

# Matemáticas discretas II

Lenguajes y gramáticas  
[carlos.andres.delgado@correounivalle.edu.co](mailto:carlos.andres.delgado@correounivalle.edu.co)

Carlos Andrés Delgado S.  
Raúl E Gutierrez de Piñerez R.

Facultad de Ingeniería. Universidad del Valle

Marzo 2018

1 Lenguajes

2 Autómatas finitos

3 Gramáticas

4 Máquinas de Turing

1 Lenguajes

2 Autómatas finitos

3 Gramáticas

4 Máquinas de Turing

## El alfabeto

Un alfabeto es un conjunto finito no vacío cuyos elementos se llaman **símbolos**.

- Sea  $\Sigma = \{a, b\}$  el alfabeto que consta de los símbolos  $a$  y  $b$ . Las siguientes son cadenas sobre  $\Sigma$ :  $aba$ ,  $abaabaaa$ ,  $aaaab$ .
- El *alfabeto binario*  $\Sigma = \{0, 1\}$  son las cadenas sobre  $\Sigma$  que se definen como secuencias finitas de ceros y unos.
- Las cadenas son *secuencias ordenadas* y finitas de símbolos. Por ejemplo,  $w = aaab \neq w_1 = baaa$ .
- Sea  $\Sigma = \{a, b, c, \dots, x, y, z\}$  el alfabeto del idioma castellano.
- El alfabeto utilizado por muchos lenguajes de programación.
- Sea  $\Sigma = \{a, b, c\}$  entonces podemos formar todas las cadenas sobre  $\Sigma$  incluyendo la cadena vacía.

Notación usada en la teoría de lenguajes	
$\Sigma, \Gamma$	denotan alfabetos.
$\Sigma^*$	denota el conjunto de todas las cadenas que se pueden formar con los símbolos del alfabeto $\Sigma$ .
$a, b, c, d, e, \dots$	denotan símbolos de un alfabeto.
$u, v, w, x, y, z, \dots$	denotan cadenas, es decir, sucesiones finitas de símbolos de un alfabeto.
$\alpha, \beta, \gamma, \dots$	
$\epsilon$	denota la cadena vacía, es decir, la única cadena que no tiene símbolos.
$A, B, C, \dots, L, M, N, \dots$	denotan lenguajes (definidos más adelante).

- Si bien un alfabeto  $\Sigma$  es un conjunto finito,  $\Sigma^*$  es siempre un conjunto infinito (enumerable).
- Hay que distinguir entre los siguientes cuatro objetos, que son diferentes entre sí:  $\emptyset, \epsilon, \{\emptyset\}, \{\epsilon\}$

## Operaciones con alfabetos

Si  $\Sigma$  es un alfabeto,  $\sigma \in \Sigma$  denota que  $\sigma$  es un símbolo de  $\Sigma$ , por tanto, si

$$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

se puede decir que  $0 \in \Sigma$

Un alfabeto es simplemente un conjunto finito no vacío que cumple las siguientes propiedades, Dados  $\Sigma_1$  y  $\Sigma_2$  alfabetos

- Entonces  $\Sigma_1 \cup \Sigma_2$  también es un alfabeto.
- $\Sigma_1 \cap \Sigma_2$ ,  $\Sigma_1 - \Sigma_2$  y  $\Sigma_2 - \Sigma_1$  también son alfabetos.

## Conjunto Universal

El conjunto de todas las cadenas sobre un alfabeto  $\Sigma$ , incluyendo la cadena vacía, se denota por  $\Sigma^*$

- Sea  $\Sigma = \{0, 1\}$   
 $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 100, 010, 110, \dots\}$
- Sea  $\Sigma = \{a, b, c\}$ , entonces  
 $\Sigma^* = \{\epsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, aab, abc, baa, \dots\}$
- Sea  $\Sigma = \{a, b\}$ , entonces  
 $\Sigma^* = \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, baa, \dots\}$

## Cadenas

Dado un alfabeto  $\Sigma$  y dos cadenas  $u, v \in \Sigma^*$ , la concatenación de  $u$  y  $v$  se denota como  $u \cdot v$  o simplemente  $uv$  y se define así:

- 1 Si  $v = \epsilon$ , entonces  $u \cdot \epsilon = \epsilon \cdot u = u$ , es decir, la concatenación de cualquier cadena  $u$  con la cadena vacía, a izquierda o derecha, es igual a  $u$ .
- 2 Si  $u = a_1 a_2 \dots a_n$ ,  $v = b_1 b_2 \dots b_m$ , entonces

$$u \cdot v = a_1 a_2 \dots a_n b_1 b_2 \dots b_m$$

Es decir,  $u \cdot v$  es la cadena formada de escribir los símbolos de  $u$  y a continuación los símbolos de  $v$ .



Dada  $w \in \Sigma^*$  y  $n \in \mathbb{N}$ , se define  $w^n$  de la siguiente forma

$$w^n = \begin{cases} \epsilon & \text{si } n = 0 \\ \underbrace{uu \dots u}_{n-\text{veces}} & \text{si } n \geq 1 \end{cases}$$

## Potencia de una cadena de manera recursiva

**La potencia de una cadena** se define como  $w \in \Sigma^*$  para  $n \in \mathbb{N}$

$$w^n = \begin{cases} \epsilon, & \text{si } n = 0 \\ ww^{n-1}, & \text{si } n > 0 \end{cases}$$

Ejemplo. Sea una cadena  $w = acc$  sobre  $\Sigma = \{a, c\}$  entonces podemos obtener  $w^3 = ww^2 = wwww^0 = accaccacc\epsilon = (acc)^3$

# Inversa de una cadena

## Longitud de una cadena

La longitud de una cadena  $w \in \Sigma^*$  se denota  $|w|$  y se define como el número de símbolos de  $w$  (contando los símbolos repetidos), es decir:

$$|w| = \begin{cases} 0, & \text{si } w = \varepsilon \\ n, & \text{si } w = a_1 a_2 \dots a_n \end{cases}$$

$$|aba| = 3, |baaa| = 4$$

## Reflexión o inversa de una cadena

La reflexión o inversa de una cadena  $w \in \Sigma^*$  se denota como  $w^l$  y se define así:

$$w^l = \begin{cases} \varepsilon, & \text{si } w = \varepsilon \\ a_n \dots a_2 a_1, & \text{si } w = a_1 a_2 \dots a_n \end{cases}$$

## Inversa de una cadena de manera recursiva

**La Inversa de una cadena** Sea  $u \in \Sigma^*$  entonces  $u^{-1}$  es la inversa.

$$w' = \begin{cases} w & \text{si } w = \varepsilon \\ y'a & \text{si } w = ay, a \in \Sigma, y \in \Sigma^* \end{cases}$$

- Sea  $x = \text{'able'}$  entonces obtener  $x'$

$$\begin{aligned} x' &= (\text{able})' = (\text{ble})'a \\ &= (\text{le})'ba \\ &= (\text{e})'lba \\ &= (\varepsilon)'elba \\ &= \varepsilon elba \\ &= elba \end{aligned}$$

- Sea la concatenación de las cadenas “ab” y “cd” que forma “abcd” sobre un alfabeto. Sabemos que  $(abcd)' = dcba$ , por tanto  $dcba = (cd)'(ab)'$ . Por lo tanto, si  $w$  e  $y$  son cadenas y si  $x = wy$ , entonces  $x' = (wy)' = y'w'$
- En general,  $(x')' = x$ , para demostrar, suponga que  $x = a_1 a_2 \dots a_n$ .

## Cadena

**Definición formal:** Una cadena  $v$  es una subcadena o subpalabra de  $u$  si existen  $x, y$  tales que  $u = xvy$ . Nótese que  $x$  o  $y$  pueden ser  $\epsilon$  y por lo tanto, la cadena vacía es una subcadena de cualquier cadena.

- Un *prefijo* de  $u$  es una cadena  $v$  tal que  $u = vw$  para alguna cadena  $w \in \Sigma^*$ . Se dice que  $v$  es un **prefijo propio** si  $v \neq u$ .
- Un *sufijo* de  $u$  es una cadena  $v$  tal que  $u = wv$  para alguna cadena  $w \in \Sigma^*$ . Se dice que  $v$  es un **sufijo propio** si  $v \neq u$ .

perro  $\rightarrow$  per  
           $\searrow$  ro

# Ejemplo de cadenas que son sufijos y prefijos

Sea  $\Sigma = \{a, b, c, d\}$  y  $u = bcbaadb$

*Prefijos de  $u$*

$\epsilon$

$b$

$bc$

$bcb$

$bcba$

$bcbaa$

$bcbaad$

$bcbaadb$

*Sufijos de  $u$*

$\epsilon$

$b$

$db$

$adb$

$aadb$

$baadb$

$cbaadb$

$bcbaadb$

## Operación binaria

Una **operación binaria** en un conjunto  $A$  es una función  $f : A \times A \rightarrow A$ , esta deberá satisfacer las siguientes propiedades:

- 1 La operación binaria deberá estar definida para cada par ordenado de  $A$ , es decir,  $f$  asigna a **UN** elemento  $f(a, b)$  de  $A$  a cada par ordenado  $(a, b)$  de elementos de  $A$ .
  - 2 Como una operación binaria es una función, sólo un elemento de  $A$  se asigna a cada par  $(a, b)$ .
- 
- Sea  $A = \mathbb{Z}$ , se define  $a * b$  como  $a + b$ . Entonces,  $*$  es una operación binaria en  $\mathbb{Z}$ .
  - Sea  $A = \mathbb{Z}^+$ , se define  $a * b$  como  $a - b$ . Entonces,  $*$  no es una operación binaria ya que no asigna un elemento de  $A$  a cualquier par ordenado de elementos de  $A$ .

# Concatenación de cadenas como una operación binaria

## Concatenación

La **operación de la concatenación**  $\cdot$  es una **operación binaria** entre cadenas de un alfabeto  $\Sigma$ , esto es:

$$\cdot : \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$$

Sean  $u, v \in \Sigma^*$  y se denota por  $u \cdot v$  o simplemente  $uv$ .

$$|uv| = |u| + |v|$$

- Dado el alfabeto  $\Sigma$  y dos cadena  $w, u \in \Sigma^*$ 
  - Entonces  $w \cdot \epsilon = \epsilon \cdot w = w$ .
  - Si  $u = a_1 a_2 a_3 \dots a_n$ ,  $w = b_1 b_2 b_3 \dots b_m$ , entonces,

$$u \cdot w = a_1 a_2 a_3 \dots a_n b_1 b_2 b_3 \dots b_m$$

Por tanto  $|u \cdot w| = n + m$

- La concatenación de cadenas es asociativa. Es decir, si  $u, v, w \in \Sigma^*$ , entonces:

$$(uv)w = u(vw)$$

## Semigrupo

Sea  $(\Sigma^*, \cdot)$  es un **semigrupo** el cual es un conjunto no vacío  $\Sigma^*$  junto con una operación binaria asociativa definida en  $\Sigma^*$ .

- El conjunto  $P(S)$ , donde  $S$  es un conjunto, junto con la operación de la unión  $(P(S), \cup)$  es un semigrupo y es también un semigrupo conmutativo.

$$* : P(S) \times P(S) \rightarrow P(S)$$

Sea  $S = \{a, b\}$  entonces  $\{a, b\} \cup (\emptyset \cup \{b\}) = (\{a, b\} \cup \emptyset) \cup \{b\}$

- El semigrupo  $(\Sigma^*, \cdot)$  no es un semigrupo conmutativo porque para  $u, w \in \Sigma^*$  no se cumple que  $u \cdot w = w \cdot u$ .
- Sea  $w = ac$ ,  $w_1 = ab$  y  $w_2 = bb$  tal que  $w, w_1, w_2 \in \Sigma^*$  entonces

$$\begin{aligned}w(w_1 w_2) &= (ww_1)w_2 \\ac(abbb) &= (acab)bb \\acabbb &= acabbb\end{aligned}$$



## Monoide

Un **monoide** es un semigrupo  $(S, *)$  que tiene idéntico.

- El semigrupo  $P(S)$  con la operación de la unión tiene como idéntico a  $\emptyset$  ya que

$$\emptyset * A = \emptyset \cup A = A = A \cup \emptyset$$

- Sea  $(\Sigma^*, \cdot, \epsilon)$  un **monoide** con las siguientes propiedades:
  - 1 Es una operación binaria, es decir la concatenación es cerrada.  $\forall x, y \in \Sigma^*$ , entonces  $x \cdot y \in \Sigma^*$ .
  - 2 La concatenación es un semigrupo  $(\Sigma^*, \cdot)$  y por tanto  $\cdot$  es asociativa  $\forall x, y, z \in \Sigma^*$ ,  $(xy)z = x(yz)$
  - 3 La cadena vacía  $\epsilon$  es la idéntica para la concatenación:  $\forall x \in \Sigma^*$ ,  $\epsilon \cdot x = x \cdot \epsilon = x$

## Lenguaje

Un *lenguaje* es un conjunto de palabras o cadenas. Un lenguaje  $L$  sobre un alfabeto  $\Sigma$  es un subconjunto de  $\Sigma^*$  y si  $L = \Sigma^*$  es el lenguaje de todas las cadenas sobre  $\Sigma$ .

