

# Autómatas y Lenguajes Formales 2016-1

## Maestría en Ciencia e Ingeniería de la Computación UNAM

### Tema 2: Lenguajes regulares y autómatas finitos

Dr. Favio Ezequiel Miranda Perea  
[favio@ciencias.unam.mx](mailto:favio@ciencias.unam.mx)

Facultad de Ciencias UNAM

3 de febrero de 2020



# Lenguajes Regulares

## Definición

Un lenguaje  $L \subseteq \Sigma^*$  es regular si es generado a partir de los lenguajes básicos  $\emptyset, \{\varepsilon\}, \{a\}$  (donde  $a \in \Sigma$ ) mediante las operaciones de unión, concatenación y estrella de Kleene.

Recursivamente:

- $\emptyset$  y  $\{\varepsilon\}$  son lenguajes regulares.
- Si  $a \in \Sigma$  entonces  $\{a\}$  es regular.
- Si  $L, M$  son regulares entonces  $L \cup M, LM$  y  $L^*$  son regulares.
- Son todos.



# Ejemplos

## Lenguajes regulares

- Cualquier lenguaje finito es regular.
- En efecto si  $L = \{w_1, \dots, w_n\} \subseteq \Sigma^*$  entonces  $L = L_1 \cup L_2 \cup \dots \cup L_n$  con  $L_i = \{w_i\}$ . Cada lenguaje  $L_i$  es regular puesto que si  $w_i = a_1 \dots a_{k_i}$  con  $a_j \in \Sigma$  entonces  $L_i = \{a_1\}\{a_2\} \dots \{a_{k_i}\}$  y cada  $\{a_j\}$  es regular por definición.
- En particular cualquier alfabeto  $\Sigma$  es un lenguaje regular.



# Ejemplos

## Lenguajes regulares

- Sobre el alfabeto  $\Sigma = \{a, b\}$  tenemos los siguientes lenguajes regulares infinitos:
  - ▶  $L_1 = \{w \in \Sigma^* \mid w \text{ empieza con } b\} = \{b\}\Sigma^*$
  - ▶  $L_2 = \{w \in \Sigma^* \mid w \text{ contiene exactamente una } a\} = \{b\}^*\{a\}\{b\}^*$
  - ▶  $L_3 = \{w \in \Sigma^* \mid w \text{ contiene la subcadena } ba\} = \Sigma^*\{ba\}\Sigma^*$
  - ▶  $L_4 = \{w \in \Sigma^* \mid w \text{ contiene exactamente dos bes}\} = \{a\}^*\{b\}\{a\}^*\{b\}\{a\}^*$
  - ▶  $L_5 = \{w \in \Sigma^* \mid w \text{ contiene un número par de aes}\} = \{b\}^* \cup (\{b\}^*\{a\}\{b\}^*\{a\}\{b\}^*)^*$
- No todo lenguaje infinito es regular



# Problemas de interés acerca de lenguajes regulares

Dado un lenguaje regular  $L \subseteq \Sigma^*$  existen diversas problemas de decisión (es decir, problemas cuya respuesta es sí o no) acerca de  $L$ , enunciamos algunos de ellos que tienen solución algorítmica.

- ① Problema de vacuidad: ¿ Es  $L$  el lenguaje vacío ?
- ② Problema de finitud: ¿ Es  $L$  finito ?
- ③ Problema de la pertenencia: Dada  $w \in \Sigma^*$ , ¿  $w \in L$  ?
- ④ Problema de la equivalencia: Dado  $M \subseteq \Sigma^*$ , ¿  $M = L$  ?



# Expresiones regulares

## Definición

Un mecanismo de suma importancia para denotar lenguajes regulares es por medio de las llamadas expresiones regulares, definidas recursivamente como sigue:

- $\emptyset$  es una expresión regular.
- $\varepsilon$  es una expresión regular.
- Si  $a \in \Sigma$  entonces  $a$  es una expresión regular.
- Si  $\alpha, \beta$  son expresiones regulares entonces  $\alpha\beta$ ,  $\alpha + \beta$ ,  $\alpha^*$  son expresiones regulares.
- Son todas.



# Significado de una ER

## Expresiones Regulares

Dada una expresión regular  $\alpha$ , su significado es un lenguaje regular, denotado  $L(\alpha)$ , definido como sigue:

- $L(\emptyset) = \emptyset$
- $L(\varepsilon) = \varepsilon$
- $L(a) = \{a\}$  si  $a \in \Sigma$
- $L(\alpha\beta) = L(\alpha)L(\beta)$
- $L(\alpha + \beta) = L(\alpha) \cup L(\beta)$
- $L(\alpha^*) = L(\alpha)^*$



# Lenguajes y Expresiones Regulares

Se cumple la siguiente relación entre lenguajes y expresiones regulares:

## Proposición

*Un lenguaje  $M \subseteq \Sigma^*$  es regular si y sólo si existe una expresión regular  $\alpha$  tal que  $M = L(\alpha)$ .*

## Demostración.

La dirección  $\Leftarrow$  es directa. Para la otra dirección hacemos inducción estructural.



