Lenguajes Regulares y Autómatas Finitos

Teoría de la Computación

Grado en Ingeniería Informática

Contenidos

Contenidos

1 Lenguajes sobre alfabetos

Nuestro objetivo ahora es la **definición de lenguajes**, es decir, la especificación exacta de qué cadenas componen un determinado lenguaje.

1 Lenguajes sobre alfabetos

Nuestro objetivo ahora es la **definición de lenguajes**, es decir, la especificación exacta de qué cadenas componen un determinado lenguaje.

Ya que todos los lengujes sobre un alfabeto Σ son subconjuntos de Σ^*, comenzaremos estudiando **cuántos sublenguajes contiene $\boldsymbol{\Sigma}^*$**.

1 Lenguajes sobre alfabetos

Nuestro objetivo ahora es la **definición de lenguajes**, es decir, la especificación exacta de qué cadenas componen un determinado lenguaje.

Ya que todos los lengujes sobre un alfabeto Σ son subconjuntos de Σ^*, comenzaremos estudiando **cuántos sublenguajes contiene $\boldsymbol{\Sigma}^*$**.

Dado un alfabeto $\Sigma = \{a_1, a_2, \ldots, a_n\}$, **$\boldsymbol{\Sigma}^*$ es infinito numerable** ya que se puede establecer la siguiente correspondencia entre todas las posibles cadenas sobre Σ y los números de $\mathbb{N}$:

1 Lenguajes sobre alfabetos

Nuestro objetivo ahora es la **definición de lenguajes**, es decir, la especificación exacta de qué cadenas componen un determinado lenguaje.

Ya que todos los lengujes sobre un alfabeto Σ son subconjuntos de Σ^*, comenzaremos estudiando **cuántos sublenguajes contiene Σ^***.

Dado un alfabeto $\Sigma = \{a_1, a_2, \ldots, a_n\}$, Σ^* **es infinito numerable** ya que se puede establecer la siguiente correspondencia entre todas las posibles cadenas sobre Σ y los números de $\mathbb{N}$:

Longitud 0:	$\epsilon \rightarrow 0$

1 Lenguajes sobre alfabetos

Nuestro objetivo ahora es la **definición de lenguajes**, es decir, la especificación exacta de qué cadenas componen un determinado lenguaje.

Ya que todos los lengujes sobre un alfabeto Σ son subconjuntos de Σ^*, comenzaremos estudiando **cuántos sublenguajes contiene Σ^***.

Dado un alfabeto $\Sigma = \{a_1, a_2, \ldots, a_n\}$, Σ^* **es infinito numerable** ya que se puede establecer la siguiente correspondencia entre todas las posibles cadenas sobre Σ y los números de $\mathbb{N}$:

Longitud 0:	$\epsilon \rightarrow 0$			
Longitud 1:	$a_1 \rightarrow 1$	$a_2 \rightarrow 2$	$\ldots$	$a_n \rightarrow n$

1 Lenguajes sobre alfabetos

Nuestro objetivo ahora es la **definición de lenguajes**, es decir, la especificación exacta de qué cadenas componen un determinado lenguaje.

Ya que todos los lengujes sobre un alfabeto Σ son subconjuntos de Σ^*, comenzaremos estudiando **cuántos sublenguajes contiene Σ^***.

Dado un alfabeto $\Sigma = \{a_1, a_2, \ldots, a_n\}$, **$\Sigma^*$ es infinito numerable** ya que se puede establecer la siguiente correspondencia entre todas las posibles cadenas sobre Σ y los números de $\mathbb{N}$:

Longitud 0:	ϵ	$\rightarrow$	0									
Longitud 1:	a_1	$\rightarrow$	1	a_2	$\rightarrow$	2	$\ldots$	a_n	$\rightarrow$	n
Longitud 2:	$a_1 a_1$	$\rightarrow$	$n+1$	$a_2 a_1$	$\rightarrow$	$2n+1$	$\ldots$	$a_n a_1$	$\rightarrow$	n^2+1
	$a_1 a_2$	$\rightarrow$	$n+2$	$a_2 a_2$	$\rightarrow$	$2n+2$	$\ldots$	$a_n a_2$	$\rightarrow$	n^2+2
		$\vdots$			$\vdots$		$\vdots$		$\vdots$	
	$a_1 a_n$	$\rightarrow$	$2n$	$a_2 a_n$	$\rightarrow$	$3n$	$\ldots$	$a_n a_n$	$\rightarrow$	n^2+n

1 Lenguajes sobre alfabetos

Nuestro objetivo ahora es la **definición de lenguajes**, es decir, la especificación exacta de qué cadenas componen un determinado lenguaje.

Ya que todos los lengujes sobre un alfabeto Σ son subconjuntos de Σ^*, comenzaremos estudiando **cuántos sublenguajes contiene Σ^***.

Dado un alfabeto $\Sigma = \{a_1, a_2, \ldots, a_n\}$, **$\Sigma^*$ es infinito numerable** ya que se puede establecer la siguiente correspondencia entre todas las posibles cadenas sobre Σ y los números de $\mathbb{N}$:

Longitud 0:	$\epsilon \rightarrow 0$			
Longitud 1:	$a_1 \rightarrow 1$	$a_2 \rightarrow 2$	$\ldots$	$a_n \rightarrow n$
Longitud 2:	$a_1a_1 \rightarrow n+1$	$a_2a_1 \rightarrow 2n+1$	$\ldots$	$a_na_1 \rightarrow n^2+1$
	$a_1a_2 \rightarrow n+2$	$a_2a_2 \rightarrow 2n+2$	$\ldots$	$a_na_2 \rightarrow n^2+2$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	$a_1a_n \rightarrow 2n$	$a_2a_n \rightarrow 3n$	$\ldots$	$a_na_n \rightarrow n^2+n$
Longitud 3:	$a_1a_1a_1 \rightarrow n^2+n+1$	$\ldots$	$\ldots$	$\ldots$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Pero sabemos que $\mathbf{2}^{\mathbb{N}}$ **no es numerable**. Es decir, el conjunto de todos los posibles subconjuntos de $\mathbb{N}$ no es numerable.

Pero sabemos que $\mathbf{2}^{\mathbb{N}}$ **no es numerable**. Es decir, el conjunto de todos los posibles subconjuntos de $\mathbb{N}$ no es numerable.

Por tanto, $\mathbf{2}^{\Sigma^*}$ **tampoco es numerable**. Es decir, el conjunto de todos los posibles sublenguajes de Σ^* no es numerable.

Pero sabemos que $\mathbf{2}^{\mathbb{N}}$ **no es numerable**. Es decir, el conjunto de todos los posibles subconjuntos de $\mathbb{N}$ no es numerable.

Por tanto, $\mathbf{2}^{\Sigma^*}$ **tampoco es numerable**. Es decir, el conjunto de todos los posibles sublenguajes de Σ^* no es numerable.

Este resultado da idea de la dificultad del problema de especificar lenguajes, ya que existe una cantidad no numerable de lenguajes a especificar sobre cualquier alfabeto dado y no existe ningún método capaz de definirlos todos.

1 Lenguajes sobre alfabetos

Pero sabemos que $\mathbf{2}^{\mathbb{N}}$ **no es numerable**. Es decir, el conjunto de todos los posibles subconjuntos de $\mathbb{N}$ no es numerable.

Por tanto, $\mathbf{2}^{\Sigma^*}$ **tampoco es numerable**. Es decir, el conjunto de todos los posibles sublenguajes de Σ^* no es numerable.

Este resultado da idea de la dificultad del problema de especificar lenguajes, ya que existe una cantidad no numerable de lenguajes a especificar sobre cualquier alfabeto dado y no existe ningún método capaz de definirlos todos.

Así pues, estudiaremos lenguajes sobre alfabetos, pero no todos, sino aquellos que presenten características interesantes para la teoría de la computación, y comenzaremos con los lenguajes regulares.

Contenidos

2 Lenguajes regulares y expresiones regulares

Los *lenguajes regulares* son interesantes por las siguientes razones:

- Desde el punto de vista práctico, pueden ser utilizados para la construcción de analizadores léxicos.

2 Lenguajes regulares y expresiones regulares

Los *lenguajes regulares* son interesantes por las siguientes razones:

- Desde el punto de vista práctico, pueden ser utilizados para la construcción de analizadores léxicos.
- Desde el punto de vista teórico, los lenguajes regulares sobre un alfabeto Σ contienen al lenguaje vacío $\emptyset$, a los lenguajes unitarios $\{\epsilon\}$ y $\{a\}$, $\forall a \in \Sigma$, y constituyen el menor conjunto de lenguajes sobre Σ que es cerrado para las operaciones de unión, concatenación y cierre de Kleene.

2 Lenguajes regulares y expresiones regulares

Los *lenguajes regulares* son interesantes por las siguientes razones:

- Desde el punto de vista práctico, pueden ser utilizados para la construcción de analizadores léxicos.
- Desde el punto de vista teórico, los lenguajes regulares sobre un alfabeto Σ contienen al lenguaje vacío $\emptyset$, a los lenguajes unitarios $\{\epsilon\}$ y $\{a\}$, $\forall a \in \Sigma$, y constituyen el menor conjunto de lenguajes sobre Σ que es cerrado para las operaciones de unión, concatenación y cierre de Kleene.

Definición

Dado un alfabeto Σ, el conjunto de los **lenguajes regulares sobre Σ** se define recursivamente como sigue:

a) $\emptyset$ es un lenguaje regular.

b) $\{\epsilon\}$ es un lenguaje regular.

c) $\{a\}$ es un lenguaje regular, $\forall a \in \Sigma$.

d) Si A y B son lenguajes regulares sobre Σ, entonces también son lenguajes regulares $A \cup B$, $A \cdot B$, A^+ y A^*.

e) Ningún otro lenguaje sobre Σ es regular.