- Sea  $L = \emptyset$  el lenguaje vacío
- $\emptyset \subseteq L \subseteq \Sigma^*$
- $\Sigma = \{a, b, c\}$ .  $L = \{a, aba, aca\}$
- $\Sigma = \{a, b, c\}$ .  $L = \{a, aa, aaa\} = \{a^n : n \geq 1\}$
- $\Sigma = \{a, b, c\}$ .  $L = \{\epsilon, aa, aba, ab^2a, ab^3a\} = \{ab^n a : n \geq 0\} \cup \{\epsilon\}$
- $\Sigma = \{a, b, c\}$ .  $L = \{w \in \Sigma^* : w \text{ no contiene el símbolo } c\}$ . Por ejemplo,  $abbaab \in L$  pero  $abbcaa \notin L$ .
- Sobre  $\Sigma = \{0, 1, 2\}$  el lenguaje de las cadenas que tienen igual número de ceros, unos y dos's en cualquier orden.

- Operaciones entre lenguajes; Sean  $A, B$  lenguajes sobre  $\Sigma$  entonces  $A \cap B, A \cup B, A - B$  operaciones de conjuntos.
- Las operaciones lingüísticas son la concatenación, potencia, inverso y clausura.
- Sean  $A, B$  lenguajes sobre  $\Sigma$  entonces,

$$A \cup B = \{x | x : x \in A \text{ o } x \in B\}$$

$$\{a\} \cup \{b\} = \{a, b\}$$

$$\{a, ab\} \cup \{ab, aab, aaabb\} = \{a, ab, aab, aaabb\}$$

- Sean  $A, B$  lenguajes sobre  $\Sigma$  entonces,

$$A \cap B = \{x \mid x \in A \text{ y } x \in B\}$$

$$\{a, ab\} \cap \{ab, aab\} = \{ab\}$$

$$\{a, aab\} \cap \{a, ab, aab, aaabb\} = \{a, aab\}$$

$$\{\epsilon\} \cap \{a, ab, aab, aaabb\} = \emptyset$$

- Complemento en  $\Sigma^*$ :

$$\sim A = \{x \in \Sigma^* \mid x \notin A\}$$

$$\sim A = \Sigma^* - A$$

$A = \{\text{Cadenas de longitud par}\}$  sobre  $\Sigma = \{a, b\}$ , entonces

$\sim A = \{\text{cadenas de longitud impar}\}.$

- Sean  $A, B$  lenguajes sobre  $\Sigma$  entonces,

$$A - B = \{x \mid x \in A \text{ y } x \notin B\}$$

Sea  $B$ : El lenguaje de todas las cadenas de ceros de cualquier longitud.

Entonces:

Sea  $A = \{0, 1\}^*$  y  $B = \{0\}^*$  entonces

$$A - B = \{0, 1\}^* - \{0\}^* = 0^*1(0 \cup 1)^*$$

$A - B$  es el lenguaje de todas las cadenas de unos y ceros con al menos un uno.

## Lenguaje Universal

Si  $\Sigma \neq \emptyset$ , entonces  $\Sigma^*$  es el conjunto de todas las cadenas sobre  $\Sigma$ . Se le llama **lenguaje universal**.

- $\Sigma^*$  es un conjunto infinito de cadenas de longitud finita sobre  $\Sigma$ .

## Teorema

*Sean  $A$  y  $B$  dos lenguajes sobre el alfabeto  $\Sigma$ . Entonces  $A = B$  si y sólo si  $A \subseteq B$  y  $B \subseteq A$ .*

$\Rightarrow$ ) Suponiendo que  $A = B$ , entonces si  $x \in A$ , como  $A = B$  entonces  $x \in B$  por tanto  $A \subseteq B$  de la misma forma si  $x \in B$  entonces como  $A = B$  entonces  $x \in A$  por lo tanto  $B \subseteq A$ .

$\Leftarrow$ ) Se demuestra que si  $A \subseteq B$  y  $B \subseteq A$  entonces  $A = B$ .

- Sea el lenguaje del conjunto de cadenas con igual número de ceros y unos.

$$L_1 = \{\epsilon, 01, 10, 0011, 0101, 1001, 000111, \dots\}$$

y sea

↘

$$L = \{a^n b^n : n \geq 0\} \subset L_1 \subset \{0, 1\}^*$$

- La concatenación de lenguajes de dos lenguajes  $A$  y  $B$  sobre  $\Sigma$ , notada por  $A.B$  o simplemente  $AB$ .
- $AB = \{uv : u \in A, v \in B\}$
- $A \cdot \emptyset = \emptyset \cdot A = \emptyset$

$$A \cdot \emptyset = \{uw : u \in A, w \in \emptyset\} = \emptyset$$

$\emptyset \neq \{\epsilon\}$

↑                  ↑

■  $A \cdot \{\varepsilon\} = \{\varepsilon\} \cdot A = A$

$$A \cdot \{\epsilon\} = \{uw : u \in A, w \in \{\epsilon\}\} = \{u : u \in A\} = A$$

- Las propiedad distributiva generalizada de la concatenación con respecto a la unión.

$$A \cdot \bigcup_{i \in I} B_i = \bigcup_{i \in I} (A \cdot B_i)$$

$$x \in A \cdot \bigcup_{i \in I} B_i \iff x = u \cdot v, u \in A, v \in \bigcup_{i \in I} B_i$$

$$\iff x = u \cdot v, u \in A, v \in B_j,$$

$$\exists j \in I$$

$$\iff x \in A \cdot B_j, \exists j \in I$$

$$\iff x \in \bigcup_{i \in I} (A \cdot B_i)$$



- Ejemplo. Sean  $A = \{ab\}$ ,  $B_1 = \{a, b\}$ , y  $B_2 = \{abb, b\}$

$$A \cdot \bigcup_{i \in I} B_i = \bigcup_{i \in I} (A \cdot B_i)$$

$$A \cdot \bigcup_{i \in I=2} B_i = A \cdot (B_1 \cup B_2)$$

$$A \cdot \bigcup_{i \in I=2} B_i = \{ab\} \cdot (\{a, b\} \cup \{abb, b\})$$

$$\{ab\} \cdot (\{a, b\} \cup \{abb, b\}) = (\{ab\} \cdot \{a, b\}) \cup (\{ab\} \cdot \{abb, b\})$$

- De igual forma se puede demostrar que:

$$\left( \bigcup_{i \in I} B_i \right) \cdot A = \bigcup_{i \in I} (B_i \cdot A)$$

La concatenación no es distributiva con respecto a la intersección, es decir, no se cumple que  $A \cdot (B \cap C) \neq A \cdot B \cap A \cdot C$ . Contraejemplo: Sea  $A = \{a, \epsilon\}$ ,  $B = \{\epsilon\}$ ,  $C = \{a\}$  se tiene:

$$A \cdot \underbrace{(B \cap C)}_{\emptyset} = \{a, \epsilon\} \cdot \emptyset = \emptyset$$

Por otro lado,

$$\begin{aligned} A \cdot B \cap A \cdot C &= \{a, \epsilon\} \cdot \{\epsilon\} \cap \{a, \epsilon\} \cdot \{a\} \\ &= \underbrace{\{a, \epsilon\}} \cap \underbrace{\{a^2, a\}} = a \end{aligned}$$

## Potencia del lenguaje

**Potencia del lenguaje** Dado un lenguaje  $A$  sobre  $\Sigma$  y  $(A \subseteq \Sigma^*)$  y  $n \in \mathbb{N}$ , se define

$$A^n = \begin{cases} \{\epsilon\}, & \text{si } n = 0 \\ A \cdot A^{n-1}, & \text{si } n \geq 1 \end{cases}$$

**Ejemplo.** Sea  $A = \{ab\}$  sobre un alfabeto  $\Sigma = \{a, b\}$ , entonces:

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

$$A^1 = A = \{ab\}$$

$$A^2 = A \cdot A^1 = \{abab\}$$

$$A^3 = A \cdot A^2 = \{ababab\}$$

$$B = \{a, b\} \quad B^2 = \{aa, ab, ba, bb\}$$

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

$$B^1 = \{a, b\}$$

## Def. formal de Cerradura de Kleene

La cerradura de Kleene de un lenguaje  $A \subseteq \Sigma^*$  es la unión de las potencias: se denota por  $A^*$

$$A^* = \bigcup_{i \geq 0} A^i = A^0 \cup A^1 \cup A^2 \cup \dots \cup A^n$$

$\{a\}^* = \{\epsilon, a, aa, \dots\}$

- Observación:  $A^*$  se puede describir de la siguiente manera:

$$A^* = \{u_1 u_2 \dots u_n : u_i \in A, n \geq 0\}$$

Es el conjunto de todas las concatenaciones de la cadena  $A$ , incluyendo  $\epsilon$

- la cerradura positiva se denota por  $A^+$

$$A^+ = \bigcup_{i \geq 1} A^i = A^1 \cup A^2 \cup A^3 \cup \dots \cup A^n$$

$$A = \{ a, ab \}$$

$$A^*A \rightarrow \{ \overbrace{a, ab}^{A^2}, aa, aab, aba, \underbrace{abab}_{A^2}, \dots \}$$

- Observe que  $A^* = A^+ \cup \{\epsilon\}$  y  $A^* = A^+$  si y solamente si  $\epsilon \in A$
- $A^+ = \underbrace{A^* \cdot A}_{A^+} = A \cdot A^*$

$$\begin{aligned} A \cdot A^* &= A \cdot (A^0 \cup A^1 \cup A^2 \cup \dots) \\ &= (A^1 \cup A^2 \cup A^3 \cup \dots) \\ &= A^+ \end{aligned}$$

Se demuestra lo mismo que  $A^+ = A^* \cdot A$

$$A^* A^* \rightarrow \underbrace{\epsilon A^*}_{\subseteq A^*} \cup \underbrace{A \cdot A^*}_{\sim} \cup \underbrace{A^2 A^*}_{\sim} \cup \underbrace{A^3 A^*}_{\sim}$$

$$A^* \cdot A^* = A^*$$

- 1  $\Rightarrow$ ), Sea un  $x \in A^* \cdot A^*$ , entonces  $x = u \cdot v$ , con  $u \in A^*$  y  $v \in A^*$ . Por tanto  $x = u \cdot v$ , con  $u = u_1 u_2 \dots u_n$ ,  $u_i \in A$ ,  $n \geq 0$  y  $v = v_1 v_2 \dots v_m$ ,  $v_i \in A$ ,  $m \geq 0$ . De donde

$$x = u \cdot v = u_1 u_2 \dots u_n \cdot v_1 v_2 \dots v_m$$

con  $u_i \in A$ ,  $v_i \in A$ , por lo tanto  $x$ , es una concatenación de  $n + m$  cadenas de  $A$ , así que  $x \in A^*$ .

- 2  $\Leftarrow$ ) Recíprocamente, si  $x \in A^*$ , entonces  $x = x \cdot \epsilon \in A^* \cdot A^*$ . Esto prueba la igualdad de los conjuntos  $A^* \cdot A^*$  y  $A^*$ .

- $(A^*)^n = A^*$ , para todo  $n \geq 1$

- $(A^*)^* = A^*$

- $A^+ \cdot A^+ \subseteq A^+$

$$\epsilon^n = \epsilon$$

Contraejemplo de  $A^+ \cdot A^+ = A^+$ . Sea  $\Sigma = \{a, b\}$ ,  $A = \{a\}$  se tiene que

$$\begin{aligned} A^+ &= (A^1 \cup A^2 \cup A^3 \cup \dots) \\ &= \{a\} \cup \{aa\} \cup \{aaa\} \cup \dots \\ &= \{a^n : n \geq 1\} \end{aligned}$$

Por otro lado,

$$\begin{aligned} A^+ \cdot A^+ &= \{a, a^2, a^3, \dots\} \cdot \{a, a^2, a^3, \dots\} \\ &= \{a^2, a^3, \dots\} \\ &= \{a^n : n \geq 2\} \end{aligned}$$

■  $(A^*)^+ = A^*$

$$\begin{aligned}(A^*)^+ &= (A^*)^1 \cup (A^*)^2 \cup (A^*)^3 \cup \dots \\ &= A^* \cup A^* \cup A^* \dots \\ &= A^*\end{aligned}$$

■  $(A^+)^* = A^*$

$$\begin{aligned}(A^+)^* &= (A^+)^0 \cup (A^+)^1 \cup (A^+)^2 \cup \dots \\ &= \{\epsilon\} \cup A^+ \cup A^+ A^+ \cup \dots \\ &= A^* \cup (\text{conjuntos contenidos en } A^+) \\ &= A^*\end{aligned}$$

■  $(A^+)^+ = A^+$

$$\begin{aligned}(A^+)^+ &= (A^+)^1 \cup (A^+)^2 \cup (A^+)^3 \cup \dots \\ &= (A^+)^1 \cup (\text{conjuntos contenidos en } A^+) \\ &= A^+\end{aligned}$$



Operaciones claves en los lenguajes:

■  $A^* \subseteq \Sigma^*$       $A^+ \subseteq \Sigma^+$

■  $A^+ \subseteq A^*$

■  $\{\varepsilon\}^* = \{\varepsilon\} = \{\varepsilon\}^+$

■  $\emptyset^0 = \{\varepsilon\}$

■  $\emptyset^n = \emptyset, n \geq 1$

■  $\emptyset^* = \{\varepsilon\}$       $\emptyset^+ = \emptyset$

$\emptyset^0 = \{\varepsilon\}$

## Inverso de un lenguaje

Sea  $A$  sobre  $\Sigma$ , se define  $A'$  como:

$$A' = \{u' : u \in A\}$$

Sean  $A$  y  $B$  lenguajes sobre  $\Sigma$  tal que  $(A, B \subseteq \Sigma^*)$

