobre **símbolos** de Σ .
- Si $w = 01$ e $z = 10$, entonces $wz = w \cdot z = 0110$ es una concatenación sobre **cadenas**.
- Si $L_1 = \{0, 1\}$ y $L_2 = \{1, 0\}$, entonces $L_1 L_2 = \{01, 00, 10, 11\}$ es una concatenación sobre **lenguajes**.

¹Una palabra es palíndroma si se lee igual de izquierda a derecha que de derecha a izquierda.

Definición 9

Podemos definir la *potencia* de una cadena w como $w^n = w \cdot w \cdots w$ (n veces), donde n es un número natural. Por ejemplo, si $w = 01$, entonces $w^3 = 010101$.

De manera inductiva tenemos que:

- $w^0 = \lambda$ (la cadena vacía)
- $w^{n+1} = w^n \cdot w$

La potencia la podemos aplicar a **símbolos**, **cadenas**, **lenguajes** y **alfabetos**.

Ejemplo:

Si w es una **cadena**, digamos $w = 01$, entonces:

- $w^0 = \lambda$
- $w^1 = 01$
- $w^2 = 0101$
- $w^3 = 010101$

Si L es un **lenguaje**, digamos $L = \{00, 1\}$, entonces:

- $L^0 = \{\lambda\}$
- $L^1 = \{00, 1\}$
- $L^2 = \{(00)00, (00)1, 1(00), (1)1\} = \{0000, 001, 100, 11\}$

Para lenguajes L_1 y L_2 , la potencia se define como:

- $L_1^0 = \{\lambda\}$
- $L_1^1 = L_1$
- $L_1^2 = \{xy \mid x \in L_1, y \in L_1\}$
- $L_1^3 = \{xyz \mid x \in L_1, y \in L_1, z \in L_1\}$

Aquí hay otro ejemplo con $L = \{ab, aab\}$:

$$\begin{aligned}\{ab, aab\}^0 &= \{\epsilon\} \\ \{ab, aab\}^1 &= \{ab, aab\} \\ \{ab, aab\}^2 &= \{abab, abaab, aabab, aabaaab\} \\ \{ab, aab\}^3 &= \{ab(ab)ab, ab(ab)aab, ab(aab)ab, (aab)abab, aab(aab)ab, aab(ab)aab, ab(aab)aab, aab(aab)aab\}\end{aligned}$$

Como tengo 2 cadenas y quiero ver todas las combinaciones de 2 cadenas, entonces tengo $2^2 = 4$ combinaciones. Si quiero ver todas las combinaciones de 3 cadenas, entonces tengo $2^3 = 8$ combinaciones.

Definición 10

También podemos definir la *potencia* de un alfabeto Σ como $\Sigma^n = \{w^n \mid w \in \Sigma^*\}$, donde Σ^* es el conjunto de todas las cadenas sobre Σ .

Por ejemplo, si $\Sigma = \{0, 1\}$, entonces $\Sigma^2 = \{00, 01, 10, 11\}$.

$$\begin{aligned}
\Sigma^0 &= \{\lambda\} \\
\Sigma^1 &= \{0, 1\} \\
\Sigma^2 &= \{00, 01, 10, 11\} \\
\Sigma^3 &= \{000, 001, 010, 011, 100, 101, 110, 111\} \\
&\vdots \\
\Sigma^n &= \{w^n \mid w \in \Sigma^*\}
\end{aligned}$$

Definición 11

Vamos a definir a la *cerradura de Kleene* o la *estrella de Kleene* de un alfabeto Σ como Σ^* ,

$$\begin{aligned}
\Sigma^* &= \bigcup_{n \geq 0} \Sigma^n = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots \\
\Sigma^+ &= \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \dots \\
\Sigma^* &= \Sigma^+ \cup \{\lambda\}
\end{aligned}$$

Donde Σ^n es la potencia del alfabeto. Podemos describir a Σ^* (sigma estrella) como el conjunto de todas las cadenas (incluida la cadena vacía) que se pueden formar a partir de los símbolos de Σ .

Ejemplo:

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

$$\Sigma^* = \{\lambda, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111, \dots\}$$

es decir, el conjunto de todos los números binarios (incluidos los de longitud cero, a.k.a. la cadena vacía).

Lema 1

Sea Q un conjunto finito. Entonces, el número de subconjuntos de Q es $2^{|Q|}$, donde $|Q|$ es el número de elementos en Q . A este conjunto se le llama *conjunto potencia* de Q y se denota como $\mathcal{P}(Q)$ ó 2^Q .