2 Lenguajes regulares y expresiones regulares

Por ejemplo, dado $\Sigma = \{a, b, c\}$, son lenguajes regulares:

- $\emptyset \qquad \{\epsilon\} \qquad \{a\} \qquad \{b\} \qquad \{a, b\} \qquad \{ab\} \qquad \{a, b, ab\}$

2 Lenguajes regulares y expresiones regulares

Por ejemplo, dado $\Sigma = \{a, b, c\}$, son lenguajes regulares:

- $\emptyset \qquad \{\epsilon\} \qquad \{a\} \qquad \{b\} \qquad \{a, b\} \qquad \{ab\} \qquad \{a, b, ab\}$
- $\{a^i \mid i \geq 0\} = \{a\}^*$

2 Lenguajes regulares y expresiones regulares

Por ejemplo, dado $\Sigma = \{a, b, c\}$, son lenguajes regulares:

- $\emptyset \qquad \{\epsilon\} \qquad \{a\} \qquad \{b\} \qquad \{a, b\} \qquad \{ab\} \qquad \{a, b, ab\}$
- $\{a^i \mid i \geq 0\} = \{a\}^*$
- $\{a^i b^j \mid i, j \geq 0\} = \{a\}^* \cdot \{b\}^*$

Por ejemplo, dado $\Sigma = \{a, b, c\}$, son lenguajes regulares:

- $\emptyset \qquad \{\epsilon\} \qquad \{a\} \qquad \{b\} \qquad \{a, b\} \qquad \{ab\} \qquad \{a, b, ab\}$
- $\{a^i \mid i \geq 0\} = \{a\}^*$
- $\{a^i b^j \mid i, j \geq 0\} = \{a\}^* \cdot \{b\}^*$
- $\{(ab)^i \mid i \geq 0\} = \{ab\}^*$

2 Lenguajes regulares y expresiones regulares

Por ejemplo, dado $\Sigma = \{a, b, c\}$, son lenguajes regulares:

- $\emptyset$ $\quad \{\epsilon\}$ $\quad \{a\}$ $\quad \{b\}$ $\quad \{a, b\}$ $\quad \{ab\}$ $\quad \{a, b, ab\}$
- $\{a^i \mid i \geq 0\} = \{a\}^*$
- $\{a^i b^j \mid i, j \geq 0\} = \{a\}^* \cdot \{b\}^*$
- $\{(ab)^i \mid i \geq 0\} = \{ab\}^*$
- El lenguaje de todas las cadenas que no contienen la subcadena *ac*, ya que puede definirse como:

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

Por ejemplo, dado $\Sigma = \{a, b, c\}$, son lenguajes regulares:

- $\emptyset \qquad \{\epsilon\} \qquad \{a\} \qquad \{b\} \qquad \{a, b\} \qquad \{ab\} \qquad \{a, b, ab\}$
- $\{a^i \mid i \geq 0\} = \{a\}^*$
- $\{a^i b^j \mid i, j \geq 0\} = \{a\}^* \cdot \{b\}^*$
- $\{(ab)^i \mid i \geq 0\} = \{ab\}^*$
- El lenguaje de todas las cadenas que no contienen la subcadena *ac*, ya que puede definirse como:
$$\{c\}^* \cdot (\{a\} \cup \{b\} \cdot \{c\}^*)^*$$
- Sin embargo, $\{a^i b^i \mid i \geq 0\}$ no es un lenguaje regular.

2 Lenguajes regulares y expresiones regulares

Por ejemplo, dado $\Sigma = \{a, b, c\}$, son lenguajes regulares:

- $\emptyset \quad \{\epsilon\} \quad \{a\} \quad \{b\} \quad \{a, b\} \quad \{ab\} \quad \{a, b, ab\}$
- $\{a^i \mid i \geq 0\} = \{a\}^*$
- $\{a^i b^j \mid i, j \geq 0\} = \{a\}^* \cdot \{b\}^*$
- $\{(ab)^i \mid i \geq 0\} = \{ab\}^*$
- El lenguaje de todas las cadenas que no contienen la subcadena *ac*, ya que puede definirse como:

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

- Sin embargo, $\{a^i b^i \mid i \geq 0\}$ no es un lenguaje regular.

Más adelante veremos que:

- Existen métodos para saber si un lenguaje es regular o no.

Por ejemplo, dado $\Sigma = \{a, b, c\}$, son lenguajes regulares:

- $\emptyset \qquad \{\epsilon\} \qquad \{a\} \qquad \{b\} \qquad \{a, b\} \qquad \{ab\} \qquad \{a, b, ab\}$
- $\{a^i \mid i \geq 0\} = \{a\}^*$
- $\{a^i b^j \mid i, j \geq 0\} = \{a\}^* \cdot \{b\}^*$
- $\{(ab)^i \mid i \geq 0\} = \{ab\}^*$
- El lenguaje de todas las cadenas que no contienen la subcadena *ac*, ya que puede definirse como:

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

- Sin embargo, $\{a^i b^i \mid i \geq 0\}$ no es un lenguaje regular.

Más adelante veremos que:

- Existen métodos para saber si un lenguaje es regular o no.
- Existen también herramientas para verificar la pertenencia de cualquier cadena a un lenguaje regular dado.

2 Lenguajes regulares y expresiones regulares

La especificación de un lenguaje regular se puede simplificar introduciendo una notación abreviada que denominaremos *expresión regular*.

2 Lenguajes regulares y expresiones regulares

La especificación de un lenguaje regular se puede simplificar introduciendo una notación abreviada que denominaremos *expresión regular*.

Definición

Una **expresión regular** sobre un alfabeto Σ se define recursivamente como sigue:

a) $\emptyset$ es una expresión regular.

b) ϵ es una expresión regular.

c) a es una expresión regular, $\forall a \in \Sigma$.

d) Si r y s son expresiones regulares sobre Σ, entonces también son expresiones regulares $r \cup s$, $r \cdot s$, r^+ y r^*.

e) Ninguna otra secuencia de símbolos es una expresión regular sobre Σ.

2 Lenguajes regulares y expresiones regulares

La especificación de un lenguaje regular se puede simplificar introduciendo una notación abreviada que denominaremos *expresión regular*.

Definición

Una **expresión regular** sobre un alfabeto Σ se define recursivamente como sigue:

a) $\emptyset$ es una expresión regular.

b) ϵ es una expresión regular.

c) a es una expresión regular, $\forall a \in \Sigma$.

d) Si r y s son expresiones regulares sobre Σ, entonces también son expresiones regulares $r \cup s$, $r \cdot s$, r^+ y r^*.

e) Ninguna otra secuencia de símbolos es una expresión regular sobre Σ.

Toda expresión regular r tiene asociado un lenguaje regular que denotamos con $L(r)$.

2 Lenguajes regulares y expresiones regulares

La especificación de un lenguaje regular se puede simplificar introduciendo una notación abreviada que denominaremos *expresión regular*.

Definición

Una **expresión regular** sobre un alfabeto Σ se define recursivamente como sigue:

a) $\emptyset$ es una expresión regular.

b) ϵ es una expresión regular.

c) a es una expresión regular, $\forall a \in \Sigma$.

d) Si r y s son expresiones regulares sobre Σ, entonces también son expresiones regulares $r \cup s$, $r \cdot s$, r^+ y r^*.

e) Ninguna otra secuencia de símbolos es una expresión regular sobre Σ.

Toda expresión regular r tiene asociado un lenguaje regular que denotamos con $L(r)$.

Dos expresiones regulares r y s son equivalentes si $L(r) = L(s)$.

2 Lenguajes regulares y expresiones regulares

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* =$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$

2 Lenguajes regulares y expresiones regulares

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* =$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* = (a \cup b)^*$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* = (a \cup b)^*$
- $(\epsilon \cup aa)^* =$

2 Lenguajes regulares y expresiones regulares

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* = (a \cup b)^*$
- $(\epsilon \cup aa)^* = (aa)^*$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* = (a \cup b)^*$
- $(\epsilon \cup aa)^* = (aa)^*$
- $(a \cup \epsilon) a^* b =$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^*b)^* \cup (b^*a)^* = (a \cup b)^*$
- $(\epsilon \cup aa)^* = (aa)^*$
- $(a \cup \epsilon)a^*b = a^*b$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* = (a \cup b)^*$
- $(\epsilon \cup aa)^* = (aa)^*$
- $(a \cup \epsilon) a^* b = a^* b$
- $(a \cup b)^* a (a \cup b)^* =$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* = (a \cup b)^*$
- $(\epsilon \cup aa)^* = (aa)^*$
- $(a \cup \epsilon) a^* b = a^* b$
- $(a \cup b)^* a (a \cup b)^* = (a \cup b)^* a (a \cup b)^*$ (no se puede simplificar más)

2 Lenguajes regulares y expresiones regulares

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* = (a \cup b)^*$
- $(\epsilon \cup aa)^* = (aa)^*$
- $(a \cup \epsilon) a^* b = a^* b$
- $(a \cup b)^* a (a \cup b)^* = (a \cup b)^* a (a \cup b)^*$ (no se puede simplificar más)
- $(aa)^* a \cup (aa)^* =$

Ejemplos de expresiones regulares equivalentes:

$$r \cup s = s \cup r \qquad r \cup \emptyset = \emptyset \cup r = r$$

$$r \cup r = r \qquad r \cdot \epsilon = \epsilon \cdot r = r$$

$$r \cdot \emptyset = \emptyset \cdot r = \emptyset \qquad r \cdot r^* = r^* \cdot r = r^+$$

$$r^* = \epsilon \cup r^+ \qquad \ldots$$

Ejercicio

Simplifique todo lo posible las siguientes expresiones regulares:

- $\emptyset \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$
- $(a^* b)^* \cup (b^* a)^* = (a \cup b)^*$
- $(\epsilon \cup aa)^* = (aa)^*$
- $(a \cup \epsilon) a^* b = a^* b$
- $(a \cup b)^* a (a \cup b)^* = (a \cup b)^* a (a \cup b)^*$ (no se puede simplificar más)
- $(aa)^* a \cup (aa)^* = a^*$ ({nº impar de *a*es} $\cup$ {nº par de *a*es} = {cq. nº de *a*es})

.../...