■  $\underbrace{(A \cdot B)}' = \underbrace{B'} \cdot \underbrace{A'}$

$$\begin{aligned}x \in (A \cdot B)' &\iff x = u', \text{ donde, } u \in A \cdot B \\&\iff x = u', \text{ donde, } u = vw, v \in A, w \in B \\&\iff x = (vw)', \text{ donde, } v \in A, w \in B \\&\iff x = w'v', \text{ donde, } v \in A, w \in B \\&\iff x = B'A'\end{aligned}$$

Sean  $A$  y  $B$  lenguajes sobre  $\Sigma$  tal que  $(A, B \subseteq \Sigma^*)$

- $(A \cup B)' = A' \cup B'$
- $(A \cap B)' = A' \cap B'$
- $(A')' = A$
- $(A^*)' = (A')^*$
- $(A^+)' = (A')^+$

# Lenguajes regulares

Los lenguajes regulares sobre un alfabeto  $\Sigma$  se definen recursivamente como:

- $\emptyset$ ,  $\{\varepsilon\}$  y  $\{a\}$ ,  $a \in \Sigma$  son lenguajes regulares.
- si  $A$  y  $B$  son lenguajes regulares, también lo son:

$A \cup B$  (Unión)

$A \cdot B$  (Concatenación)

$A^*$  (Cerradura de Kleene)

Ejemplo 1. Dado  $\Sigma = \{a, b\}$  el lenguaje  $A$  de todas las palabras que tienen exactamente una  $a$ :  $A = \{b\}^* \cdot \{a\} \cdot \{b\}^*$   $\leftarrow a, a.b$

Ejemplo 2. Lenguaje de todas las cadenas que comienzan con  $b$ :

$B = \{b\} \cdot \{(a \cup b)\}^*$

Ejemplo 3. Lenguaje de todas las cadenas que contienen la cadena  $ba$ :

$C = \{(a \cup b)\}^* \cdot \{ba\} \cdot \{(a \cup b)\}^*$

Ejemplo 4: Lenguaje de todas las cadenas que inician en  $a$ , contienen  $ba$  y finalizan en  $bab$   $a\{(a \cup b)\}^*ba\{(a \cup b)\}^*bab$

## Teorema

Si  $L$ ,  $L_1$  y  $L_2$  son lenguajes regulares sobre un alfabeto  $\Sigma$ , también lo son:

1  $L_1 \cup L_2$

2  $L_1 L_2$

3  $L^+$

4  $\bar{L} = \Sigma^* - L$

5  $L^*$

6  $L_1 \cap L_2$

7  $L_1 - L_2$

8  $L_1 \triangle L_2 \rightarrow \forall x \in L_1 \vee \forall x \in L_2 \quad x \notin L_1 \vee L_2$

## Observación

Un sublenguaje (subconjunto) de un lenguaje regular no es necesariamente regular, es decir, la familia de los lenguajes regulares no es cerrada para subconjuntos.

$\{a^n b^n\}$     qqq

## Observación

- *Un lenguaje regular puede contener sublenguajes No-regulares. Sea  $L = \{a^n b^n\}$  es un sublenguaje del lenguaje regular  $a^* b^*$*
- *Todo lenguaje finito es regular y la unión finita de lenguajes regulares es regular.*
- *La unión infinita de lenguajes no necesariamente es regular.*

$$L = \{a^n b^n : n \geq 1\} = \bigcup_{i \geq 1} \{a^i b^i\}$$

*Donde cada  $\{a^i b^i\}$  regular, pero  $L$  No lo es.*

# Definición formal de expresiones regulares

Las expresiones regulares sobre un alfabeto  $\Sigma$  se definen recursivamente como:

- $\emptyset$ ,  $\epsilon$  y  $a$ ,  $a \in \Sigma$  son expresiones regulares.
- si  $A$  y  $B$  son expresiones regulares, también lo son:

$A \cup B$  (Unión)

$A \cdot B$  (Concatenación)

$A^*$  (Cerradura de Kleene)

- Son expresiones regulares  $aab^*$ ,  $ab^+$ ,  $(aaba^*)^+$
- Sea el conjunto  $\{\epsilon, aa, aba, ab^2a, ab^3a, ab^4a, \dots\}$  entonces  $\{\epsilon\} \cup ab^*a$  es una expresión regular.
- Expresión regular de todas las cadenas impares sobre  $\Sigma = \{a, b\}$

$$a(aa \cup ab \cup ba \cup bb)^* \cup b(aa \cup ab \cup ba \cup bb)^*$$

## Teorema

*Sean  $r, s$  y  $t$  expresiones regulares sobre  $\Sigma$ , entonces:*

1.  $r \cup s = s \cup r$
2.  $r \cup \emptyset = r = \emptyset \cup r$
3.  $r \cup r = r$
4.  $(r \cup s) \cup t = r \cup (s \cup t)$
5.  $r\varepsilon = r = \varepsilon r$
6.  $r\emptyset = \emptyset = \emptyset r$
7.  $(rs)t = r(st)$
8.  $r(s \cup t) = rs \cup rt$  y  $(r \cup s)t = rt \cup st$
9.  $r^* = r^{**} = r^*r^* = (\varepsilon \cup r)^* = r^*(r \cup \varepsilon) = (r \cup \varepsilon)r^* = \varepsilon \cup rr^*$
10.  $(r \cup s)^* = (r^* \cup s^*)^* = (r^*s^*)^* = (r^*s)^*r^* = r^*(sr^*)^*$
11.  $r(sr)^* = (rs)^*r$
12.  $(r^*s)^* = \varepsilon \cup (r \cup s)^*s$
13.  $(rs^*)^* = \varepsilon \cup r(r \cup s)^*$
14.  $s(r \cup \varepsilon)^*(r \cup \varepsilon) \cup s = sr^*$
15.  $rr^* = r^*r$



**Ejemplo 1.** Muestre que si  $r = s^*t$  implica que  $r = sr \cup t$

$$\begin{aligned}r = s^*t &= (\varepsilon \cup s^+)t \text{ ya que } s^* = \varepsilon \cup s^+ \\&= (\varepsilon \cup ss^*)t \\&= \varepsilon t \cup s \underbrace{s^*t}_r \\&= t \cup sr \\&= sr \cup t\end{aligned}$$

**Ejemplo 2.** Probar que  $(b \cup aa^*b) \cup (b \cup aa^*b)(a \cup ba^*b)^*(a \cup ba^*b)$  y  $a^*b(a \cup ba^*b)^*$  son equivalentes.

**Ejemplo 2.** Probar que  $(b \cup aa^*b) \cup (b \cup aa^*b)(a \cup ba^*b)^*(a \cup ba^*b)$  y  $a^*b(a \cup ba^*b)^*$  son equivalentes.

### Teorema

Sean  $r, s$  y  $t$  expresiones regulares sobre  $\Sigma$ , entonces:

1.  $r \cup s = s \cup r$
2.  $r \cup \emptyset = r = \emptyset \cup r$
3.  $r \cup r = r$
4.  $(r \cup s) \cup t = r \cup (s \cup t)$
5.  $r\varepsilon = r = \varepsilon r$
6.  $r\emptyset = \emptyset = \emptyset r$
7.  $(rs)t = r(st)$
8.  $r(s \cup t) = rs \cup rt$  y  $(r \cup s)t = rt \cup st$
9.  $r^* = r^{**} = r^*r^* = (\varepsilon \cup r)^* = r^*(r \cup \varepsilon) = (r \cup \varepsilon)r^* = \varepsilon \cup rr^*$
10.  $(r \cup s)^* = (r^* \cup s^*)^* = (r^*s^*)^* = (r^*s)^*r^* = r^*(sr^*)^*$
11.  $r(sr^*)^* = (rs)^*r$
12.  $(r^*s)^* = \varepsilon \cup (r \cup s)^*s$
13.  $(rs^*)^* = \varepsilon \cup r(r \cup s)^*$
14.  $s(r \cup \varepsilon)^*(r \cup \varepsilon) \cup s = sr^*$
15.  $rr^* = r^*r$

$$a^*b = r$$

$$ba^*b = s$$

$$\boxed{r(a \cup s)^*}$$

$$(b \cup ar) \cup (b \cup ar)(a \cup s)^*(a \cup s)$$

$$(b \cup ar)(\varepsilon \cup (a \cup s)^*(a \cup s))$$

$$(b \cup ar)(\varepsilon \cup (a \cup s)^+)$$

$$(b \cup ar) \text{ (a \cup s)^*}$$

$$(b \cup aa^*b) \text{ (a \cup s)^*}$$

$$(\varepsilon \cup aa^*)b(a \cup s)^* \rightarrow (\varepsilon \cup a^+)b(a \cup s)^* \rightarrow \boxed{a^*b(a \cup ba^*b)^*}$$

**Ejemplo 3.** ¿Las siguientes expresiones regulares representan el mismo lenguaje?

$$(a^*b)^* \quad \text{y} \quad \epsilon \cup (a \cup b)^*b$$

**Ejemplo 4.** Demostrar que  $r(\underline{sr})^* = (\underline{rs})^*r$

$\Rightarrow$ ) Sea  $w \in r(sr)^*$ , entonces

$$w = r_0(s_1r_1)(s_2r_2) \dots (s_nr_n), \text{ para } n \geq 0$$

$$\begin{aligned} & \epsilon \cup sr \cup sr sr \dots \\ & r \cup r sr \cup r.sr sr \dots \end{aligned}$$

$$w = r_0(s_1r_1)(s_2r_2) \dots (s_nr_n)$$

$$w = (r_0s_1)(r_1s_2)(r_2s_3) \dots (r_{n-1}s_n)\underline{r_n}$$

Por lo tanto,  $r(sr)^* \subseteq (rs)^*r$

$\Leftarrow$ )

Sea  $w \in (rs)^*r$ , entonces

$$w = (r_0s_0)(r_1s_1) \dots (r_{n-1}s_{n-1})r_n, \text{ para } n \geq 0$$

$$\begin{aligned} & \epsilon \cup rs \cup rs rs \dots \\ & r \cup r sr \cup r sr sr \dots \end{aligned}$$

# Encontrar las expresiones regulares de los siguientes lenguajes

**Ejemplo 5.**  $\Sigma = \{a, b\}$  Lenguaje de todas las palabras que comienzan con b y terminan con a.

$$b(a \cup b)^* a$$

**Ejemplo 6.**  $\Sigma = \{a, b\}$  Lenguaje de todas las palabras que tienen exactamente dos a's

$$b^* \underline{a} b^* \underline{a} b^*$$

**Ejemplo 7.**  $\Sigma = \{a, b\}$  Lenguaje de todas las palabras que tienen un número par de símbolos (palabras de longitud par)

$$(aa \cup ab \cup ba \cup bb)^*$$

aaab

**Ejemplo 8.**  $\Sigma = \{a, b\}$  Lenguaje de todas las palabras que tienen un número impar de símbolos (palabras de longitud impar)

$$a(aa \cup ab \cup ba \cup bb)^* \cup b(aa \cup ab \cup ba \cup bb)^*$$

**Ejemplo 9.**  $\Sigma = \{a, b\}$  Lenguaje de todas las palabras que tienen un número par de a's.

$$b^* (ab^*a)^* b^*$$

b a b b a a b b b a b b b

**Ejemplo 10.** Sobre  $\Sigma = \{0, 1\}$  lenguaje de todas las cadenas que tienen exactamente dos ceros:

$$1^*01^*01^*$$

**Ejemplo 11.** Sobre  $\Sigma = \{0, 1\}$  lenguaje de todas las cadenas cuyo penúltimo símbolo, de izquierda a derecha, es un 0.

$$(0 \cup 1)^*0(0 \cup 1)$$

- Las expresiones regulares sirven para la construcción de analizadores léxicos.
- <http://regexpal.com/> es un testeador de expresiones regulares en java.

```
'[A-Z][a-z]*[ ][A-Z][A-Z]'
```

Representa palabras que comienzan por una letra mayúscula seguida de un espacio en blanco y de dos letras mayúsculas. Ejemplo, reconocería Ithaca NY. Por ejemplo, Palo Alto CA no la reconocería.

1. Dado el alfabeto  $E=\{a,b,c\}$  muestre:
  1. Tres cadenas que pertenecen al conjunto universal  $E^*$
  2. Dada  $A=\{aa,bb\}$  y  $B=\{ac,bb\}$  muestre el resultado  $A.B$  y  $A \cup B$
2. Dado el alfabeto  $E=\{0,1,2\}$  diseñe expresiones regulares para:
  1. Todas las cadenas que terminan en 000
  2. Todas las cadenas que contienen 011

$$\begin{array}{l|l}
 E^* = \{\epsilon, abc, aabc, c\} & (00100)^*000 \\
 A.B = \{aaac, aabb, bbac, bbbb\} & (00100)^*011(00100)^* \\
 A \cup B = \{aa, bb, ac\} &
 \end{array}$$



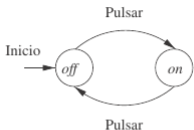
1 Lenguajes

2 Autómatas finitos

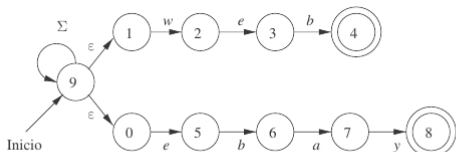
3 Gramáticas

4 Máquinas de Turing

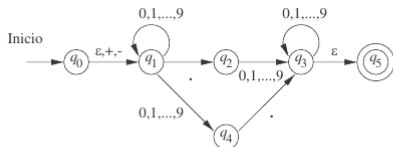
# Introducción a los autómatas finitos



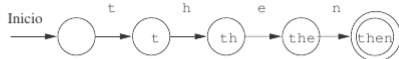
**A.F de un interruptor**



Uso de transiciones- $\epsilon$  para ayudar a reconocer palabras clave.