Demostración. Sea Q un conjunto finito con n elementos. Entonces, cada elemento de Q puede estar presente o no en un subconjunto, lo que nos da dos opciones (incluir o no incluir) cada elemento. Por lo tanto, el número total de subconjuntos de Q es $2^n = 2^{|Q|}$. ■

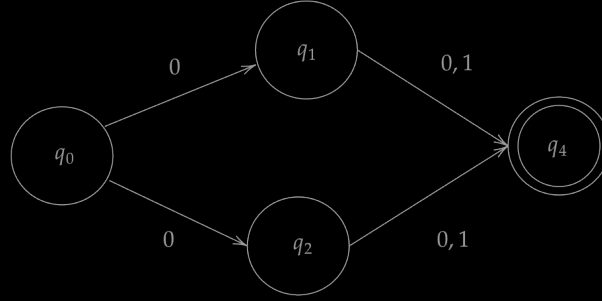


Figura 1: Ejemplo de un autómata finito determinista (DFA). Este es el autómata que acepta la expresión regular $0(0 + 1) = \{0, 00, 01\}$.

Más adelante cuando veamos autómatas finitos no deterministas, veremos su función de transición, la cual es una función que toma un estado y un símbolo de entrada y devuelve un conjunto de estados. Esta función se denota como $\delta : Q \times \{\Sigma \cup \lambda\} \rightarrow 2^Q$ (conjunto potencia), donde Q es el conjunto de estados y Σ es el alfabeto.

3.1. Operaciones sobre cadenas

Definición 12

- **Concatenación:** Dadas dos cadenas x e y , la concatenación es asociativa: $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ para toda cadena.
- La cadena vacía λ es la **identidad** de la concatenación: $x \cdot \lambda = \lambda \cdot x = x$ para toda cadena x .
- $|x \cdot y| = |x| + |y|$ para toda cadena x e y .
- $w^m w^n = w^{m+n}$

El conjunto de (Σ^*, \cdot) con λ como identidades es un monoide, ya que cumple con las propiedades de asociatividad e identidad.

3.2. Lenguajes y operaciones sobre lenguajes

Definición 13

Dado un alfabeto Σ , un **lenguaje** sobre Σ es un subconjunto de Σ^* . Dado un lenguaje $L \subseteq \Sigma^*$, definimos las siguientes operaciones sobre lenguajes:

- **Unión:** $L(a) \cup L(b) = L(a + b) = \{w \mid w \in L_1 \text{ o } w \in L_2\}$
- **Intersección:** $L_1 \cap L_2 = \{w \mid w \in L_1 \text{ y } w \in L_2\}$
- **Concatenación:** $L(a) \cdot L(b) = L(ab) = \{wz \mid w \in L(a), z \in L(b)\}$
- **Diferencia:** $L_1 - L_2 = \{w \mid w \in L_1 \text{ y } w \notin L_2\}$
- **Complemento:** $\overline{L} = \sim L = \{\lambda\} \cup (\Sigma^* - L)$
- **Cerradura de Kleene:** $L^* = \bigcup_{n \geq 0} L^n$
- $L^+ = L^* - \{\lambda\} = \bigcup_{n \geq 1} L^n$

Una analogía útil es pensar en los lenguajes es como el el lenguaje español. Sabemos que nuestro alfabeto es $\Sigma = \{a, b, c, \dots, z\}$ y que podemos formar palabras (cadenas) con estos símbolos. El lenguaje español $L(\text{esp})$ es un subconjunto de Σ^* . Otro ejemplo es pensar en un lenguaje de programación, donde el alfabeto es el conjunto de caracteres ASCII y el lenguaje es el conjunto de todas las cadenas que son programas válidos en ese lenguaje.

Algunos ejemplos de lenguajes son:

1. El lenguaje de todas las cadenas que contienen tienen el mismo número de 0s y 1s. Para contar el número de caracteres dentro de una cadena podemos usar la notación $\#0(w)$ para el número de 0s y $\#1(w)$ para el número de 1s. Entonces, este lenguaje se puede definir como:

$$L = \{w \in \Sigma^* \mid \#0(w) = \#1(w)\}$$

2. El lenguaje de todas las cadenas que comienzan con 0 y terminan con 1. Este lenguaje se puede definir como:

$$L = \{w \in \Sigma^* \mid w = 0x1, x \in \Sigma^*\}$$

3. El lenguaje de todas los números binarios que son primos

$0 = 0$	No es primo
$1 = 1$	No es primo
$10 = 2$	Es primo
$11 = 3$	Es primo
$100 = 4$	No es primo
$101 = 5$	Es primo
$110 = 6$	No es primo
$111 = 7$	Es primo
$1000 = 8$	No es primo
$1001 = 9$	No es primo
$1010 = 10$	No es primo
$1011 = 11$	Es primo
$1100 = 12$	No es primo
$1101 = 13$	Es primo
$1110 = 14$	No es primo
$1111 = 15$	No es primo

$$\{10, 11, 101, 111, 1011, 1101, \dots\}$$

Seguramente alguna vez hemos programado un verificador de números primos en algún lenguaje de programación. En general, veremos que los lenguajes son conjuntos de cadenas que cumplen ciertas propiedades. Y algunos lenguajes son capturados por expresiones regulares o autómatas.