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$
  $(a \cup b)(aa)^* \cup (a \cup b) =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$
  $(a \cup b)(aa)^* \cup (a \cup b) =$
  $(a \cup b)(aa)^*$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$
  $(a \cup b)(aa)^* \cup (a \cup b) =$
  $(a \cup b)(aa)^*$
- $(\epsilon \cup aa)(\epsilon \cup aa)^*(ab \cup b) \cup (ab \cup b) =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$
  $(a \cup b)(aa)^* \cup (a \cup b) =$
  $(a \cup b)(aa)^*$
- $(\epsilon \cup aa)(\epsilon \cup aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) \cup (ab \cup b) =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$
  $(a \cup b)(aa)^* \cup (a \cup b) =$
  $(a \cup b)(aa)^*$
- $(\epsilon \cup aa)(\epsilon \cup aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$
  $(a \cup b)(aa)^* \cup (a \cup b) =$
  $(a \cup b)(aa)^*$
- $(\epsilon \cup aa)(\epsilon \cup aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) =$
  $(aa)^* ab \cup (aa)^* b =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a(\epsilon \cup aa)^+ \cup a = a(aa)^* \cup a = a(aa)^*$
- $a(\epsilon \cup aa)^* a \cup \epsilon = a(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$
  $(a \cup b)(aa)^* \cup (a \cup b) =$
  $(a \cup b)(aa)^*$
- $(\epsilon \cup aa)(\epsilon \cup aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) =$
  $(aa)^* ab \cup (aa)^* b =$
  $((aa)^* a \cup (aa)^*)b =$

Ejercicio (continuación)

- $(\epsilon \cup aa)(\epsilon \cup aa)^* = (\epsilon \cup aa)^+ = \epsilon \cup (aa)^+ = (aa)^*$
- $a\,(\epsilon \cup aa)^*(\epsilon \cup aa) \cup a = a\,(\epsilon \cup aa)^+ \cup a = a\,(aa)^* \cup a = a\,(aa)^*$
- $a\,(\epsilon \cup aa)^* a \cup \epsilon = a\,(aa)^* a \cup \epsilon = (aa)^+ \cup \epsilon = (aa)^*$
- $(a \cup b)(\epsilon \cup aa)^*(\epsilon \cup aa) \cup (a \cup b) =$
  $(a \cup b)(aa)^* \cup (a \cup b) =$
  $(a \cup b)(aa)^*$
- $(\epsilon \cup aa)(\epsilon \cup aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) \cup (ab \cup b) =$
  $(aa)^*(ab \cup b) =$
  $(aa)^* ab \cup (aa)^* b =$
  $((aa)^* a \cup (aa)^*)b =$
  $a^* b$

Contenidos

3 Autómata finito determinista (AFD)

Consideremos el lenguaje regular A denotado por $c^*(a \cup bc^*)^*$ y una cadena w. Para ver si $w \in A$, tendríamos que analizar no sólo los símbolos de w, sino también sus posiciones relativas. Por ejemplo, $abc^5ab \in A$, pero $cabac^3bc \notin A$.

3 Autómata finito determinista (AFD)

Consideremos el lenguaje regular A denotado por $c^*(a \cup bc^*)^*$ y una cadena w. Para ver si $w \in A$, tendríamos que analizar no sólo los símbolos de w, sino también sus posiciones relativas. Por ejemplo, $abc^5ab \in A$, pero $cabac^3bc \notin A$.

Sin embargo, podemos construir una estructura algebraica que nos ayude a determinar esta cuestión. Para los lenguajes regulares, dicha estructura es el *autómata finito*.

3 Autómata finito determinista (AFD)

Consideremos el lenguaje regular A denotado por $c^*(a \cup bc^*)^*$ y una cadena w. Para ver si $w \in A$, tendríamos que analizar no sólo los símbolos de w, sino también sus posiciones relativas. Por ejemplo, $abc^5ab \in A$, pero $cabac^3bc \notin A$.

Sin embargo, podemos construir una estructura algebraica que nos ayude a determinar esta cuestión. Para los lenguajes regulares, dicha estructura es el *autómata finito*.

Definición

Un **autómata finito determinista (AFD)** M es una colección de cinco elementos:

$$M = (Q, \Sigma, s, \delta, F)$$

donde:

- Q es un conjunto finito de estados,
- Σ es el alfabeto de los símbolos de entrada,
- $s \in Q$ es el estado inicial del autómata,
- $\delta : Q \times \Sigma \to Q$ es la función de transición que determina el estado siguiente para cada par (q, σ), donde q es el estado actual del autómata y σ es el primer símbolo pendiente de procesar de la cadena de entrada,
- $F \subseteq Q$ es el conjunto de estados finales o de aceptación.

Ejemplo:

$$Q = \{q_0, q_1, q_2\} \qquad \delta : Q \times \Sigma \to Q$$

$$\Sigma = \{a, b\} \qquad (q_0, a) \rightsquigarrow q_1$$

$$s = q_0 \qquad (q_0, b) \rightsquigarrow q_2$$

$$F = \{q_0\} \qquad (q_1, a) \rightsquigarrow q_2$$

$$(q_1, b) \rightsquigarrow q_0$$

$$(q_2, a) \rightsquigarrow q_2$$

$$(q_2, b) \rightsquigarrow q_2$$

Ejemplo:

$$Q = \{q_0, q_1, q_2\} \qquad \delta : Q \times \Sigma \to Q$$

$$\Sigma = \{a, b\} \qquad (q_0, a) \rightsquigarrow q_1$$

$$s = q_0 \qquad (q_0, b) \rightsquigarrow q_2$$

$$F = \{q_0\} \qquad (q_1, a) \rightsquigarrow q_2$$

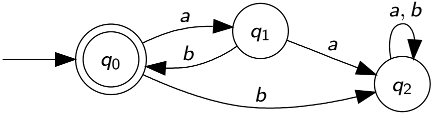
$$(q_1, b) \rightsquigarrow q_0$$

$$(q_2, a) \rightsquigarrow q_2$$

$$(q_2, b) \rightsquigarrow q_2$$

Formas de representación:

- Mediante un grafo (útil para la visualización):



Ejemplo:

$$Q = \{q_0, q_1, q_2\} \qquad \delta : Q \times \Sigma \to Q$$

$$\Sigma = \{a, b\} \qquad (q_0, a) \rightsquigarrow q_1$$

$$s = q_0 \qquad (q_0, b) \rightsquigarrow q_2$$

$$F = \{q_0\} \qquad (q_1, a) \rightsquigarrow q_2$$

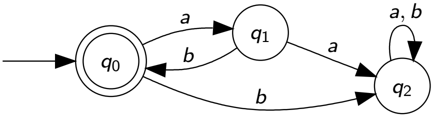
$$(q_1, b) \rightsquigarrow q_0$$

$$(q_2, a) \rightsquigarrow q_2$$

$$(q_2, b) \rightsquigarrow q_2$$

Formas de representación:

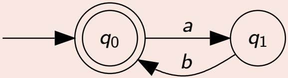
- Mediante un grafo (útil para la visualización):



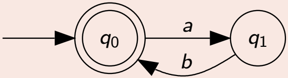
- Mediante una tabla (útil para la implementación):

δ	a	b
$\to *q_0$	q_1	q_2
q_1	q_2	q_0
q_2	q_2	q_2

¿Es este autómata finito un AFD?

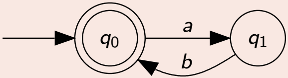


¿Es este autómata finito un AFD?



Respuesta: No, tal y como lo hemos definido, porque la función de transición no está completamente especificada, ya que no sabemos quiénes son $\delta(q_0, b)$ ni $\delta(q_1, a)$.

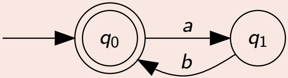
¿Es este autómata finito un AFD?



Respuesta: No, tal y como lo hemos definido, porque la función de transición no está completamente especificada, ya que no sabemos quiénes son $\delta(q_0, b)$ ni $\delta(q_1, a)$.

Solución: Crear un estado sumidero al que van todos los arcos que faltan y del cual no se puede salir (como es el caso de q_2 en el ejemplo anterior).

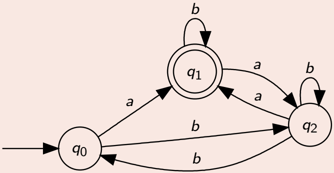
¿Es este autómata finito un AFD?



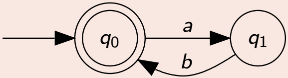
Respuesta: No, tal y como lo hemos definido, porque la función de transición no está completamente especificada, ya que no sabemos quiénes son $\delta(q_0, b)$ ni $\delta(q_1, a)$.

Solución: Crear un estado sumidero al que van todos los arcos que faltan y del cual no se puede salir (como es el caso de q_2 en el ejemplo anterior).

¿Y este otro?



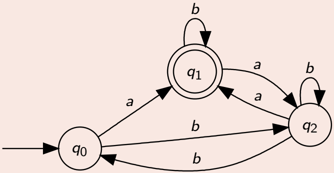
¿Es este autómata finito un AFD?



Respuesta: No, tal y como lo hemos definido, porque la función de transición no está completamente especificada, ya que no sabemos quiénes son $\delta(q_0, b)$ ni $\delta(q_1, a)$.

Solución: Crear un estado sumidero al que van todos los arcos que faltan y del cual no se puede salir (como es el caso de q_2 en el ejemplo anterior).

¿Y este otro?



Este autómata finito tampoco es un AFD, ya que $\delta(q_2, b)$ tiene dos posibles valores, y por tanto δ no sería una función.

3 Autómata finito determinista (AFD)

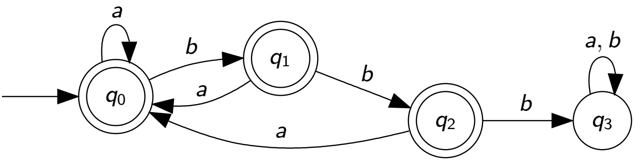
Una cadena $w = a_1 a_2 \dots a_n$ es aceptada por un AFD cuando, empezando en el estado inicial s, procesamos todos sus símbolos y terminamos en un estado final. Es decir, si $\hat{\delta}(s, w) \in F$, donde:

$$\hat{\delta}(s, w) = \delta(\dots \delta(\delta(\delta(s, a_1), a_2), a_3) \dots, a_n)$$

Una cadena $w = a_1 a_2 \dots a_n$ es aceptada por un AFD cuando, empezando en el estado inicial s, procesamos todos sus símbolos y terminamos en un estado final. Es decir, si $\hat{\delta}(s, w) \in F$, donde:

$$\hat{\delta}(s, w) = \delta(\dots \delta(\delta(\delta(s, a_1), a_2), a_3) \dots, a_n)$$

Ejemplo:

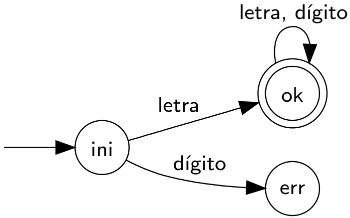


$$\hat{\delta}(q_0, aabbab) = q_1 \in F$$

$$\hat{\delta}(q_0, ababbb \dots) = q_3 \notin F$$

3 Autómata finito determinista (AFD)

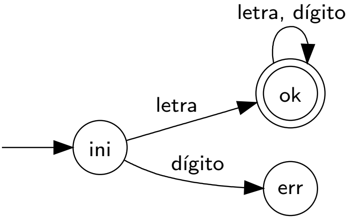
Otro ejemplo:



Este es un AFD que acepta los identificadores de un lenguaje de programación.