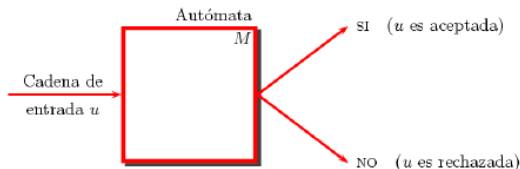
Un AFN- $\epsilon$  que acepta números decimales.



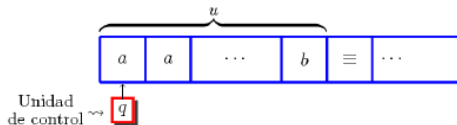
**Reconocimiento de la palabra then**

# Autómatas finitos

Son máquinas abstractas que procesan cadenas, las cuales son aceptadas o rechazadas.



El autómata posee **unidad de control** que inicialmente escanea o lee la casilla desde el extremo izquierdo de la cinta. Tiene unos estados o configuraciones internas.



# Función de transición



Sea un autómata  $M = (Q, \Sigma, q_0, T, \delta)$

$\delta$	$a$	$b$
$q_0$	$q_0$	$q_1$
$q_1$	$q_1$	$q_2$
$q_2$	$q_1$	$q_1$

$$\delta(q_0, a) = q_0$$

$$\delta(q_0, b) = q_1$$

$$\delta(q_1, a) = q_1$$

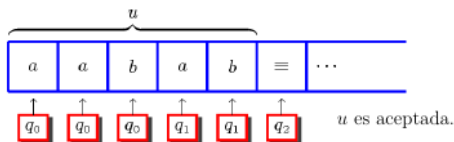
$$\delta(q_1, b) = q_2$$

$$\delta(q_2, a) = q_1$$

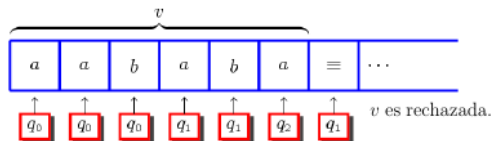
$$\delta(q_2, b) = q_1.$$

$F = \{q_0, q_2\}$ , estados de aceptación.

1.  $u = aabab.$

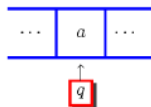


2.  $v = aababa.$



# Lenguaje aceptado por un autómata

Caso especial: la cadena  $\lambda$  es la cadena de entrada.



Dado un autómata  $M$ , el **lenguaje aceptado o reconocido** por  $M$  se denota  $L(M)$  y se define por

$$L(M) := \{u \in \Sigma^* : M \text{ termina el procesamiento de la cadena de entrada } u \text{ en un estado } q \in F\}.$$

# Autómatas finitos (FSAs: Finite State-Automata)

Los autómatas finitos se dividen en autómatas finitos deterministas (AFD) (es función) y en autómatas finitos no deterministas (AFN) (es una relación).

## Autómata finito determinista

Sea  $M = (\underline{Q}, \underline{\Sigma}, \underline{q_0}, \underline{T}, \underline{\delta})$  un AFD entonces:

- $\Sigma$ : es el alfabeto de entrada.
- $Q$ : es el conjunto de estados
- $q_0$ : Estado inicial
- $T$ : Conjunto de estados finales.
- $\delta : \underline{Q} \times \underline{\Sigma} \longrightarrow Q$  determina un único estado siguiente para el par  $\delta(q_i, \gamma)$  correspondiente al estado actual y la entrada.

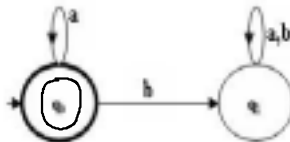
Un AFD puede ser representado por un grafo dirigido y etiquetado.

**Ejemplo 1.** Diseñar el AFD sobre  $\Sigma = \{a, b\}$  que reconozca el lenguaje  $L = a^* = \{\epsilon, a, a^2, a^3, \dots\}$

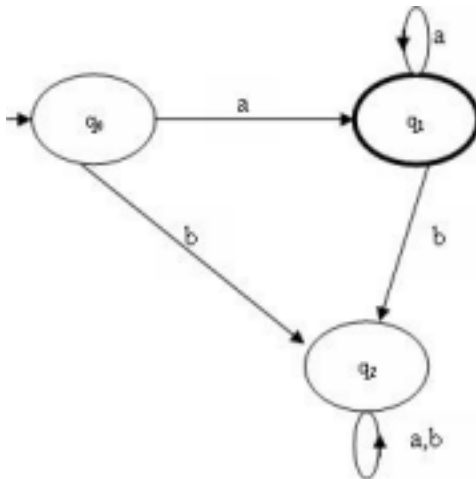
$\delta$	$a$	$b$
$q_0$	$q_0$	$q_1$
$q_1$	$q_1$	$q_1$

$$\delta(q_0, a) = q_0 \quad \delta(q_0, b) = q_1$$

$$\delta(q_1, a) = q_1 \quad \delta(q_1, b) = q_1$$



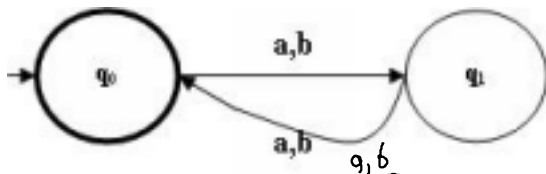
**Ejemplo 2.** Diseñar el AFD sobre  $\Sigma = \{a, b\}$  que reconozca el lenguaje  $L = a^+ = \{a, a^2, a^3, \dots\}$



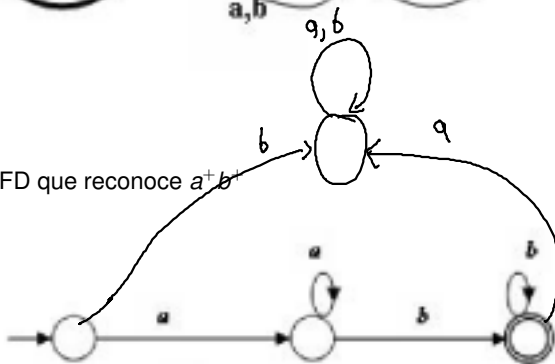


## Ejemplos autómatas finitos deterministas

**Ejemplo 3.** Diseñar el AFD sobre  $\Sigma = \{a, b\}$  que reconozca el lenguaje de todas las cadenas que tienen un número par de símbolos



**Ejemplo 4.** AFD que reconoce  $a^+b^+$



# Ejemplos autómatas finitos deterministas

**Ejemplo 5.** El diagrama y tabla de transición en cierta forma determinan si es un autómata finito determinista o no determinista.

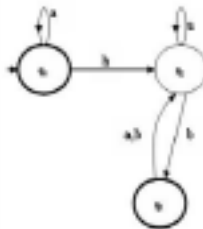
Sea  $\Sigma = \{a, b\}$ ,  $Q = \{q_0, q_1, q_2\}$

$q_0$  : estado inicial

$T = \{q_0, q_2\}$  estados finales o de aceptación.

	$a$	$b$
$q_0$	$q_0$	$q_1$
$q_1$	$q_1$	$q_2$
$q_2$	$q_1$	$q_2$

$$\begin{aligned}\delta(q_0, a) &= q_0 & \delta(q_0, b) &= q_1 \\ \delta(q_1, a) &= q_1 & \delta(q_1, b) &= q_2 \\ \delta(q_2, a) &= q_1 & \delta(q_2, b) &= q_2\end{aligned}$$



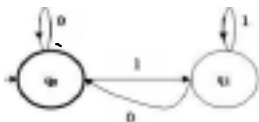
Es importante anotar que en la tabla de transición por cada pareja  $(q_i, \gamma)$  hay un sólo estado  $q_j$  por eso  $\delta$  es una función de transición.  
el lenguaje que reconoce este AFD es:

$$a^*(b(a + ba + bb)^*b) + a^*$$

Ahora como el estado inicial es un estado final este AFD reconoce  $\epsilon$

## Ejemplos autómatas finitos deterministas

**Ejemplo 6.** Diseñar el AF sobre  $\Sigma = \{0, 1\}$  que reconozca en binario el lenguaje de todos los múltiplos de 2.



**Binario Decimal**

0	0
10	2
100	4
110	6
1000	8
1010	10
1100	12
1110	14
⋮	⋮

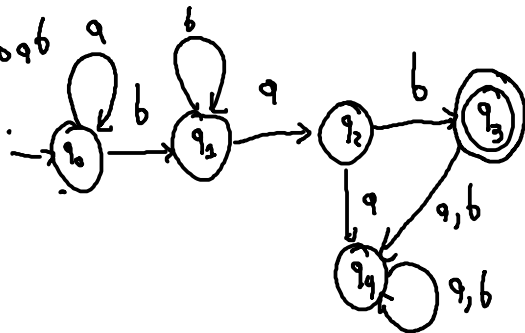
q.    0    1  
q<sub>0</sub>    q<sub>0</sub>    q<sub>1</sub>  
q<sub>1</sub>    q<sub>0</sub>    q<sub>1</sub>

1011110

$a^*b^+ab$

$\Sigma = \{a, b\}$

?



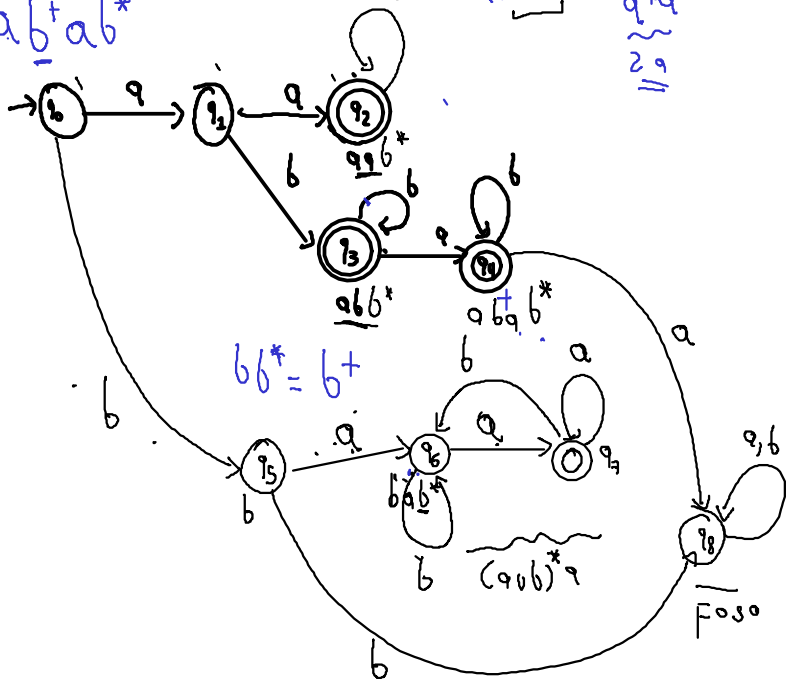
$q$	$a$	$b$
$q_0$	$q_0$	$q_1$
$q_1$	$q_2$	$q_1$
$q_2$	$q_4$	$q_3$
$q_3$	$q_4$	$q_4$
$q_4$	$q_4$	$q_4$

$$\underline{a}b^*(a \cup \underline{b})b^* \cup \underline{b}a^*(a \cup b)^*a \quad \Sigma = \{a, b\}$$

$\cdot aab^*$   
 $\cdot abb^*$   
 $a \underline{b^+} ab^*$

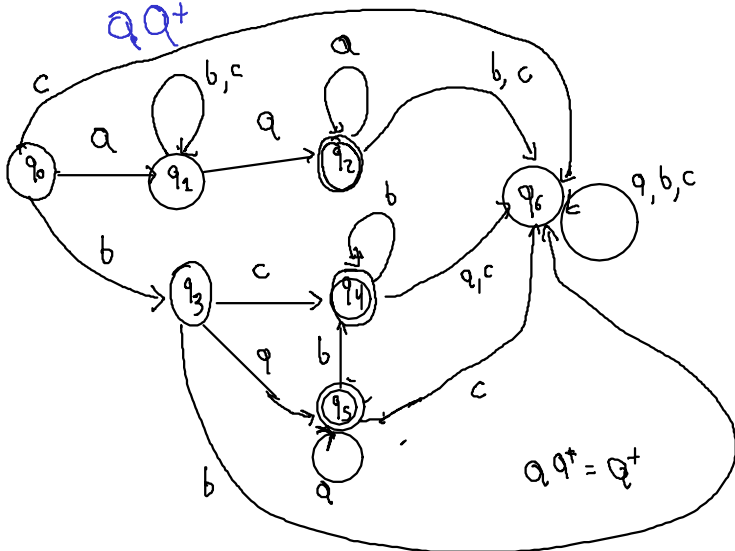
$b a a^+$   
 $b a (a \cup b)^+ a$

$a^+ a$   
 $\approx$   
 $2a$



$$a(\underline{buc})^*a^+ \cup b(\underline{ccu}a^+)b^*$$

$aa^+$



## Autómatas finitos No determinísticos

Sea  $M = (Q, \Sigma, q_0, T, \Delta)$  un AFN entonces:

- $\Sigma$ : es el alfabeto de entrada.
- $Q$ : es el conjunto de estados
- $q_0$ : Estado inicial
- $T$ : Conjunto de estados finales.
- $\Delta$ : es una relación tal que:

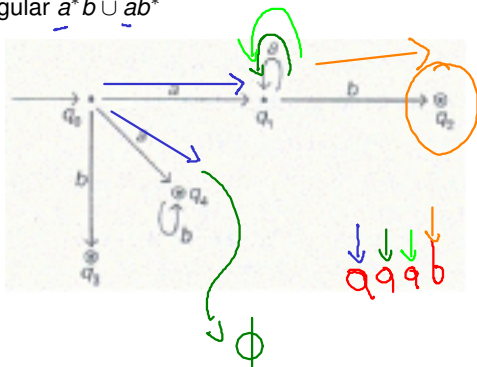
$$(Q \times \Sigma) \rightarrow 2^Q$$

Donde  $2^Q$  denota el conjunto potencia de  $Q$  o el conjunto de todos los subconjuntos de  $Q$ .

$$2^Q = \{A | A \subseteq Q\}$$

# Ejemplos Autómatas finitos No determinísticos

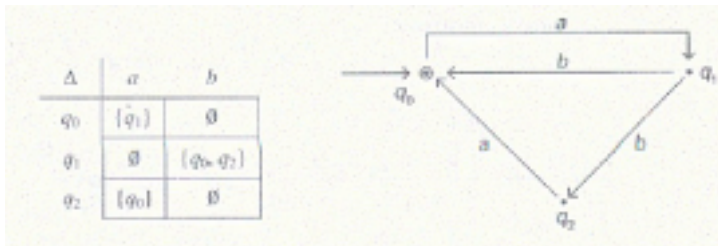
**Ejemplo 1.** Diseñar el AFN sobre  $\Sigma = \{a, b\}$  que reconozca el lenguaje regular  $\underline{a^*b} \cup \underline{ab^*}$



$\Delta$	$a$	$b$
$q_0$	$\{q_1, q_3\}$	$\{q_3\}$
$q_1$	$\{q_1\}$	$\{q_2\}$
$q_2$	$\emptyset$	$\emptyset$
$q_3$	$\emptyset$	$\emptyset$
$q_4$	$\emptyset$	$\{q_4\}$

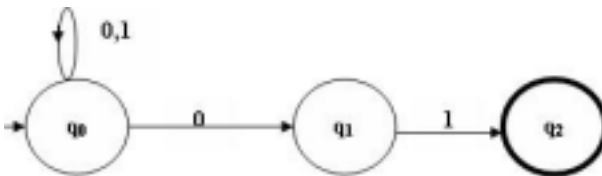


**Ejemplo 2.** Diseñar el AFN sobre  $\Sigma = \{a, b\}$  que reconozca el lenguaje  $(\underline{a}b \cup \underline{a}ba)^*$



# Ejemplos Autómatas finitos No determinísticos

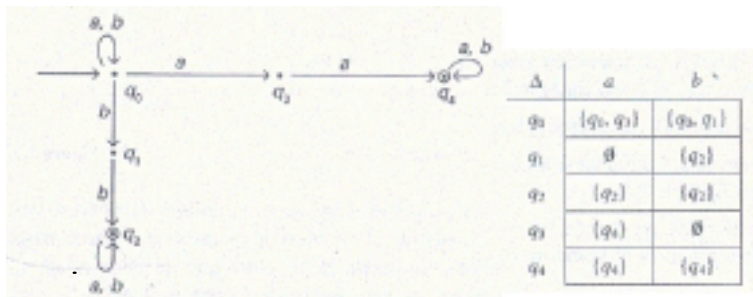
**Ejemplo 3.** Diseñar el AF sobre  $\Sigma = \{0, 1\}$  que reconozca el lenguaje de todas las cadenas que terminan en 01



$\Delta$	0	1
$q_0$	$\{q_0, q_1\}$	$q_0$
$q_1$	$\emptyset$	$q_2$
$q_2$	$\emptyset$	$\emptyset$

# Ejemplos Autómatas finitos No determinísticos

**Ejemplo 4.** Obtener la expresión regular del siguiente AFN sobre  $\Sigma = \{a, b\}$ .



$$(a \cup b)^*(aa \cup bb)(a \cup b)^*$$

## Teorema

*Sea  $M = (Q, \Sigma, q_0, T, \Delta)$  un AFN. Entonces existe un AFD  $M' = (Q', \Sigma', q'_0, T', \delta)$  tal que  $L(M) = L(M')$ .*

- *El conjunto  $q_0$  se corresponde con  $q'_0$*
- *El conjunto de estados finales  $T'$  de  $Q'$  se corresponde con los conjuntos de estados de  $Q$  que contienen un estado de  $T$*
- *El conjunto de estados de  $Q'$  se corresponde con el conjunto de estados de  $Q$  que se vaya formando mediante el análisis de una cadena sobre  $M$*

# Equivalencia entre autómatas

## Autómatas equivalentes

Dos AFD son equivalentes  $M_1$  y  $M_2$  son equivalentes si  $L(M_1) = L(M_2)$ .

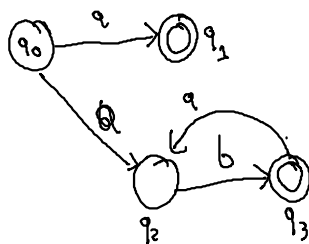
Sean  $M_1$  y  $M_2$  sobre el alfabeto  $\Sigma = \{a\}$ ,



$$L(M_1) = L(M_2) = a^+$$

$(a \cup ab)^+$

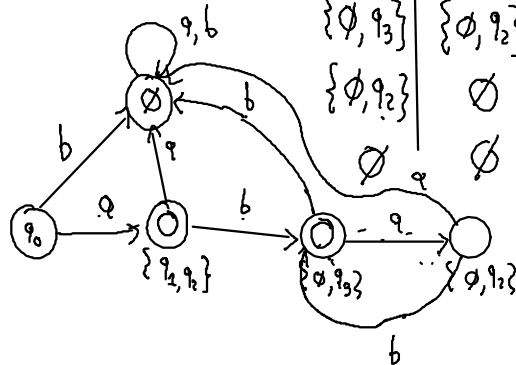
AFN



	a	b
q <sub>0</sub>	{q <sub>1</sub> , q <sub>2</sub> }	∅
q <sub>1</sub>	∅	∅
q <sub>2</sub>	∅	q <sub>3</sub>
q <sub>3</sub>	q <sub>2</sub>	∅

$$T = \{q_1, q_3\}$$

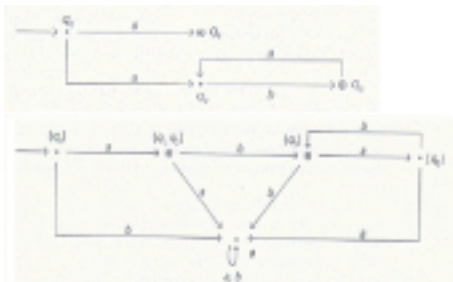
	a	b
q <sub>0</sub>	<u>{q<sub>1</sub>, q<sub>2</sub>}</u>	∅
{q <sub>1</sub> , q <sub>2</sub> }	∅	<u>{∅, q<sub>3</sub>}</u>
{∅, q <sub>3</sub> }	{∅, q <sub>2</sub> }	∅
{∅, q <sub>2</sub> }	∅	<u>{∅, q<sub>3</sub>}</u>
∅	∅	∅



AFD

# Ejemplos equivalencia de AFN y AFD

**Ejemplo 1.** Consideremos el AFN M que acepta  $a \cup (ab)^+$



Para este AFN se tiene:

$$\Delta(q_0, a) = \{q_1, q_2\}$$

$$\Delta(\{q_1, q_2\}, a) = \emptyset$$

$$\Delta(\emptyset, b) = \Delta(\emptyset, b) = \emptyset$$

$$\Delta(q_3, b) = \emptyset$$

$$\Delta(q_0, b) = \emptyset$$

$$\Delta(\{q_1, q_2\}, b) = \{q_3\}$$

$$\Delta(q_3, a) = \{q_2\}$$

$$\Delta(q_2, a) = \emptyset$$

$$\Delta(q_2, b) = \{q_3\}$$

## Ejemplos equivalencia de AFN y AFD

Entonces se verifica que la regla de transición es una función. Por tanto,  $M' = (Q', \Sigma', q'_0, T', \delta)$  donde:

$$Q' = \{\emptyset, \{q_0\}, \{q_2\}, \{q_3\}, \{q_1, q_2\}\}$$

$$\Sigma' = \Sigma$$

$$s' = \{q_0\}$$

$$T' = \{\{q_3\}, \{q_1, q_2\}\}$$

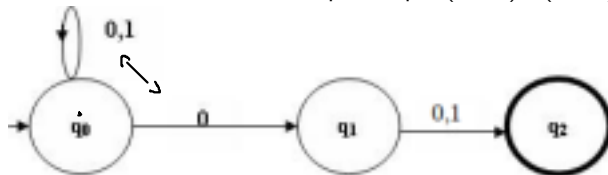
y  $\delta$  viene dada por la siguiente tabla:

$\delta$	$a$	$b$
$\emptyset$	$\emptyset$	$\emptyset$
$\{q_0\}$	$\{q_1, q_2\}$	$\emptyset$
$\{q_2\}$	$\emptyset$	$\{q_3\}$
$\{q_3\}$	$\{q_2\}$	$\emptyset$
$\{q_1, q_2\}$	$\emptyset$	$\{q_3\}$

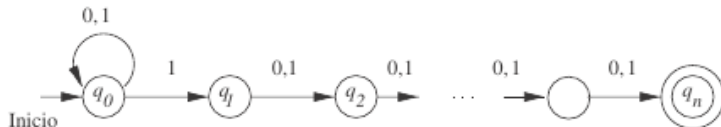


# Ejemplos equivalencia de AFN y AFD

- **Ejemplo 2.** Consideremos el AFN  $M$  que acepta  $(0 \cup 1)^*0(0 \cup 1)$



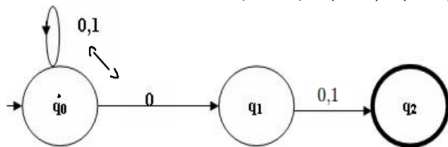
- **Caso desfavorable para la construcción de subconjuntos**



Este AFN no tiene un AFD equivalente con menos de  $2^n$  estados.

Crecimiento exponencial del número de estados para el AFD.

■ **Ejemplo 2.** Consideremos el AFN M que acepta  $(0 \cup 1)^* 0(0 \cup 1)$



q      0      1

q<sub>0</sub>    {q<sub>0</sub>, q<sub>1</sub>}    q<sub>0</sub>

{q<sub>0</sub>, q<sub>1</sub>}    {q<sub>0</sub>, q<sub>1</sub>, q<sub>2</sub>}    {q<sub>0</sub>, q<sub>2</sub>}

{q<sub>0</sub>, q<sub>1</sub>, q<sub>2</sub>}    {q<sub>0</sub>, q<sub>1</sub>, q<sub>2</sub>, ∅}    {q<sub>0</sub>, q<sub>2</sub>, ∅}

{q<sub>0</sub>, q<sub>1</sub>, q<sub>2</sub>, ∅}    {q<sub>0</sub>, q<sub>1</sub>, q<sub>2</sub>, ∅}    {q<sub>0</sub>, q<sub>2</sub>, ∅}

{q<sub>0</sub>, q<sub>2</sub>, ∅}    {q<sub>0</sub>, q<sub>2</sub>, ∅}    ...

(4)

$$2^4 = 16$$

## Teorema

*Si  $L_1$  y  $L_2$  son lenguajes regulares, también lo es  $L_1 \cap L_2$ .*

Sean  $L_1 = L(M_1)$  y  $L_2 = L(M_2)$  donde:  $M_1 = (Q_1, \Sigma_1, q_1, T_1, \delta_1)$  y  $M_2 = (Q_2, \Sigma_2, q_2, T_2, \delta_2)$  Entonces construimos:

$$M = (Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, (q_1, q_2), T_1 \times T_2, \delta)$$

donde

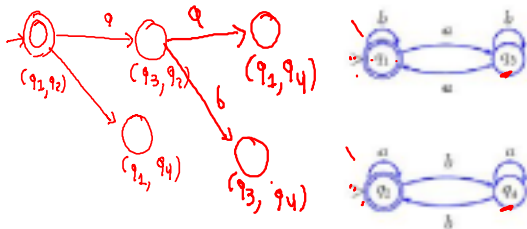
$$\begin{aligned}\delta : Q_1 \times Q_2 \times \Sigma &\rightarrow Q_1 \times Q_2 \\ \delta((q_i, q_j), a) &= (\delta_1(q_i, a), \delta_2(q_j, a))\end{aligned}$$

Esta función satisface:

$$L(M) = L(M_1) \cap L(M_2)$$

## Ejemplo intersección de lenguajes

**Ejemplo.** Construir el AFD que acepte el lenguaje  $L$  de todas las palabras sobre  $\Sigma = \{a, b\}$  que tienen un número par de a's y un número par de b's.