# Equivalencia de Expresiones regulares

## Definición

*Dadas dos expresiones regulares  $\alpha, \beta$ , decimos que son equivalentes y escribimos  $\alpha = \beta$ , si y sólo si  $\alpha$  y  $\beta$  tienen el mismo significado, es decir, si y sólo si  $L(\alpha) = L(\beta)$ .*



# Propiedades

## Expresiones Regulares

$$\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma \quad \alpha + \beta = \beta + \alpha$$

$$\alpha + \emptyset = \alpha \quad \alpha + \alpha = \alpha$$

$$\alpha\varepsilon = \alpha \quad \alpha\emptyset = \emptyset$$

$$\alpha(\beta\gamma) = (\alpha\beta)\gamma$$



# Propiedades

## Expresiones Regulares

$$\alpha(\beta + \gamma) = \alpha\beta + \alpha\gamma \quad (\beta + \gamma)\alpha = \beta\alpha + \gamma\alpha$$

$$\varepsilon^* = \varepsilon \quad \emptyset^* = \varepsilon$$

$$\alpha\alpha^* = \alpha^*\alpha \quad \alpha^* = \alpha^*\alpha^* = (\alpha^*)^*$$

$$\alpha^* = \varepsilon + \alpha\alpha^* \quad (\alpha + \beta)^* = (\alpha^* + \beta^*)^*$$

Si  $L(\alpha) \subseteq L(\beta)$  entonces  $\alpha + \beta = \beta$



# Propiedades de cerradura

## Lenguajes regulares

Las propiedades de cerradura nos permiten construir nuevos lenguajes regulares a partir de lenguajes ya conocidos por medio de algunas operaciones entre lenguajes.

Si  $L, M$  son lenguajes regulares entonces:

- $L \cup M$  es regular.
- $LM$  es regular.
- $L^*$  es regular.
- $L^+$  es regular.
- $\bar{L}$  es regular
- $L \cap M$  es regular.
- $L - M$  es regular.



# Aplicaciones de las expresiones regulares

- Búsqueda y sustitución en editores de texto (vi, emacs, etc.)
- Caza de patrones (pattern matching) y procesamiento de textos y conjuntos de datos, por ejemplo en minería de datos (grep, sed, awk, perl, etc.)
- Implementación de lenguajes de programación (fase de análisis léxico): conversión del programa fuente en una sucesión de elementos léxicos (lexemas o tokens), en esta sucesión se identifican las distintas componentes de un programa, como son palabras reservadas, identificadores, tipos de datos, etc. (lex, flex, etc.)



# ¿Cuando no es $L$ regular?

## Lenguajes Regulares

No todos los lenguajes son regulares, por ejemplo:

$$L = \{a^n b^n \mid n \geq 1\}$$

no es definible mediante una expresión regular.

- ¿Cómo decidir cuando un lenguaje no es regular?
- Mediante propiedades de cerradura.
- Mediante el lema del bombeo (pendiente).



# Autómatas

## Introducción

- Un autómata es una representación abstracta de una máquina.
- Los aspectos relevantes para nosotros son diseño y especificación de autómatas.
- La abstracción captura únicamente el comportamiento de una máquina, es decir, las secuencias de eventos que ocurren.



# Autómatas

## Aplicaciones

- Diseño de circuitos digitales.
- Analizadores léxicos para compiladores.
- Búsqueda de palabras clave en internet.
- Verificación de sistemas de estados finitos. (por ejemplo protocolos de comunicación).
- Modelado de sistemas discretos en general.



# Tipos de Autómatas

- Autómatas finitos: deterministas, no deterministas.
- Autómatas de Pila.
- Autómatas Linealmente Acotados.
- Máquinas de Turing.
- Otros: probabilísticos, celulares, paralelos,etc.



# Tipos de Autómatas

¿Cómo se diferencian?

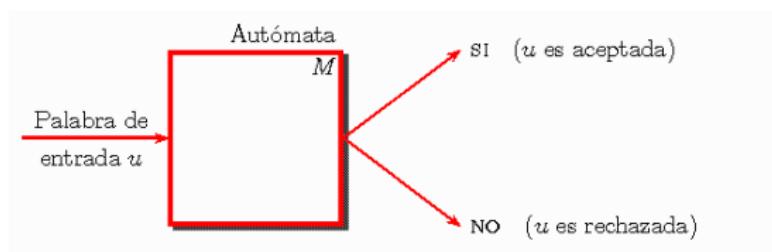
- Detalles particulares del modelo.
- Complejidad.
- Funciones que pueden calcular.
- Lenguajes que pueden reconocer o aceptar.