3 Autómata finito determinista (AFD)

Otro ejemplo:



Este es un AFD que acepta los identificadores de un lenguaje de programación.

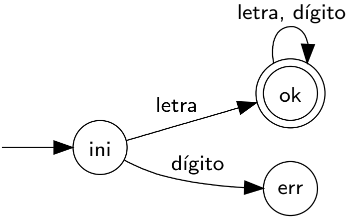
Definición

Dado $M = (Q, \Sigma, s, \delta, F)$ un AFD arbitario, definimos el **lenguaje aceptado por** M como sigue:

$$L(M) = \{w \in \Sigma^* \mid \hat{\delta}(s, w) \in F\}$$

3 Autómata finito determinista (AFD)

Otro ejemplo:



Este es un AFD que acepta los identificadores de un lenguaje de programación.

Definición

Dado $M = (Q, \Sigma, s, \delta, F)$ un AFD arbitario, definimos el **lenguaje aceptado por** M como sigue:

$$L(M) = \{w \in \Sigma^* \mid \hat{\delta}(s, w) \in F\}$$

Definición

Dados M_1 y M_2 dos AFD,s arbitrarios, decimos que son **equivalentes** cuando aceptan el mismo lenguaje, es decir, si:

$$L(M_1) = L(M_2)$$

3 Autómata finito determinista (AFD)

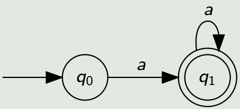
Ejercicio

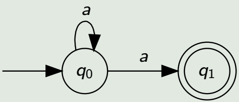
Construya un autómata finito que acepte el lenguaje a^+.

Ejercicio

Construya un autómata finito que acepte el lenguaje a^+.

Solución:



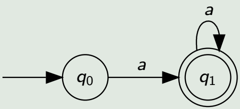


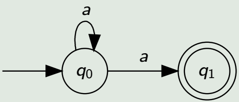
El primero es un autómata finito determinista. El segundo es un autómata finito no determinista. Ambos son equivalentes. Más adelante veremos la relación entre ellos.

Ejercicio

Construya un autómata finito que acepte el lenguaje a^+.

Solución:





El primero es un autómata finito determinista. El segundo es un autómata finito no determinista. Ambos son equivalentes. Más adelante veremos la relación entre ellos.

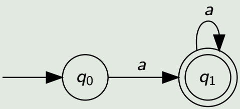
Ejercicio

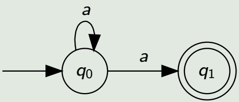
Construya un autómata finito que acepte el lenguaje a^*.

Ejercicio

Construya un autómata finito que acepte el lenguaje a^+.

<u>Solución</u>:



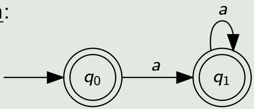


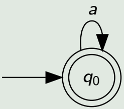
El primero es un autómata finito determinista. El segundo es un autómata finito no determinista. Ambos son equivalentes. Más adelante veremos la relación entre ellos.

Ejercicio

Construya un autómata finito que acepte el lenguaje a^*.

<u>Solución</u>:





El primero es un AFD. El segundo es un AFD mínimo. Ambos son equivalentes. Más adelante hablaremos del concepto de AFD mínimo y de los métodos de minimización.

Ejercicio

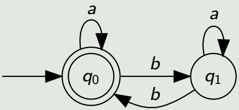
Obtenga la expresión regular que represente al lenguaje formado por todas las cadenas sobre $\{a, b\}$ que tienen un número par de $b$es. Construya el AFD correspondiente.

Ejercicio

Obtenga la expresión regular que represente al lenguaje formado por todas las cadenas sobre $\{a, b\}$ que tienen un número par de *b*es. Construya el AFD correspondiente.

Solución:

$(a^* b\, a^* b\, a^*)^* \cup a^*$ o bien $(a \cup b\, a^* b)^*$

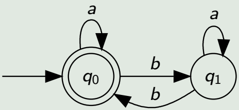


Ejercicio

Obtenga la expresión regular que represente al lenguaje formado por todas las cadenas sobre $\{a, b\}$ que tienen un número par de $b$es. Construya el AFD correspondiente.

Solución:

$(a^* b\, a^* b\, a^*)^* \cup a^*$ o bien $(a \cup b\, a^* b)^*$



Ejercicio

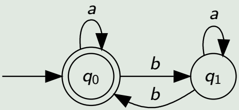
Construya un AFD que acepte el lenguaje sobre $\{a, b\}$ formado por todas las cadenas donde toda a está entre dos $b$es.

Ejercicio

Obtenga la expresión regular que represente al lenguaje formado por todas las cadenas sobre $\{a, b\}$ que tienen un número par de *b*es. Construya el AFD correspondiente.

Solución:

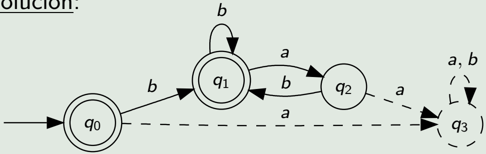
$(a^* \, b \, a^* \, b \, a^*)^* \cup a^*$ o bien $(a \cup b \, a^* \, b)^*$



Ejercicio

Construya un AFD que acepte el lenguaje sobre $\{a, b\}$ formado por todas las cadenas donde toda *a* está entre dos *b*es.

Solución:



- q_0 es final para aceptar ϵ
- q_1 lleva memoria de que ha llegado una b
- q_2 lleva memoria de que ha llegado una a
- q_3 es un estado sumidero opcional

3 Autómata finito determinista (AFD)

Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

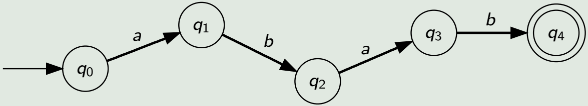
$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ contiene la subcadena } abab\}$$

Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ contiene la subcadena } abab\}$$

Solución:

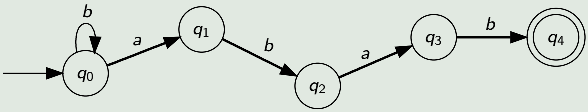


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ contiene la subcadena } abab\}$$

Solución:

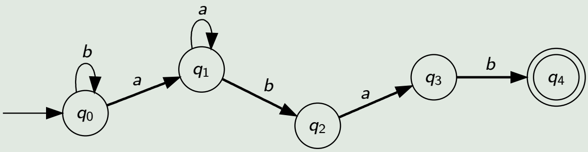


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ contiene la subcadena } abab\}$$

Solución:

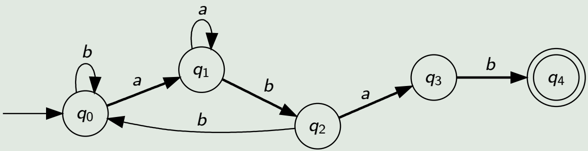


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ contiene la subcadena } abab\}$$

Solución:

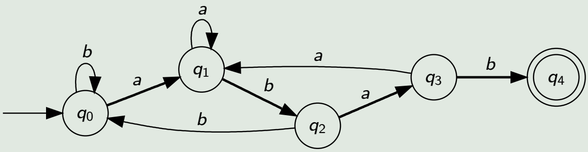


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ contiene la subcadena } abab\}$$

Solución:

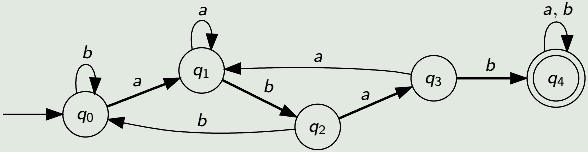


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ contiene la subcadena } abab\}$$

Solución:



Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

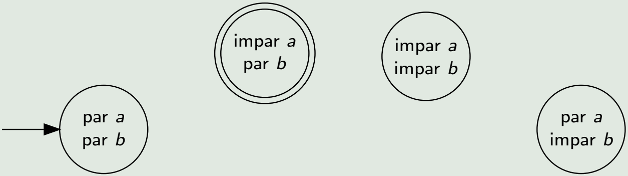
$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ tiene número impar } a\text{es y número par de } b\text{es}\}$$

Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ tiene número impar } a\text{es y número par de } b\text{es}\}$$

<u>Solución</u>:

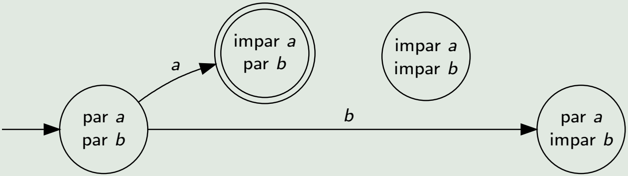


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ tiene número impar } a\text{es y número par de } b\text{es}\}$$

<u>Solución</u>:

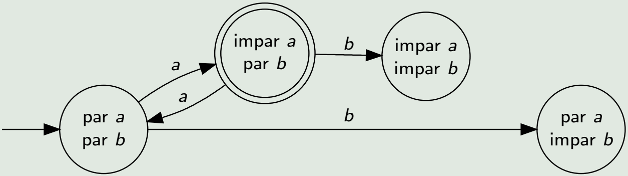


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ tiene número impar } a\text{es y número par de } b\text{es}\}$$

Solución:

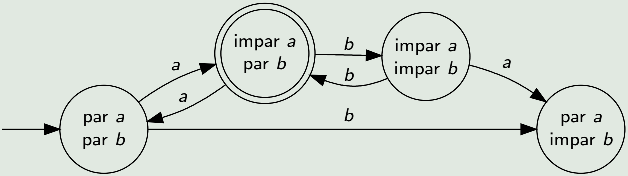


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ tiene número impar } a\text{es y número par de } b\text{es}\}$$

<u>Solución</u>:

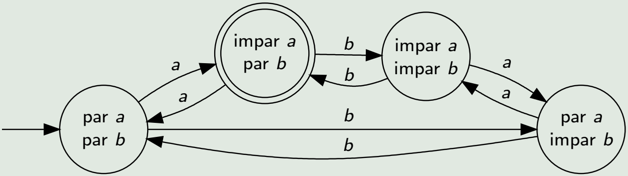


Ejercicio

Construya un AFD que acepte el siguiente lenguaje:

$$\{w \mid w \in \{a, b\}^* \text{ y } w \text{ tiene número impar } a\text{es y número par de } b\text{es}\}$$

<u>Solución</u>:



Ejercicio

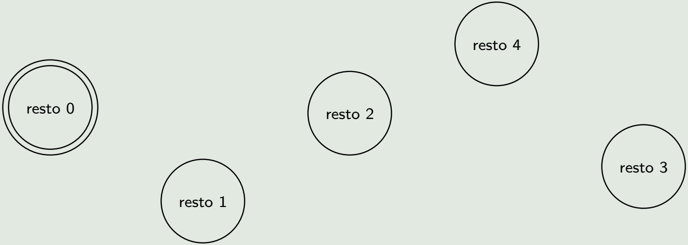
Las cadenas de ceros y unos pueden interpretarse como representaciones binarias de los números naturales. Construya un AFD sobre $\Sigma = \{0, 1\}$ que acepte los múltiplos de 5.