Entonces el lenguaje  $L(M) = L(M_1) \cap L(M_2)$  tiene cuatro estados:

$$Q_1 \times Q_2 = \{(q_1, q_2), (q_1, q_4), (q_3, q_2), (q_3, q_4)\}$$

$$T_1 \times T_2 = \{(q_1, q_2)\}$$

# Ejemplo intersección de lenguajes

Entonces  $\delta$  se define como:

$$\delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a)) = (q_3, q_2)$$

$$\delta((q_1, q_2), b) = (\delta_1(q_1, b), \delta_2(q_2, b)) = (q_1, q_4)$$

$$\delta((q_1, q_4), a) = (\delta_1(q_1, a), \delta_2(q_4, a)) = (q_3, q_4)$$

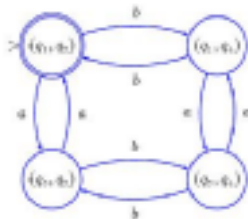
$$\delta((q_1, q_4), b) = (\delta_1(q_1, b), \delta_2(q_4, b)) = (q_1, q_2)$$

$$\delta((q_3, q_2), a) = (\delta_1(q_3, a), \delta_2(q_2, a)) = (q_1, q_2)$$

$$\delta((q_3, q_2), b) = (\delta_1(q_3, b), \delta_2(q_2, b)) = (q_3, q_4)$$

$$\delta((q_3, q_4), a) = (\delta_1(q_3, a), \delta_2(q_4, a)) = (q_1, q_4)$$

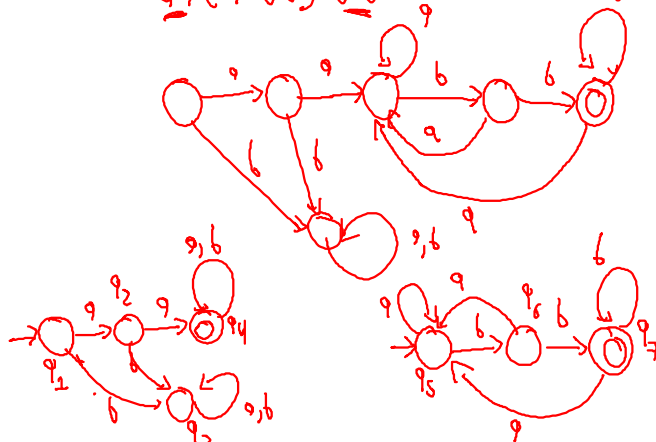
$$\delta((q_3, q_4), b) = (\delta_1(q_3, b), \delta_2(q_4, b)) = (q_3, q_2)$$



¿Como harían ustedes un AFD que reconozca las palabras que inician por aa y terminan en bb?

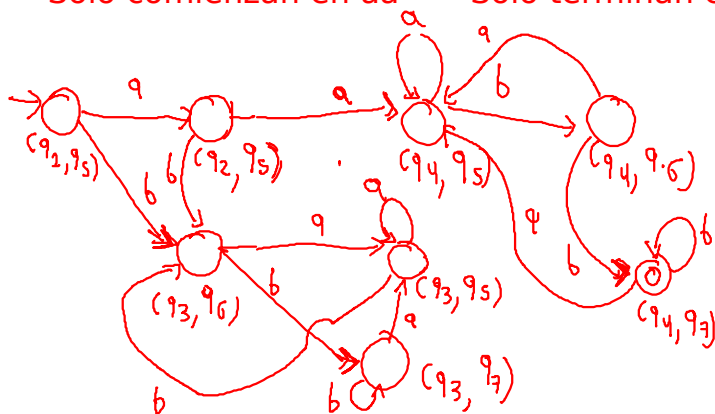
$\Sigma = \{a, b\}$

aa(a u b)\*bb



Solo comienzan en aa

Solo terminan en bb



## Autómatas con $\varepsilon$ -transiciones

Autómatas con  $\varepsilon$ -transiciones: Un autómata con  $\varepsilon$ -transiciones es un AFN  $M = (Q, \Sigma, q_0, T, \Delta)$  en el que la relación de transición está definida así:

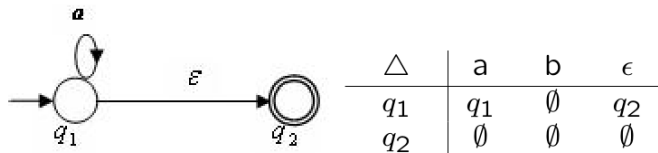
$$\Delta : Q \times (\Sigma \cup \varepsilon) \longrightarrow 2^Q$$

La  $\varepsilon$ -transición permite al autómata cambiar internamente de estado sin consumir el símbolo leído sobre la cinta.

Donde  $2^Q$  denota el conjunto potencia de  $Q$  o el conjunto de todos los subconjuntos de  $Q$ .

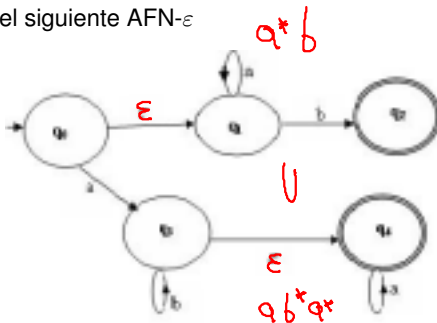
$$2^Q = \{A \mid A \subseteq Q\}$$

**Ejemplo 1.** Se puede representar el lenguaje de la expresión regular  $a^*$  sin necesidad de colocar el estado inicial como estado final.





**Ejemplo 2.** Sea el siguiente AFN- $\epsilon$



La  $\epsilon$ -transición en el AFN permite que se reconozcan cadenas como:

$w=aaab$

$w=abbbbbaaa$

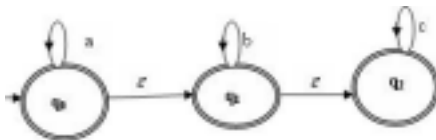
$w=a$

$w=b$

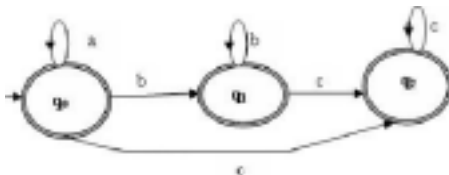
Expresión regular del autómata

$a^* b \cup a b^* a^*$

**Ejemplo 3.** Construir un AFN- $\epsilon$  que reconozca sobre  $\Sigma = \{a, b, c\}$ , el lenguaje  $L = a^*b^*c^*$



El siguiente AFN reconoce el mismo lenguaje que reconoce el AFN- $\epsilon$  anterior.



## Teorema

*Teorema de Kleene. Un lenguaje regular si y sólo si es aceptado por un autómata finito (AFD o AFN o AFN- $\epsilon$ )*

- Construcción de autómatas finitos a partir de expresiones regulares.
- Construcción de expresiones regulares a partir de autómatas:
  - 1 Lema de Arden (Ecuaciones de Lenguaje)
  - 2 Conversión de AFN a expresiones regulares por eliminación de estados.

## Teorema

*Dado un AFN- $\epsilon$   $M = (Q, \Sigma, q_0, T, \Delta)$ , se puede construir un AFN  $M'$  equivalente a  $M$ , es decir  $L(M) = L(M')$ .*

## Teorema

*Un lenguaje regular si y sólo si es aceptado por un autómata finito (AFD o AFN o AFN- $\epsilon$ )*

## Teorema

*Para toda expresión regular  $R$  se puede construir un AFN- $\epsilon$   $M$  tal que  $L(R) = L(M)$ .*

## Paso Básico

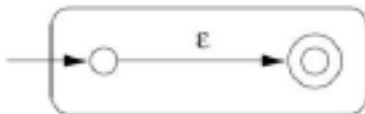
- EL autómata



acepta el lenguaje vacío  $\emptyset$

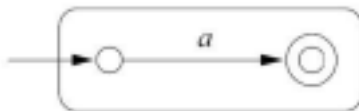
# Autómatas finitos y lenguajes regulares

- EL autómata



acepta el lenguaje  $\{\epsilon\}$

- EL autómata



acepta el lenguaje  $\{a\}$

## PASO INDUCTIVO

1. Existe un autómata que acepta  $R \cup S$



Sean  $M_1 = (Q_1, \Sigma_1, s_1, T_1, \Delta_1)$  y  $M_2 = (Q_2, \Sigma_2, s_2, T_2, \Delta_2)$  para el nuevo  $M = (Q, \Sigma, s, T, \Delta)$  tenemos que:

1  $\Sigma = \Sigma_1 \cup \Sigma_2$

2 En  $T$  se agrega un estado  $s'$  si y sólo si

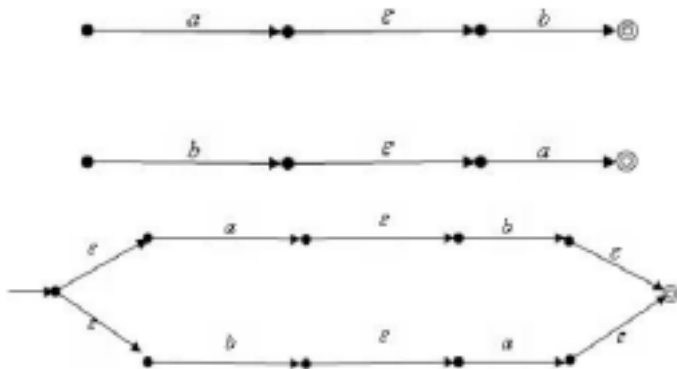
$$\Delta = \Delta_1 \cup \Delta_2 \cup \{(s, \epsilon, s_1), (s, \epsilon, s_2)\} \cup \\ \{(T_1, \epsilon, s'), (T_2, \epsilon, s')\}$$

$s'$  es un estado final NUEVO.

3  $Q = Q_1 \cup Q_2 \cup \{s\} \cup \{s'\}$  donde  $s$  es el nuevo estado inicial.

# Autómatas finitos y lenguajes regulares

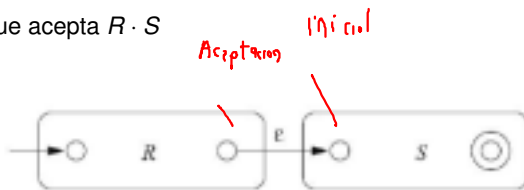
Por ejemplo se construye  $ab \cup ba$ .



Ejemplo. Sobre  $\Sigma = \{a, b\}$  el lenguaje de todas las palabras sobre  $\Sigma$  que tienen un  $n$



## 2. Autómata que acepta $R \cdot S$



Sean  $M_1 = (Q_1, \Sigma_1, s_1, T_1, \Delta_1)$  y  $M_2 = (Q_2, \Sigma_2, s_2, T_2, \Delta_2)$  para el nuevo AFN  $M = (Q, \Sigma, s, T, \Delta)$  que acepta  $L(M_1) \cdot L(M_2)$  tenemos que:

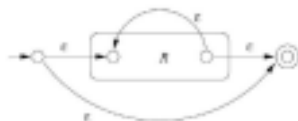
1  $Q = Q_1 \cup Q_2$

2  $s_1 = s$

3  $T = T_2$

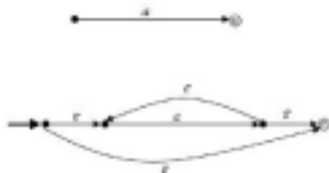
$$\Delta = \Delta_1 \cup \Delta_2 \cup (T_1 \times \{\epsilon\} \times s_2)$$

## 3. Autómata que reconoce $R^*$



Sean  $M_1 = (Q_1, \Sigma_1, s_1, T_1, \Delta_1)$  entonces el nuevo AFN  $M = (Q, \Sigma, s, T, \Delta)$  que acepta  $L(M) = (L(M_1))^*$  viene dado por

- 1  $Q = Q_1 \cup \{s\} \cup \{s'\}$ , donde  $s'$  es un nuevo estado final.
- 2  $T = \{s'\}$
- 3  $\Delta = \Delta_1 \cup \{(s, \epsilon, s_1), (s, \epsilon, s')\} \cup (T_1 \times \{\epsilon\} \times s') \cup (T_1 \times \{\epsilon\} \times s_1)$

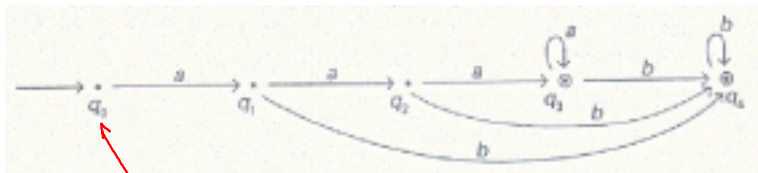


NO INGRESA AL PARCIAL

## Ecuacion del lenguaje

Sea  $\Sigma$  un alfabeto y sean  $E$  y  $A$  subconjuntos de  $\Sigma^*$ , entonces la ecuación del lenguaje  $X = E \cup A \cdot X$  admite la solución  $X = A^* \cdot E$  cualquier otra solución  $Y$  deberá contener  $A \cdot X$ , además  $\epsilon \notin A$ ,  $X = A^* \cdot E$  es la única solución.