3.2.1. Muchas cosas se pueden resolver con lenguajes y autómatas, y muchas cosas que no

Una de las cosas que resuelven los autómatas es decidir si una cadena pertenece o no a un lenguaje. Este es un problema de decisión (Entscheidungsproblem), gracias a Turing sabemos que no podemos obtener un sí o no para cualquier problema.

Un problema de decisión puede ser determinar si una cadena w pertenece a un lenguaje L . Por ejemplo, pensemos en el lenguaje de los números primos, si tenemos cualquier cadena podríamos preguntarnos si esta representación binaria es un número primo. ¿Dada una cadena w , $w \in L_p$?

En el siguiente capítulo trataremos las cadenas que sabemos que un autómata sí puede responder. Veremos que este autómata tiene limitaciones y más adelante trataremos de rebasarlas con el autómata con pila. Por ejemplo, el lenguaje $\{0^n 1^n \mid n \geq 0\}$ es el conjunto de todas las cadenas que tienen el mismo número de 0's seguido del mismo número de 1's. Este lenguaje no puede ser reconocido por un autómata finito, pero sí por un autómata con pila. Aunque este último tipo de autómata tiene más potencia, todavía hay lenguajes que no puede reconocer. Y más adelante veremos que hay lenguajes que no pueden ser reconocidos por ningún autómata, ni siquiera por una máquina de Turing. El quiebre del programa de Hilbert por querer encontrar un procedimiento efectivo para decidir si una proposición es verdadera o falsa quedó muerto.

Tenemos algunas propiedades sobre los lenguajes:

- **Conmutatividad de la unión:** $L_1 \cup L_2 = L_2 \cup L_1$
- **Asociatividad de la unión:** $(L_1 \cup L_2) \cup L_3 = L_1 \cup (L_2 \cup L_3)$



Figura 2: David Hilbert (1862-1943), un matemático alemán triste.

- **Conmutatividad de la intersección:** $L_1 \cap L_2 = L_2 \cap L_1$
- **Asociatividad de la intersección:** $(L_1 \cap L_2) \cap L_3 = L_1 \cap (L_2 \cap L_3)$
- **Distributividad de la intersección sobre la unión:** $L_1 \cap (L_2 \cup L_3) = (L_1 \cap L_2) \cup (L_1 \cap L_3)$
- **Distributividad de la unión sobre la intersección:** $L_1 \cup (L_2 \cap L_3) = (L_1 \cup L_2) \cap (L_1 \cup L_3)$
- **Identidad de la unión:** $L \cup \emptyset = \emptyset \cup L = L$
- **Identidad de la concatenación:** $L \cdot \lambda = \lambda \cdot L = L$
- **El aniquilador:** $L \cdot \emptyset = \emptyset \cdot L = \emptyset$
- **Distributividad de la concatenación(1):** $A(B \cup C) = (AB) \cup (AC)$
- **Distributividad de la concatenación(2):** $(A \cup B)C = (AC) \cup (BC)$
- **Ley de Morgan(1):** $\sim (L_1 \cup L_2) = \sim L_1 \cap \sim L_2$
- **Ley de Morgan(2):** $\sim (L_1 \cap L_2) = \sim L_1 \cup \sim L_2$
- **Propiedades de la cerradura de Kleene(1):** $L^* L^* = L^*$
- **Propiedades de la cerradura de Kleene(2):** $L^{**} = L^*$
- **Propiedades de la cerradura de Kleene(3):** $L^* = \{\lambda\} \cup LL^* = \lambda \cup L^* L$
- **Propiedades de la cerradura de Kleene(4):** $\emptyset^* = \{\lambda\}$
- **Propiedades de la cerradura de Kleene(5):** $L^+ = LL^* = L^* L$

La concatenación no se distribuye sobre la intersección, por ejemplo si $A = \{a, ab\}$, $B = \{b\}$ y $C = \{\epsilon\}$, entonces $A(B \cap C) = \{a, ab\}(\{b\} \cap \{\epsilon\}) = \{a, ab\}\emptyset = \emptyset$ pero $(AB) \cap (AC) = \{ab, abb\} \cap \{a, ab\} = ab$. Pero $\emptyset \neq ab$ **Contradicción**.

4. Expresiones Regulares