3 Autómata finito determinista (AFD)

Ejercicio

Las cadenas de ceros y unos pueden interpretarse como representaciones binarias de los números naturales. Construya un AFD sobre $\Sigma = \{0, 1\}$ que acepte los múltiplos de 5.

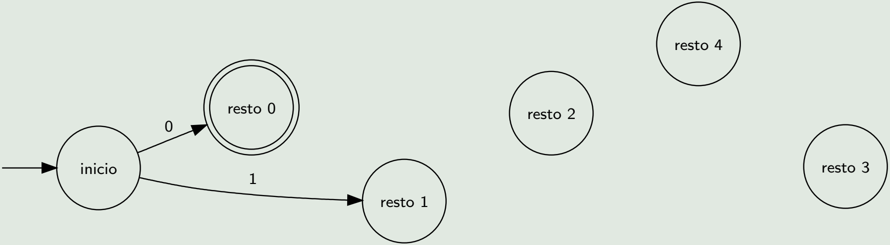
Solución:
Cada estado se corresponde con los posibles restos de dividir entre 5.
Si llega un 0, se multiplica por dos; si llega un 1, se multiplica por dos y se suma uno.
Estas mismas operaciones se pueden hacer con los restos, ya que son congruentes en cualquier $\mathbb{Z}_n$ (clase de restos módulo n).



Ejercicio

Las cadenas de ceros y unos pueden interpretarse como representaciones binarias de los números naturales. Construya un AFD sobre $\Sigma = \{0, 1\}$ que acepte los múltiplos de 5.

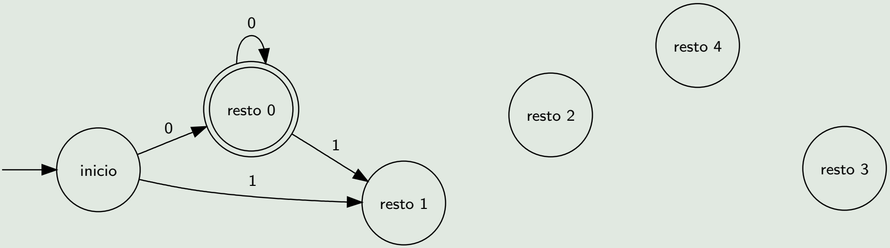
Solución:
Cada estado se corresponde con los posibles restos de dividir entre 5.
Si llega un 0, se multiplica por dos; si llega un 1, se multiplica por dos y se suma uno.
Estas mismas operaciones se pueden hacer con los restos, ya que son congruentes en cualquier $\mathbb{Z}_n$ (clase de restos módulo n).



Ejercicio

Las cadenas de ceros y unos pueden interpretarse como representaciones binarias de los números naturales. Construya un AFD sobre $\Sigma = \{0, 1\}$ que acepte los múltiplos de 5.

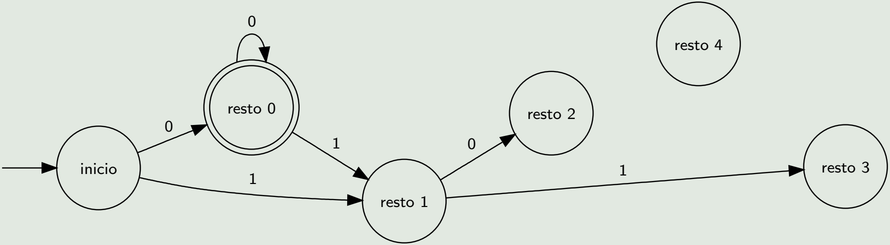
Solución:
Cada estado se corresponde con los posibles restos de dividir entre 5.
Si llega un 0, se multiplica por dos; si llega un 1, se multiplica por dos y se suma uno.
Estas mismas operaciones se pueden hacer con los restos, ya que son congruentes en cualquier $\mathbb{Z}_n$ (clase de restos módulo n).



Ejercicio

Las cadenas de ceros y unos pueden interpretarse como representaciones binarias de los números naturales. Construya un AFD sobre $\Sigma = \{0, 1\}$ que acepte los múltiplos de 5.

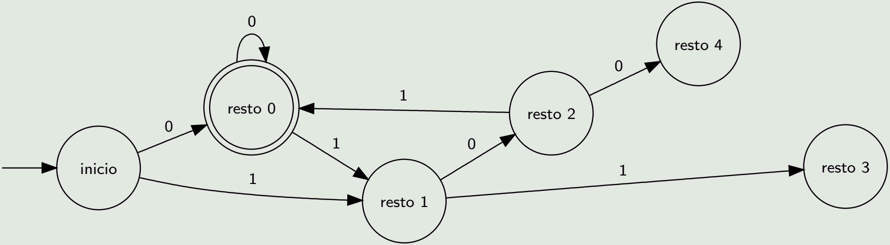
Solución:
Cada estado se corresponde con los posibles restos de dividir entre 5.
Si llega un 0, se multiplica por dos; si llega un 1, se multiplica por dos y se suma uno.
Estas mismas operaciones se pueden hacer con los restos, ya que son congruentes en cualquier $\mathbb{Z}_n$ (clase de restos módulo n).



Ejercicio

Las cadenas de ceros y unos pueden interpretarse como representaciones binarias de los números naturales. Construya un AFD sobre $\Sigma = \{0, 1\}$ que acepte los múltiplos de 5.

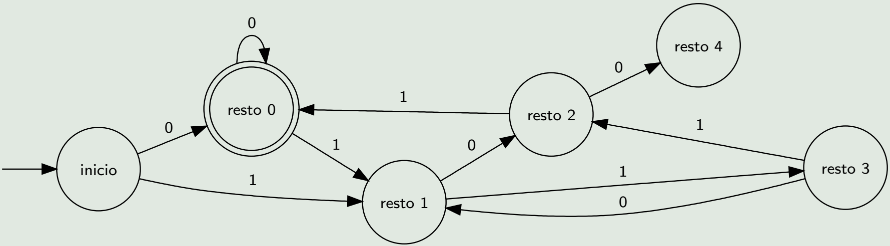
Solución:
Cada estado se corresponde con los posibles restos de dividir entre 5.
Si llega un 0, se multiplica por dos; si llega un 1, se multiplica por dos y se suma uno.
Estas mismas operaciones se pueden hacer con los restos, ya que son congruentes en cualquier $\mathbb{Z}_n$ (clase de restos módulo n).



Ejercicio

Las cadenas de ceros y unos pueden interpretarse como representaciones binarias de los números naturales. Construya un AFD sobre $\Sigma = \{0, 1\}$ que acepte los múltiplos de 5.

Solución:
Cada estado se corresponde con los posibles restos de dividir entre 5.
Si llega un 0, se multiplica por dos; si llega un 1, se multiplica por dos y se suma uno.
Estas mismas operaciones se pueden hacer con los restos, ya que son congruentes en cualquier $\mathbb{Z}_n$ (clase de restos módulo n).

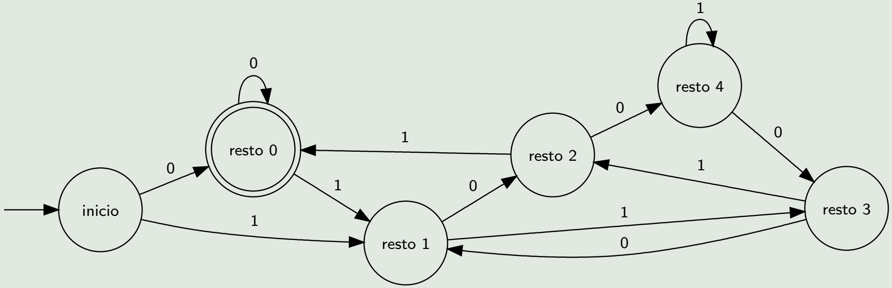


Ejercicio

Las cadenas de ceros y unos pueden interpretarse como representaciones binarias de los números naturales. Construya un AFD sobre $\Sigma = \{0, 1\}$ que acepte los múltiplos de 5.

Solución:

Cada estado se corresponde con los posibles restos de dividir entre 5.
Si llega un 0, se multiplica por dos; si llega un 1, se multiplica por dos y se suma uno.
Estas mismas operaciones se pueden hacer con los restos, ya que son congruentes en cualquier $\mathbb{Z}_n$ (clase de restos módulo n).

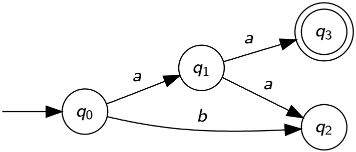


Contenidos

4 Autómata finito no determinista (AFN)

Introducimos ahora el concepto de *autómata finito no determinista*. Se trata de un autómata que presenta algún estado, el cual, ante un mismo símbolo de entrada, puede transitar a varios estados diferentes, en lugar de a uno sólo.

Ejemplo:

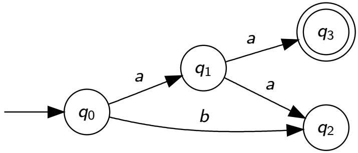


En este autómata, la cadena *aa* no se aceptaría a través de $q_0 \to q_1 \to q_2$, pero sí a través de $q_0 \to q_1 \to q_3$.

4 Autómata finito no determinista (AFN)

Introducimos ahora el concepto de *autómata finito no determinista*. Se trata de un autómata que presenta algún estado, el cual, ante un mismo símbolo de entrada, puede transitar a varios estados diferentes, en lugar de a uno sólo.

Ejemplo:



En este autómata, la cadena *aa* no se aceptaría a través de $q_0 \to q_1 \to q_2$, pero sí a través de $q_0 \to q_1 \to q_3$.