**Ejemplo 1.** Encontrar la expresión del siguiente AFD.



Entonces el sistema de ecuaciones a resolver:

$$x_0 = \underline{ax_1}$$

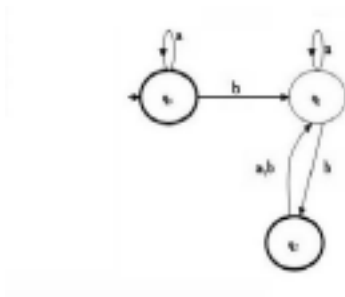
$$x_1 = ax_2 + \underline{bx_4}$$

$$x_2 = ax_3 + bx_4$$

$$x_3 = ax_3 + bx_4 + \underline{\epsilon}$$

$$x_4 = bx_4 + \underline{\epsilon}$$

**Ejemplo 2.** Encontrar la expresión regular del siguiente AFD usando el lema del Arden:



El siguiente es el sistema de ecuaciones a resolver:

$$x_0 = ax_0 + bx_1 + \epsilon$$

$$x_1 = ax_1 + bx_2$$

$$x_2 = (a \cup b)x_1 + \epsilon$$

## Teorema

Sean  $n \geq 2$  considere el sistema de ecuaciones cuyas incógnitas  $x_1, x_2, \dots, x_n$  dado por:

$$\begin{aligned}x_1 &= E_1 \cup A_{11}x_1 \cup A_{12}x_2 \cup \dots \cup A_{1,n}x_n \\x_2 &= E_2 \cup A_{21}x_1 \cup A_{22}x_2 \cup \dots \cup A_{2,n}x_n \\&\vdots \\x_{n-1} &= E_{n-1} \cup A_{(n-1)1}x_1 \cup \dots \cup A_{(n-1),n}x_n \\x_n &= E_n \cup A_{n1}x_1 \cup A_{n2}x_2 \cup \dots \cup A_{n,n}x_n\end{aligned}$$

Entonces el sistema tiene una única solución:

- En  $\forall i, j \in \{1, \dots, n\}, \epsilon \notin A_i$

- Entonces el nuevo sistema se obtiene hasta  $n - 1$ :

$$\begin{aligned}x_1 &= \widehat{E}_1 \cup \widehat{A}_{11}x_1 \cup \widehat{A}_{12}x_2 \cup \dots \cup \widehat{A}_{1,(n-1)}x_{n-1} \\x_2 &= \widehat{E}_2 \cup \widehat{A}_{21}x_1 \cup \widehat{A}_{22}x_2 \cup \dots \cup \widehat{A}_{2,(n-1)}x_{n-1} \\&\vdots \\x_{n-1} &= \widehat{E}_{n-1} \cup \widehat{A}_{(n-1)1}x_1 \cup \dots \cup \widehat{A}_{(n-1),(n-1)}x_{n-1}\end{aligned}$$

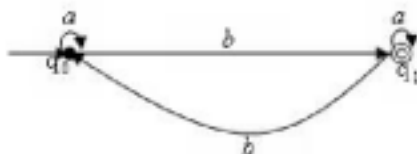
Entonces  $\widehat{E}_i$  y  $\widehat{A}_{ij}$  se definen como:

$$\begin{aligned}\widehat{E}_i &= E_i \cup (A_{in}A_{nn}^*E_n), \quad i = 1, \dots, n-1 \\ \widehat{A}_{ij} &= A_{ij} \cup (A_{in}A_{nn}^*A_{nj}), \quad \forall i, j = 1, \dots, n-1\end{aligned}$$

Donde:

$$E_i = \begin{cases} \emptyset & \text{si } q_i \notin F \\ \epsilon & \text{si } q_i \in F \end{cases}$$

**Ejemplo 1.** Obtener la expresión regular del siguiente AFD usando ecuaciones del lenguaje y la solución única.



El sistema de ecuaciones inicial es:

$$x_1 = ax_1 + bx_2$$

$$x_2 = bx_1 + ax_2 + \epsilon$$



Se aplica el teorema de solución de ecuaciones:

$$x_1 = \hat{E}_1 + \hat{A}_{11}x_1$$

Se obtiene  $\hat{E}_1$

$$\hat{E}_1 = E_1 + (A_{12}A_{22}^*E_2)$$

$$\hat{E}_1 = \emptyset + (b \cdot a^* \cdot \epsilon)$$

$$\hat{E}_1 = ba^*$$

Se obtiene  $\hat{A}_{11}$

$$\hat{A}_{11} = A_{11} + (A_{12}A_{22}^*A_{21})$$

$$\hat{A}_{11} = a + (b \cdot a^* \cdot b)$$

$$\hat{A}_{11} = a + ba^*b$$

Reemplazando  $\hat{E}_1$  y  $\hat{A}_{11}$  en  $x_1$

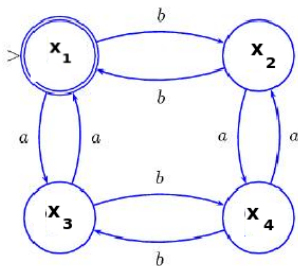
$$x_1 = \hat{E}_1 + \hat{A}_{11}x_1$$

$$x_1 = ba^* + (a + ba^*b)x_1$$

Aplicando solución única se tiene:

$$x_1 = \overbrace{(a + ba^*b)^*}^{\text{red underline}} ba^*$$

# Sistema de ecuaciones por reducción de variables



$$x_1 = ax_3 + bx_2 + \varepsilon$$

$$x_2 = ax_4 + bx_1$$

$$x_3 = ax_1 + bx_4$$

$$x_4 = ax_2 + bx_3$$

$$x_1 = \hat{E}_1 \cup \hat{A}_{11}x_1 \cup \hat{A}_{12}x_2 \cup \hat{A}_{13}x_3$$

$$x_2 = \hat{E}_2 \cup \hat{A}_{21}x_1 \cup \hat{A}_{22}x_2 \cup \hat{A}_{23}x_3$$

$$x_3 = \hat{E}_3 \cup \hat{A}_{31}x_1 \cup \hat{A}_{32}x_2 \cup \hat{A}_{33}x_3$$

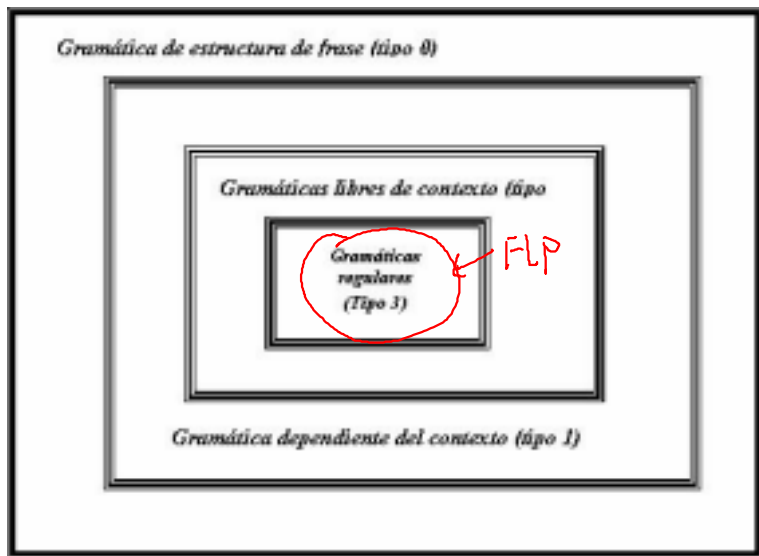
1 Lenguajes

2 Autómatas finitos

3 Gramáticas

4 Máquinas de Turing

Según Chomsky los tipos de gramáticas se clasifican así:



## Gramáticas Regulares (Tipo 3)

Una gramática regular  $G$  es una 4-tupla  $G = (N, \Sigma, S, P)$  que consiste de un conjunto  $N$  de no terminales, un alfabeto  $\Sigma$ , un símbolo inicial  $S$  y de un conjunto de producciones  $P$ . Las reglas son de la forma  $A \rightarrow w$ , donde  $A \in N$  y  $w$  es una cadena sobre  $\Sigma \cup N$  que satisface lo siguiente:

- 1  $w$  contiene un no terminal como máximo.
- 2 Si  $w$  contiene un no terminal, entonces es el símbolo que está en el extremo derecho de  $w$ .
- 3 El conjunto de reglas  $P$  se define así:

$$P \subseteq N \times \Sigma^*(N \cup \epsilon) \quad \text{o} \quad P \subseteq N \times (N \cup \epsilon)\Sigma^*$$

# Definición de gramática regular por la derecha

## Gramáticas regulares

Sobre

$$G = (N, \Sigma, S, P)$$

Una gramática es regular por la derecha si sus producciones son de la forma:

$$\left( \begin{array}{l} A \rightarrow wB, \\ A \rightarrow \varepsilon \end{array} \quad w \in \Sigma^*, B \in N \right)$$

**Ejemplo** Considere la siguiente gramática regular  $G = (N, \Sigma, S, P)$ , que genera  $a^*$ , donde  $\Sigma = \{a, b\}$ ,  $N = \{S, A\}$

$$P : S \rightarrow aA \mid \varepsilon$$

$$A \rightarrow aA \mid \varepsilon$$

**Ejemplo.** Sea la siguiente gramática regular  $G = (N, \Sigma, S, P)$  que genera el lenguaje de la expresión regular  $(a \cup b)^*$

$$\Sigma = \{a, b\}$$

$$N = \{S, A\}$$

$$P : S \rightarrow aS \mid bS \mid \varepsilon$$

**Ejemplo** Considere la siguiente gramática regular  $G = (N, \Sigma, S, P)$ , que genera  $(a \cup b)^+$ , donde  $\Sigma = \{a, b\}$ ,  $N = \{S, A\}$

$P : S \rightarrow aS \mid bS \mid a \mid b$

**Ejemplo** Considere la siguiente gramática regular  $G = (N, \Sigma, S, P)$ , que genera  $a^+b^+$ , donde  $\Sigma = \{a, b\}$ ,  $N = \{S, A\}$

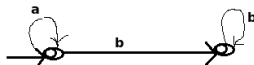
$P : S \rightarrow \underline{a}S \mid \underline{a}A$

$A \rightarrow bA \mid b$

**Ejemplo** Considere la siguiente gramática regular  $G = (N, \Sigma, S, P)$ , que genera  $a^*b^*$ , donde  $\Sigma = \{a, b\}$ ,  $N = \{S, A\}$

$P : S \rightarrow aS \mid bA \mid \varepsilon$

$A \rightarrow bA \mid \varepsilon$





Gramatica regular      000      111      00

$$S \rightarrow 0 \underline{A}$$

$$A \rightarrow 0 B$$

$$B \rightarrow 0 C$$

$$000 \rightarrow C \rightarrow 1 D \mid 0 C$$

$$1 \rightarrow D \rightarrow 1 E \mid 0 C$$

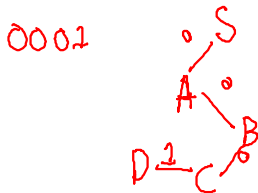
$$11 \rightarrow E \rightarrow 1 F \mid 0 C$$

$$111 \rightarrow F \rightarrow 0 G \mid 1 F$$

$$0 \leftarrow G \rightarrow 0 H \mid 1 F$$

$$00 \rightarrow H \rightarrow \epsilon$$

AFN, AFD



## Gramáticas tipo 2

Una gramática independiente del contexto  $G = (N, \Sigma, S, P)$  consiste de un conjunto  $N$  de no terminales, un alfabeto  $\Sigma$ , un símbolo inicial  $S$  y de un conjunto de producciones  $P$ .

## Definición

*Sea  $G = (N, \Sigma, S, P)$  una gramática independiente del contexto. El lenguaje generado por  $G$  (o el lenguaje de  $G$ ) denotado por  $L(G)$ , es el conjunto de todas las cadenas de terminales que se derivan del estado inicial  $S$ . en otras palabras:*

$$L(G) = \{w \in \Sigma^* / S \Rightarrow^* w\}$$

$$P \subseteq N \times (N \cup \Sigma)^*$$

$S$

## Ejemplo de gramática tipo 2

### Gramáticas ind contexto tipo II

Sea  $G = (N, \Sigma, S, P)$  una gramática con  $\Sigma = \{0, 1\}$  el conjunto  $N = \{S\}$  y  $P$  el conjunto de producciones:

$$S \rightarrow 0S1$$

$$S \rightarrow \varepsilon$$

Automatos de pila

↓  
memoria

**Ejemplo.** Una GIC que genera el lenguaje de los palíndromes sobre  $\Sigma = \{a, b\}$

$$\{ S \rightarrow aSa \mid bSb \mid a \mid b \mid \varepsilon \}$$

$aSa$   
 $bSb$

**Ejemplo.** Una GIC que genera el siguiente lenguaje sobre  $\Sigma = \{a, b\}$  Sea

$$L = \{a^n b^m \mid n \leq m \leq 2n\}$$

$aSb$

$$S \rightarrow aSb \mid aSbb \mid \varepsilon$$

$a^n b^n$   
 $a^n b^{2n}$

## Gramatica tipo I dep del contexto

- 1 El lenguaje de todas las cadenas de paréntesis anidados y equilibrados, por ejemplo:  
 $(( ))(( ))$ , entonces la gramática sería:

$$S \rightarrow \underline{(S)S} \mid \varepsilon$$

- 2 Sea  $T = \{0, 1, (, ), +, *, \emptyset, \varepsilon\}$ .  $T$  es el conjunto de símbolos usados para definir el lenguaje de las expresiones regulares sobre  $\Sigma = \{0, 1\}$ . Se puede diseñar un GIC que genere las expresiones regulares.

$$S \rightarrow \underline{S+S} \mid \underline{SS} \mid \underline{S^*} \mid \underline{(S)} \mid \underline{0} \mid \underline{1} \mid \underline{\emptyset} \mid \varepsilon$$

Handwritten notes below the grammar rule:

- $S \mid S$  (under the first S)
- $U$  (with an equals sign below it)
- $0 \cup 1^*(1)$  (with a brace under the 1's)
- $1^+$  (with a plus sign)
- $0S \mid 1S \mid \emptyset S$  (written as  $0S \mid 1S \mid \emptyset S$ )

## Tipo 0

Sea una 4-tupla  $G = (N, \Sigma, S, P)$  que consiste de un conjunto  $N$  de no terminales, un alfabeto  $\Sigma$ , un símbolo inicial  $S$  y de un conjunto de producciones  $P$ .