# Modelo General

## Autómatas

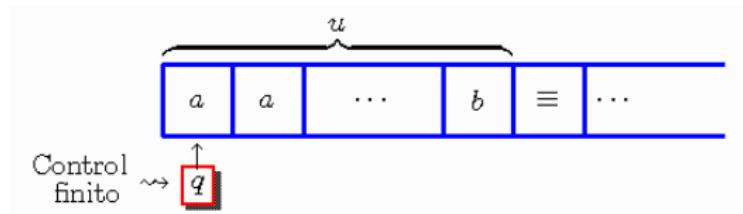
Un autómata finito es una máquina abstracta que procesa cadenas aceptándolas o rechazándolas.



# Lectura y unidad de control

## Autómatas

Las cadenas de entrada se escriben sobre una cinta potencialmente infinita hacia la izquierda dividida en casillas y posee una unidad de control o **cabeza lectora** que lee desde la primera casilla y puede cambiar el estado interno de la máquina dependiendo del símbolo leido.



# Autómatas Finitos Deterministas

## Definición

Un autómata finito determinista (AFD) es una quintupla

$$M = \langle Q, \Sigma, \delta, q_0, F \rangle$$

donde

- $Q$  es un conjunto finito de estados.
- $\Sigma$  es el alfabeto de entrada.
- $\delta : Q \times \Sigma \rightarrow Q$  es la función de transición.
- $q_0 \in Q$  es el estado inicial.
- $F \subseteq Q$  es el conjunto de estados finales.



# Autómatas Finitos Deterministas

## Ejemplo

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

$$Q = \{q_0, q_1, q_2\}$$

$q_0$  : estado inicial

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

Función de transición  $\delta$ :

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

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

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

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

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

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

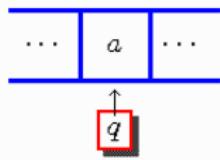
$$\delta(q_2, b) = q_1$$



# Determinismo

## Autómatas finitos

Los autómatas finitos descritos arriba se denominan **autómatas finitos deterministas** ya que para cada estado  $q$  y para cada símbolo  $a \in \Sigma$ , la función de transición  $\delta(q, a)$  siempre está definida. Es decir, la función de transición  $\delta$  *determina únicamente* la acción que el autómata realiza cuando el control finito se encuentra en un estado  $q$  leyendo un símbolo  $a$  sobre la cinta:



# Lenguaje de aceptación

## Autómatas Finitos

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

$$L(M) := \{u \in \Sigma^* : M \text{ se detiene al procesar } u \text{ en un estado } q \in F\}.$$



# Diagrama de estados

## Autómatas finitos

Un autómata finito se puede representar por medio de un grafo dirigido y etiquetado. El grafo de un autómata se obtiene como sigue:

- Los vértices o nodos son los estados del autómata.
- El estado  $q$  se representa por:



- El estado inicial  $q_0$  se representa por:



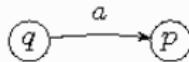
# Diagrama de estados

## Autómatas finitos

- Un estado final  $q$  se representa por:



- La transición  $\delta = (q, a) = p$  se representa en la forma:



# Diagrama de estados

## Ejemplo

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

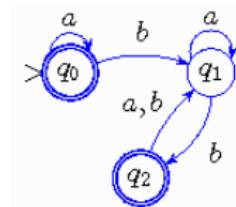
$$Q = \{q_0, q_1, q_2\}$$

$q_0$  : estado inicial

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

Función de transición  $\delta$ :

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



# Estado de error

## Diagrama de estados

Para autómatas deterministas se adopta la siguiente convención adicional con respecto a los diagramas de estados:

- Se asume que las aristas no dibujadas explícitamente conducen a un estado de no-aceptación llamado estado de error.
- Es decir, en el diagrama de estados se indican únicamente las aristas que conduzcan a trayectorias de aceptación. Esto permite simplificar considerablemente los diagramas.



# Problemas de interés

## Autómatas finitos

Los problemas que nos interesa resolver son:

- Dado un autómata  $M$  determinar el lenguaje aceptado por  $M$ .
- Dado un lenguaje regular  $L$  diseñar un autómata finito determinista  $M$  que acepte o reconozca a  $L$ , es decir, tal que  $L(M) = L$ .
- Más adelante se demostrará que estos problemas **siempre** tienen solución.