Definición

Un **autómata finito no determinista (AFN)** M es una colección de cinco elementos:

$$M = (Q, \Sigma, s, \Delta, F)$$

donde:

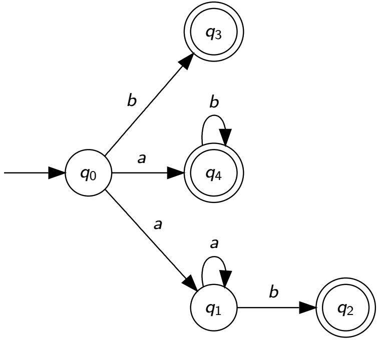
- Q, Σ, s y F son lo mismo que en un AFD,
- $\Delta : Q \times \Sigma \to 2^Q$ es la función de transición que determina el estado o estados siguientes para cada par (q, σ). Es decir, $\Delta(q, \sigma) \subseteq Q$.

4 Autómata finito no determinista (AFN)

Formas de representación de un AFN:

- Mediante un grafo: igual que los AFD,s.
- Mediante una tabla: igual que los AFD,s salvo que cada celda de la tabla contiene ahora un conjunto de estados.

Ejemplo:



Δ	a	b
$\rightarrow q_0$	$\{q_1, q_4\}$	$\{q_3\}$
q_1	$\{q_1\}$	$\{q_2\}$
$*\ q_2$	$\emptyset$	$\emptyset$
$*\ q_3$	$\emptyset$	$\emptyset$
$*\ q_4$	$\emptyset$	$\{q_4\}$

Faltan por definir los siguientes aspectos:

- **¿Cómo se realizan las transiciones?** Es decir, si desde un estado, con un mismo símbolo, se puede ir a varios estados, ¿eligimos uno o trabajamos con todos a la vez para considerar todos los posibles caminos de la cadena en el autómata?

4 Autómata finito no determinista (AFN)

Faltan por definir los siguientes aspectos:

- **¿Cómo se realizan las transiciones?** Es decir, si desde un estado, con un mismo símbolo, se puede ir a varios estados, ¿eligimos uno o trabajamos con todos a la vez para considerar todos los posibles caminos de la cadena en el autómata?

  La respuesta correcta es la segunda opción. Por tanto, hay que ampliar también la definición de Δ, para que pueda trabajar con varios estados y con varios símbolos a la vez:

$$\Delta(\{q_1, q_2, \ldots, q_k\}, \sigma) = \bigcup_{i=1}^{k} \Delta(q_i, \sigma)$$

$$\hat{\Delta}(\{s\}, w) = \Delta(\ldots\Delta(\Delta(\Delta(\{s\}, a_1), a_2), a_3)\ldots, a_n)$$

Faltan por definir los siguientes aspectos:

- **¿Cómo se realizan las transiciones?** Es decir, si desde un estado, con un mismo símbolo, se puede ir a varios estados, ¿eligimos uno o trabajamos con todos a la vez para considerar todos los posibles caminos de la cadena en el autómata?

  La respuesta correcta es la segunda opción. Por tanto, hay que ampliar también la definición de Δ, para que pueda trabajar con varios estados y con varios símbolos a la vez:

$$\Delta(\{q_1, q_2, \ldots, q_k\}, \sigma) = \bigcup_{i=1}^{k} \Delta(q_i, \sigma)$$

$$\hat{\Delta}(\{s\}, w) = \Delta(\ldots \Delta(\Delta(\Delta(\{s\}, a_1), a_2), a_3) \ldots, a_n)$$

- **¿Y cuál es la condición de aceptación?**

4 Autómata finito no determinista (AFN)

Faltan por definir los siguientes aspectos:

- **¿Cómo se realizan las transiciones?** Es decir, si desde un estado, con un mismo símbolo, se puede ir a varios estados, ¿eligimos uno o trabajamos con todos a la vez para considerar todos los posibles caminos de la cadena en el autómata?

  La respuesta correcta es la segunda opción. Por tanto, hay que ampliar también la definición de Δ, para que pueda trabajar con varios estados y con varios símbolos a la vez:

$$\Delta(\{q_1, q_2, \ldots, q_k\}, \sigma) = \bigcup_{i=1}^{k} \Delta(q_i, \sigma)$$

$$\hat{\Delta}(\{s\}, w) = \Delta(\ldots\Delta(\Delta(\Delta(\{s\}, a_1), a_2), a_3)\ldots, a_n)$$

- **¿Y cuál es la condición de aceptación?**

  Una cadena es aceptada si en el autómata existe al menos un camino de procesamiento para ella que termine en un estado final.

Definición

Dado $M = (Q, \Sigma, s, \Delta, F)$ un AFN arbitario, definimos el **lenguaje aceptado por** M como sigue:

$$L(M) = \{w \in \Sigma^* \mid \hat{\Delta}(\{s\}, w) \cap F \neq \emptyset\}$$

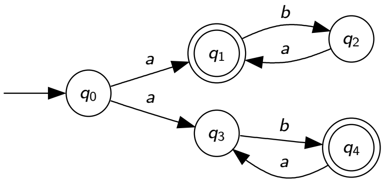
4 Autómata finito no determinista (AFN)

Definición

Dado $M = (Q, \Sigma, s, \Delta, F)$ un AFN arbitario, definimos el **lenguaje aceptado por** M como sigue:

$$L(M) = \{w \in \Sigma^* \mid \hat{\Delta}(\{s\}, w) \cap F \neq \emptyset\}$$

Ejemplo:



Las *configuraciones instantáneas* incluyen ahora un conjunto de estados, en lugar de un estado sólo:

$$\begin{array}{l}
(\{q_0\}, abab) \\
(\{q_1, q_3\}, bab) \\
(\{q_2, q_4\}, ab) \\
(\{q_1, q_3\}, b) \\
(\{q_2, q_4\}, \epsilon)
\end{array}$$

$$L(M) = a \cdot (ba)^* \cup (ab)^+$$

$$q_4 \in F \Rightarrow abab \in L(M)$$

Contenidos

5 Equivalencia entre AFN,s y AFD,s

Dado un AFD, ¿se puede construir un AFN equivalente?

Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

5 Equivalencia entre AFN,s y AFD,s

Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?

5 Equivalencia entre AFN,s y AFD,s

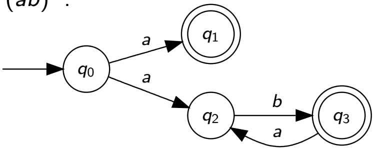
Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:

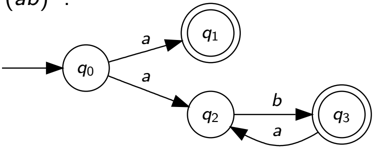


5 Equivalencia entre AFN,s y AFD,s

Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:



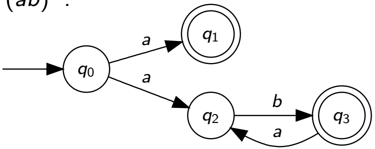
- AFD que acepta el mismo lenguaje:

5 Equivalencia entre AFN,s y AFD,s

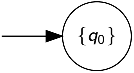
Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:



- AFD que acepta el mismo lenguaje:

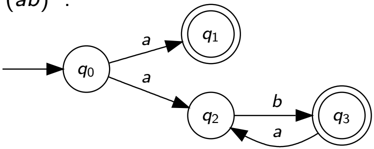


5 Equivalencia entre AFN,s y AFD,s

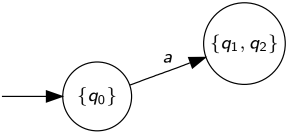
Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:



- AFD que acepta el mismo lenguaje:

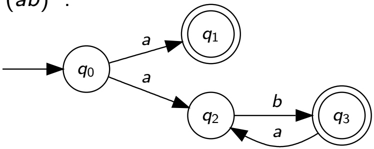


5 Equivalencia entre AFN,s y AFD,s

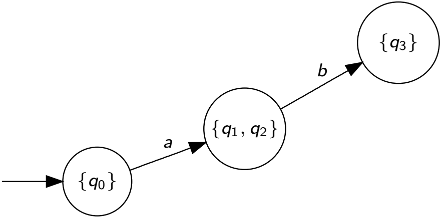
Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:



- AFD que acepta el mismo lenguaje:

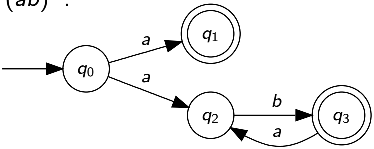


5 Equivalencia entre AFN,s y AFD,s

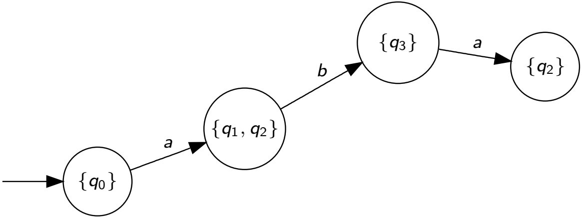
Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:



- AFD que acepta el mismo lenguaje:

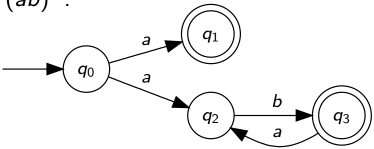


5 Equivalencia entre AFN,s y AFD,s

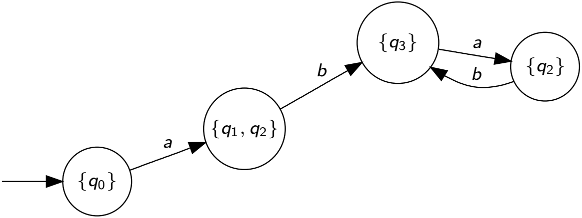
Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:



- AFD que acepta el mismo lenguaje:

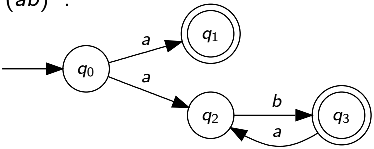


5 Equivalencia entre AFN,s y AFD,s

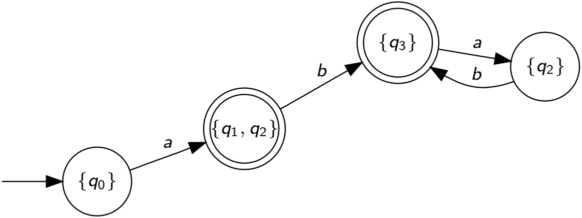
Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:



- AFD que acepta el mismo lenguaje:

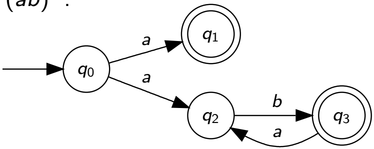


5 Equivalencia entre AFN,s y AFD,s

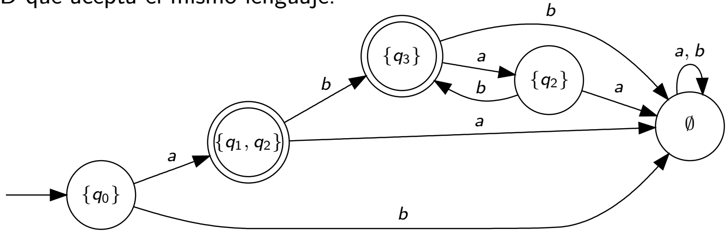
Dado un AFD, ¿se puede construir un AFN equivalente?
Sí, el propio AFD vale, ya que la función de transición del AFN es más general.