- $N$  es el alfabeto de símbolos no terminales
- $\Sigma$  al alfabeto tal que  $N \cap \Sigma = \emptyset$
- $S \in N$  es el símbolo inicial
- $P$  es el conjunto de reglas de producciones de la forma  $\alpha \rightarrow \beta$ , donde  $\alpha \in (N \cup \Sigma)^+$  y  $\beta \in (N \cup \Sigma)^*$ , es decir

$$P \subset (N \cup \Sigma)^+ \times (N \cup \Sigma)^*$$

# Gramáticas no restringidas (Gramáticas de tipo 0 y ?)

**Ejemplo** Sea  $G = (N, \Sigma, S, P)$  una gramática con  $\Sigma = \{0, 1, 2\}$  el conjunto  $N = \{S, A, B\}$  y  $P$  el conjunto de producciones:

$$\begin{array}{ll} S \longrightarrow 0SAB \mid \varepsilon & S \rightarrow 0SAB \\ BA \longrightarrow AB & 00SABAB \\ 0A \longrightarrow 01 & 00ABAB \\ 1A \longrightarrow 11 & 00AABB \\ 1B \longrightarrow 12 & 001ABB \\ 2B \longrightarrow 22 & 0011BB \\ & 00112B \\ & 001122 \end{array}$$

El lenguaje que genera esta gramática dependiente del contexto es:

$$L(G) = \{0^n 1^n 2^n / n = 0, 1, 2, \dots\}$$

Sea  $w=001122$  una cadena que puede ser reconocida por la gramática y que además pertenece al lenguaje.

# Tipos de gramáticas

Tipos de gramáticas		
Tipo	Transiciones	Restricciones en la producciones $w_1 \rightarrow w_2$
0		Sin restricciones
1		$ w_1  \leq  w_2 $ o $w_2 = \epsilon$
2	$P \subseteq N \times (N \cup \Sigma)^*$	$w_1 = A$ , siendo $A$ un símbolo no terminal
3	$P \subseteq N \times \Sigma^*(N \cup \Sigma)$ o $P \subseteq N \times (N \cup \Sigma)\Sigma$	$w_1 = A$ y $w_2 = aB$ o $w_2 = \epsilon$ siendo $A, B \in N$ y $a \in \Gamma$ $S \rightarrow \epsilon$

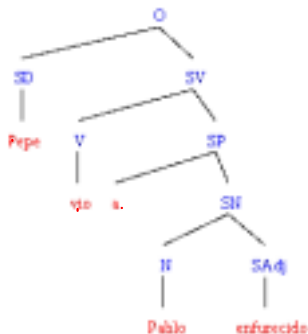
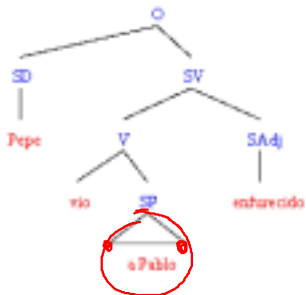
- la familia de los lenguajes de tipo  $i$  contiene a la familia de tipo  $i + 1$ .
- $GR \subseteq GIC \subseteq GDC \subseteq GEF$

Gramática	Lenguaje	Máquina
Tipo 0: Gramática sin restricciones	Recurivamente enumerables / sin restricciones	Máquina de Turing (MT)
Tipo 1: Gramática sensible del contexto	Dependiente del contexto	Autómata Linealmente Acotado (ALA)
Tipo 2: Gramática de contexto libre	Independiente del contexto	Autómata de Pila (AP)
Tipo 3: Gramática Regular	Regular	Autómata finito (AF)

# Arboles de derivación

## Ambigüedad

Una gramática se dice que es *ambigua* si hay dos o más árboles de derivación distintos para la misma cadena. una gramática en la cual, para toda cadena  $w$ , todas las derivaciones de  $w$  tienen el mismo árbol de derivación, es no *ambigua*.



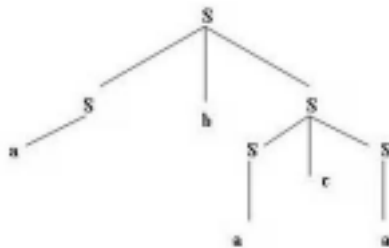
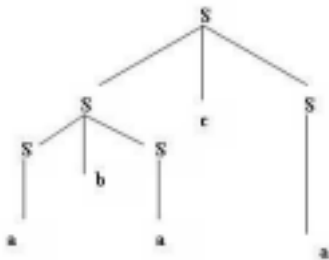


**Ejemplo 2.** Consideremos la siguiente gramática:

$$S \rightarrow SbS \mid ScS \mid a$$

y se la cadena  $w = abaca$  y sus derivaciones:

■  $S \Rightarrow SbS \Rightarrow SbScS \Rightarrow SbSca \Rightarrow abaca$



■  $S \Rightarrow ScS \Rightarrow SbScS \Rightarrow abScS \Rightarrow abacS \Rightarrow abaca$

## Forma de Backus-Naur

La forma de Backus-Naur se emplea para especificar reglas sintácticas de muchos lenguajes de programación y de lenguaje natural: En lugar de utilizar el símbolo  $\rightarrow$  usamos  $::=$  y colocamos los símbolos no terminales entre  $\langle \rangle$ .

La forma BNF se usa frecuentemente para especificar la sintaxis de lenguajes de programación, como Java y LISP; lenguajes de bases de datos, como SQL, y lenguajes de marcado como XML.

**Ejemplo 1.** sea la siguiente GIC:

$O \rightarrow SN \quad SV$

$SN \rightarrow \text{articulo} \quad \text{sustantivo}$

$SV \rightarrow \text{verbo} \quad \text{sustantivo}$

$\text{articulo} \rightarrow \text{el}$

$\text{verbo} \rightarrow \text{come}$

$\text{sustantivo} \rightarrow \text{perro} \mid \text{salchicha}$

La forma Backus-Naur es:

$\langle O \rangle ::= \langle SN \rangle \langle SV \rangle$

$\langle SN \rangle ::= \langle \text{articulo} \rangle \langle \text{sustantivo} \rangle$

$\langle SV \rangle ::= \langle \text{verbo} \rangle \langle \text{sustantivo} \rangle$

$\langle \text{articulo} \rangle ::= \text{el}$

$\langle \text{verbo} \rangle ::= \text{come}$

$\langle \text{sustantivo} \rangle ::= \text{perro} \mid \text{salchicha}$

**Ejemplo 2.** Sea la siguiente gramática:

$$A \rightarrow Aa \mid a \mid AB$$

La forma Backus-Naur es:

$$\langle A \rangle ::= \langle A \rangle a \mid a \mid \langle A \rangle \langle B \rangle$$

**Ejemplo 3.** La producción de enteros son signo en notación decimal. (Un **entero con signo** es un natural precedido por un signo más o un signo menos). La forma Backus-Naur para la gramática que produce los enteros con signo es:

$$\langle \text{entero con signo} \rangle ::= \langle \text{signo} \rangle \langle \text{entero} \rangle$$

$$\langle \text{signo} \rangle ::= + \mid -$$

$$\langle \text{entero} \rangle ::= \langle \text{dígito} \rangle \mid \langle \text{dígito} \rangle \langle \text{entero} \rangle$$

$$\langle \text{dígito} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

+561

1 Lenguajes

2 Autómatas finitos

3 Gramáticas

4 Máquinas de Turing

No

## Introducción

- Su nombre se debe a Alan Mathinson Turing. Quien introdujo el concepto en 1936
- Es un autómata que se puede representar como un dispositivo mecánico
- Se tiene una cinta infinita dividida en celdas
- Contiene un cabezal de escritura/lectura que se mueve sobre la cinta, avanzando una celda cada vez

## Introducción

El movimiento de la máquina de Turing depende del símbolo explorado como la cabeza y el estado actual de la máquina, el resultado puede ser:

- Cambio de estado
- Imprime un símbolo en la cinta, reemplazando el símbolo leído
- Se mueve la cabeza de la cinta a la izquierda o derecha o se para

## Definición

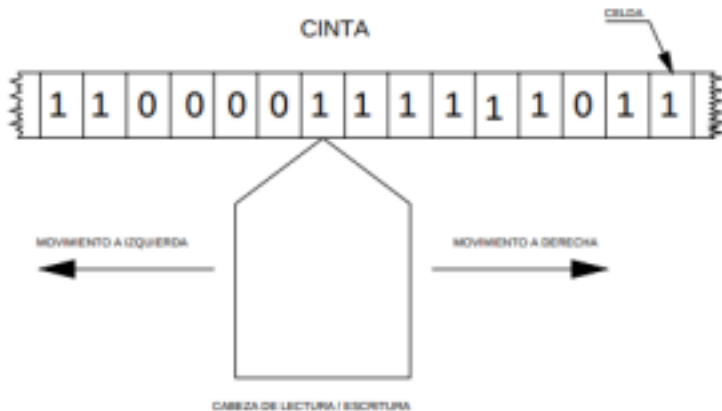
Formalmente, una máquina de Turing es un autómata, el cual esta formando por una quintupla de la forma  $MT = (E, S, Q, f, g)$ , sin embargo suele utilizarse la siguiente denotación:

$$MT = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$$

- $Q$  es un conjunto de estados
- $\Gamma$  conjunto de símbolos permitidos en la cinta
- $B \in \Gamma$  símbolo blanco
- $\Sigma \in \Gamma$  conjunto de símbolos de entrada
- $\delta$  Función de movimiento (Derecha (D), izquierda (I), Parar (S))
- $q_0 \in Q$  Estado inicial
- $F \subset Q$  Conjunto estados finales



# Máquinas de Turing



## Definición

El lenguaje aceptado por una máquina de Turing, lo denotaremos como  $L(MT)$ . Inicialmente una MT está situada a la izquierda de la cadena a reconocer y su estado inicial es  $q_0$ . La MT es capaz de reconocer a un lenguaje  $L$  si para una palabra dada, la máquina termina en un estado de aceptación.

## Ejemplo

Diseñar una máquina de Turing que reconozca el Lenguaje  $L = \{0^n 1^n, n \geq 1\}$ . La cinta contendrá  $0^n 1^n$  con ambos lados rodeados de blancos. El algoritmo de reconocimiento será así:

- La cabeza se mueve al 0 más a la izquierda
- Este es reemplazado por X
- La cabeza se mueva el 1 más a la izquierda
- Es reemplazado por Y
- Después se mueva la izquierda hasta encontrar el X y se mueve uno a la derecha, repitiendo el ciclo

## Ejemplo

- $Q = \{q_0, q_1, q_2, q_3, q_4\}$
- $\Gamma = (0, 1, X, Y, B)$
- $\Sigma = (0, 1)$
- $F = \{q_4\}$  Conjunto estados finales

## Ejemplo

La función  $\delta$  es de la siguiente forma:

$\delta$	0	1	X	Y	B
$q_0$	$q_1, X, D$	-	-	$q_3, Y, D$	-
$q_1$	$q_1, 0, D$	$q_2, Y, I$	-	$q_1, Y, D$	-
$q_2$	$q_2, 0, I$	-	$q_0, X, D$	$q_2, Y, I$	-
$q_3$	-	-	-	$q_3, Y, D$	$q_4, B, D$
$q_4$	S	S	S	S	S

Los guiones (-) significan estados imposibles. La máquina primero escribe, luego cambia de estado y por último se mueve.

## Ejemplo

$q_0$  0011  $\rightarrow$   $Xq_1$ 011  $\rightarrow$   $X0q_1$ 11  $\rightarrow$   $Xq_2$ 0Y1  $\rightarrow$   $q_2$  X0Y1  $\rightarrow$   $Xq_0$ 0Y1  $\rightarrow$   $XXq_1$ Y1  
 $\rightarrow$   $XXYq_1$ 1  $\rightarrow$   $XXq_2$ YY  $\rightarrow$   $Xq_2$ XYY  $\rightarrow$   $XXq_0$ YY  $\rightarrow$   $XXYq_3$  Y  $\rightarrow$   $XXYYq_3$   $\rightarrow$   
 $XXYYBq_4$   $\rightarrow$  S

## Computabilidad

Las máquinas de Turing proveen un marco teórico para definir los problemas computacionales. Los problemas estudiados a través de la máquina de Turing son los problemas de decisión, los cuales tienen como respuesta **si** o **no**.

## Decidibilidad

Si un problema se puede solucionar con una máquina de Turing es solucionable o decidible. En caso contrario, es no solucionable o no decidible.

## Decibilidad

El problema de la parada, se considera un problema no decidable o no solucionable, este consiste en:

- La entrada es una máquina de Turing  $MT'$  codificada en la cinta de entrada
- Así mismo, se tiene en la cinta una entrada  $X$  para esa máquina de Turing
- El objetivo es diseñar un algoritmo en la (MT) de tal forma se pueda determinar que para la entrada  $X$  la máquina  $MT'$  encuentre la solución en tiempo finito



## Clases de problemas

Tenemos dos clases de problemas de decisión

- Clase P, el cual se puede solucionar en tiempo **polinomial** en una MT determinista. Son conocidos como problemas tratables.
- Clase NP, el cual se puede solución en tiempo **polinomial** en una MT no determinista. Son conocidos como problemas no-tratables.



Kenneth H. Rosen.

*Discrete Mathematics and Its Applications.*

McGraw-Hill Higher Education, 7th edition, 2011.

Chapter 13. Modeling Computation.