¿Y al revés? Es decir, dado un AFN, ¿se puede construir un AFD equivalente?
También es posible. La idea es identificar los no determinismos y crear estados que representen "varios sitios" a la vez:

- AFN que acepta $a \cup (ab)^+$:



- AFD que acepta el mismo lenguaje:



Teorema

Dado un AFN $M = (Q, \Sigma, s, \Delta, F)$, siempre existe un AFD $M' = (Q', \Sigma, s', \delta, F')$ tal que $L(M) = L(M')$, donde:

- $Q' = 2^Q$,
- $s' = \{s\}$,
- F' son todos los subconjuntos de Q que incluyen algún estado final de F,
- y $\delta : 2^Q \times \Sigma \to 2^Q$ es una función de transición tal que

$$\delta(\{q_1, q_2, \ldots, q_m\}, \sigma) = \{p_1, p_2, \ldots, p_k\}$$
$$\Leftrightarrow$$
$$\Delta(\{q_1, q_2, \ldots, q_m\}, \sigma) = \{p_1, p_2, \ldots, p_k\}$$

5 Equivalencia entre AFN,s y AFD,s

Teorema

Dado un AFN $M = (Q, \Sigma, s, \Delta, F)$, siempre existe un AFD $M' = (Q', \Sigma, s', \delta, F')$ tal que $L(M) = L(M')$, donde:

- $Q' = 2^Q$,
- $s' = \{s\}$,
- F' son todos los subconjuntos de Q que incluyen algún estado final de F,
- y $\delta : 2^Q \times \Sigma \to 2^Q$ es una función de transición tal que

$$\delta(\{q_1, q_2, \ldots, q_m\}, \sigma) = \{p_1, p_2, \ldots, p_k\}$$
$$\Leftrightarrow$$
$$\Delta(\{q_1, q_2, \ldots, q_m\}, \sigma) = \{p_1, p_2, \ldots, p_k\}$$

Ideas para la demostración: Para demostrar que $L(M) = L(M')$, hay que demostrar que para toda cadena $w \in \Sigma^*$ se cumple que

$$\hat{\delta}(s', w) = \{p_1, p_2, \ldots, p_j\} \Leftrightarrow \hat{\Delta}(s, w) = \{p_1, p_2, \ldots, p_j\}$$

es decir, M' acepta $w \Leftrightarrow M$ acepta w, lo cual puede hacerse por inducción en $|w|$. □

5 Equivalencia entre AFN,s y AFD,s

Es interesante tener en cuenta las siguientes consideraciones finales:

- ¿Este proceso termina siempre?

Es interesante tener en cuenta las siguientes consideraciones finales:

- ¿Este proceso termina siempre?
  Sí, porque el número de estados del AFN es finito. Y de hecho, el AFD resultante tendrá como mucho $2^{|Q|}$ estados y típicamente algunos menos.

5 Equivalencia entre AFN,s y AFD,s

Es interesante tener en cuenta las siguientes consideraciones finales:

- ¿Este proceso termina siempre?

  Sí, porque el número de estados del AFN es finito. Y de hecho, el AFD resultante tendrá como mucho $2^{|Q|}$ estados y típicamente algunos menos.

- ¿Qué ocurriría si aplicáramos este proceso a un AFD?

5 Equivalencia entre AFN,s y AFD,s

Es interesante tener en cuenta las siguientes consideraciones finales:

- ¿Este proceso termina siempre?

  Sí, porque el número de estados del AFN es finito. Y de hecho, el AFD resultante tendrá como mucho $2^{|Q|}$ estados y típicamente algunos menos.

- ¿Qué ocurriría si aplicáramos este proceso a un AFD?

  Obtendríamos el mismo autómata.

5 Equivalencia entre AFN,s y AFD,s

Es interesante tener en cuenta las siguientes consideraciones finales:

- ¿Este proceso termina siempre?

  Sí, porque el número de estados del AFN es finito. Y de hecho, el AFD resultante tendrá como mucho $2^{|Q|}$ estados y típicamente algunos menos.

- ¿Qué ocurriría si aplicáramos este proceso a un AFD?

  Obtendríamos el mismo autómata.

- ¿Qué implicaciones tiene este resultado con respecto a la capacidad de reconocimiento de lenguajes de los autómatas?

Es interesante tener en cuenta las siguientes consideraciones finales:

- ¿Este proceso termina siempre?

  Sí, porque el número de estados del AFN es finito. Y de hecho, el AFD resultante tendrá como mucho $2^{|Q|}$ estados y típicamente algunos menos.

- ¿Qué ocurriría si aplicáramos este proceso a un AFD?

  Obtendríamos el mismo autómata.

- ¿Qué implicaciones tiene este resultado con respecto a la capacidad de reconocimiento de lenguajes de los autómatas?

  Los AFN,s no son más potentes que los AFD,s con respecto a los lenguajes que aceptan. Ambos tipos de autómatas reconocen exactamente los mismos lenguajes.

  Simplemente ocurre que, para algún lenguaje, a la hora de construir el correspondiente autómata que lo acepte, puede resultar más sencillo construir un AFN que construir un AFD.

Contenidos

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Ampliamos aún más el no determinismo, para permitir que el autómata haga transiciones sin consumir ningún símbolo de la cadena de entrada.

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Ampliamos aún más el no determinismo, para permitir que el autómata haga transiciones sin consumir ningún símbolo de la cadena de entrada.

Definición

Un **autómata finito no determinista con ϵ-transiciones (AFN-ϵ)** M es una colección de cinco elementos:

$$M = (Q, \Sigma, s, \Delta, F)$$

donde:

- Q, Σ, s y F son lo mismo que en un AFN,
- $\Delta : Q \times (\Sigma \cup \{\epsilon\}) \to 2^Q$ es la función de transición que determina el estado o estados siguientes para cada par (q, σ) o (q, ϵ).

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Ampliamos aún más el no determinismo, para permitir que el autómata haga transiciones sin consumir ningún símbolo de la cadena de entrada.

Definición

Un **autómata finito no determinista con ϵ-transiciones (AFN-ϵ)** M es una colección de cinco elementos:

$$M = (Q, \Sigma, s, \Delta, F)$$

donde:

- Q, Σ, s y F son lo mismo que en un AFN,
- $\Delta : Q \times (\Sigma \cup \{\epsilon\}) \to 2^Q$ es la función de transición que determina el estado o estados siguientes para cada par (q, σ) o (q, ϵ).

Formas de representación:

- Mediante un grafo: igual que antes, salvo que ahora pueden aparecer arcos etiquetados con ϵ.
- Mediante una tabla: igual que antes, salvo que ahora aparece una nueva columna para el símbolo ϵ.

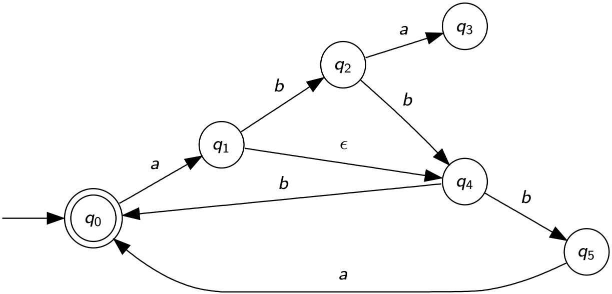
6 Autómata finito con ϵ-transiciones (AFN-ϵ)

La decisión de elegir o no una ϵ-transición se realiza de la misma forma que para cualquier transición múltiple de un AFN. Es decir, se basa en algo que no está completamente determinado, por lo que debemos considerar todas las alternativas.

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

La decisión de elegir o no una ϵ-transición se realiza de la misma forma que para cualquier transición múltiple de un AFN. Es decir, se basa en algo que no está completamente determinado, por lo que debemos considerar todas las alternativas.

Por ejemplo, el siguiente AFN-ϵ acepta, entre otras, la cadena *ab*:

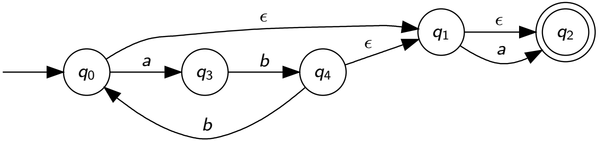


Después de procesar el símbolo a, alcanzamos q_1 y podemos elegir entre:

- consumir el símbolo b y llegar a q_2 (por ahí la cadena no se acepta),
- o bien pasar a q_4 sin cosumir nada y de ahí a q_0 con b (por aquí sí se acepta).

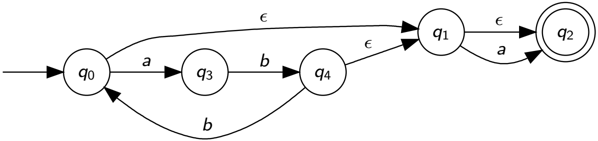
6 Autómata finito con ϵ-transiciones (AFN-ϵ)

También existen casos donde, al procesar una cadena, sería necesario considerar que antes y después de cada símbolo de entrada puede haber cualquier número de símbolos ϵ:



6 Autómata finito con ϵ-transiciones (AFN-ϵ)

También existen casos donde, al procesar una cadena, sería necesario considerar que antes y después de cada símbolo de entrada puede haber cualquier número de símbolos ϵ:



Por tanto, para saber exactamente a qué estados debemos transitar desde uno dado, es necesario definir su ϵ-*cierre*.

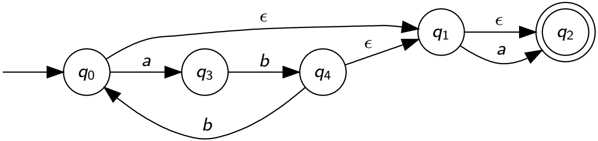
Definición

Dado $q \in Q$, un estado cualquiera de un AFN-ϵ, el **ϵ-cierre** de q se define como sigue:

$$\epsilon\text{-cierre}\,(q) = \{p \in Q \mid p \text{ es accesible desde } q \text{ mediante } \epsilon\text{-transiciones}\}$$

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

También existen casos donde, al procesar una cadena, sería necesario considerar que antes y después de cada símbolo de entrada puede haber cualquier número de símbolos ϵ:



Por tanto, para saber exactamente a qué estados debemos transitar desde uno dado, es necesario definir su ϵ-*cierre*.

Definición

Dado $q \in Q$, un estado cualquiera de un AFN-ϵ, el **ϵ-cierre** de q se define como sigue:

$$\epsilon\text{-cierre}\,(q) = \{p \in Q \mid p \text{ es accesible desde } q \text{ mediante } \epsilon\text{-transiciones}\}$$

En el ejemplo anterior:

$$\begin{array}{lll} \epsilon\text{-cierre}\,(q_0) = \{q_0, q_1, q_2\} & \epsilon\text{-cierre}\,(q_2) = \{q_2\} & \epsilon\text{-cierre}\,(q_4) = \{q_4, q_1, q_2\} \\ \epsilon\text{-cierre}\,(q_1) = \{q_1, q_2\} & \epsilon\text{-cierre}\,(q_3) = \{q_3\} & \end{array}$$

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Seguidamente, para ver qué estados se alcanzan, por ejemplo, desde el estado q_0 con el símbolo a, calculamos ϵ-cierre(q_0), transitamos desde todos esos estados con los arcos etiquetados con a y finalmente volvemos a calcular el ϵ-cierre de los nuevos destinos.

Seguidamente, para ver qué estados se alcanzan, por ejemplo, desde el estado q_0 con el símbolo a, calculamos ϵ-cierre (q_0), transitamos desde todos esos estados con los arcos etiquetados con a y finalmente volvemos a calcular el ϵ-cierre de los nuevos destinos.

Es necesario entonces extender el ϵ-cierre para conjuntos de estados:

$$\epsilon\text{-cierre}\,(\{q_1, q_2, \ldots, q_k\}) = \bigcup_{i=1}^{k} \epsilon\text{-cierre}\,(q_i)$$

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Seguidamente, para ver qué estados se alcanzan, por ejemplo, desde el estado q_0 con el símbolo a, calculamos ϵ-cierre(q_0), transitamos desde todos esos estados con los arcos etiquetados con a y finalmente volvemos a calcular el ϵ-cierre de los nuevos destinos.

Es necesario entonces extender el ϵ-cierre para conjuntos de estados:

$$\epsilon\text{-cierre}\left(\{q_1, q_2, \ldots, q_k\}\right) = \bigcup_{i=1}^{k} \epsilon\text{-cierre}\left(q_i\right)$$

Y si definimos también la siguiente función:

$$d(q, \sigma) = \{p \in Q \mid \text{hay transiciones de } q \text{ a } p \text{ etiquetadas con } \sigma\}$$

y la extendemos para conjuntos de estados:

$$d(\{q_1, q_2, \ldots, q_k\}) = \bigcup_{i=1}^{k} d(q_i)$$

entonces la **función de transición de un AFN-ϵ**, $\forall q \in Q$ y $\forall \sigma \in \Sigma$, puede venir dada por:

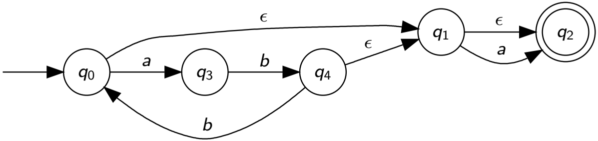
$$\Delta(q, \sigma) = \epsilon\text{-cierre}\left[d(\epsilon\text{-cierre}\left(q\right), \sigma)\right]$$

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Esta idea podemos ponerla en práctica al generar las distintas configuraciones instantáneas correspondientes al proceso de reconocimiento de una cadena de entrada.

Esta idea podemos ponerla en práctica al generar las distintas configuraciones instantáneas correspondientes al proceso de reconocimiento de una cadena de entrada.

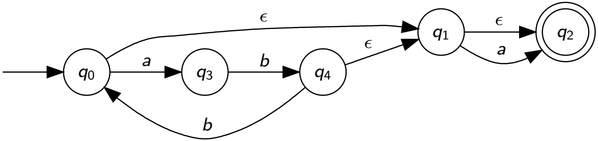
Por ejemplo, consideremos el AFN-ϵ visto anteriormente y los ϵ-cierres de sus estados:



ϵ-cierre $(q_0) = \{q_0, q_1, q_2\}$ $\quad$ ϵ-cierre $(q_2) = \{q_2\}$ $\quad$ ϵ-cierre $(q_4) = \{q_4, q_1, q_2\}$
ϵ-cierre $(q_1) = \{q_1, q_2\}$ $\quad$ ϵ-cierre $(q_3) = \{q_3\}$

Esta idea podemos ponerla en práctica al generar las distintas configuraciones instantáneas correspondientes al proceso de reconocimiento de una cadena de entrada.

Por ejemplo, consideremos el AFN-ϵ visto anteriormente y los ϵ-cierres de sus estados:



ϵ-cierre $(q_0) = \{q_0, q_1, q_2\}$ $\quad$ ϵ-cierre $(q_2) = \{q_2\}$ $\quad$ ϵ-cierre $(q_4) = \{q_4, q_1, q_2\}$

ϵ-cierre $(q_1) = \{q_1, q_2\}$ $\quad$ ϵ-cierre $(q_3) = \{q_3\}$

La secuencia de configuraciones instantáneas por las que pasa este autómata al procesar la cadena *aba* sería la siguiente:

$$
\begin{array}{lcl}
(\epsilon\text{-cierre}\,\{q_0\}, aba) & = & (\{q_0, q_1, q_2\}, aba) \\
(\epsilon\text{-cierre}\,\{q_3, q_2\}, ba) & = & (\{q_3, q_2\}, ba) \\
(\epsilon\text{-cierre}\,\{q_4\}, a) & = & (\{q_4, q_1, q_2\}, a) \\
(\epsilon\text{-cierre}\,\{q_2\}, \epsilon) & = & (\{q_2\}, \epsilon)
\end{array}
$$

$$q_2 \in F \Rightarrow aba \in L(M)$$

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Y por otra parte, el hecho de que podamos precalcular a priori los valores de la nueva función de transición para todos los pares (q, σ), es decir, para todas combinaciones posibles de estados de Q y símbolos de Σ, da lugar al siguiente resultado, el cual constituye en sí mismo un procedimiento de conversión de un AFN-ϵ en un AFN.

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Y por otra parte, el hecho de que podamos precalcular a priori los valores de la nueva función de transición para todos los pares (q, σ), es decir, para todas combinaciones posibles de estados de Q y símbolos de Σ, da lugar al siguiente resultado, el cual constituye en sí mismo un procedimiento de conversión de un AFN-ϵ en un AFN.

Teorema

Dado un AFN-ϵ $M = (Q, \Sigma, s, \Delta, F)$, siempre existe un AFN sin ϵ-transiciones $M' = (Q, \Sigma, s, \Delta', F')$ tal que $L(M) = L(M')$, donde:

- $\Delta'(q, \sigma) = \epsilon\text{-cierre}\,[d(\epsilon\text{-cierre}\,(q), \sigma)]$

  Es decir, la nueva función de transición es la composición de ϵ-cierre, d y ϵ-cierre que vimos anteriormente.

- $F' = F \cup \{q \in Q \mid \epsilon\text{-cierre}\,(q) \cap F \neq \emptyset\}$

  Es decir, los nuevos estados finales son los de F más los que pueden alcanzar algún estado de F mediante ϵ-transiciones. □

6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Y por otra parte, el hecho de que podamos precalcular a priori los valores de la nueva función de transición para todos los pares (q, σ), es decir, para todas combinaciones posibles de estados de Q y símbolos de Σ, da lugar al siguiente resultado, el cual constituye en sí mismo un procedimiento de conversión de un AFN-ϵ en un *AFN*.

Teorema

Dado un AFN-ϵ $M = (Q, \Sigma, s, \Delta, F)$, siempre existe un AFN sin ϵ-transiciones $M' = (Q, \Sigma, s, \Delta', F')$ tal que $L(M) = L(M')$, donde:

- $\Delta'(q, \sigma) = \epsilon\text{-cierre}\,[d(\epsilon\text{-cierre}\,(q), \sigma)]$

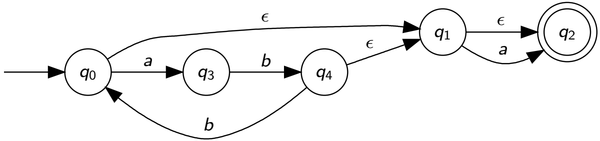
  Es decir, la nueva función de transición es la composición de ϵ-cierre, d y ϵ-cierre que vimos anteriormente.

- $F' = F \cup \{q \in Q \mid \epsilon\text{-cierre}\,(q) \cap F \neq \emptyset\}$

  Es decir, los nuevos estados finales son los de F más los que pueden alcanzar algún estado de F mediante ϵ-transiciones. $\square$

Una vez más, podemos afirmar que **los AFN-ϵ,s no son más potentes que los AFN,s** con respecto a los lenguajes que aceptan, sino que ambos tipos de autómatas reconocen exactamente los mismos lenguajes.

Por ejemplo, para el AFN-ϵ visto anteriormente, tenemos que:



$$\begin{aligned}
\epsilon\text{-c}\,(q_0) &= \{q_0, q_1, q_2\}\\
\epsilon\text{-c}\,(q_1) &= \{q_1, q_2\}\\
\epsilon\text{-c}\,(q_2) &= \{q_2\}\\
\epsilon\text{-c}\,(q_3) &= \{q_3\}\\
\epsilon\text{-c}\,(q_4) &= \{q_4, q_1, q_2\}
\end{aligned}$$

$$\Delta'(q_0, a) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_0), a)] = \epsilon\text{-c}\,[d(\{q_0, q_1, q_2\}, a)] = \epsilon\text{-c}\,(\{q_3, q_2\}) = \{q_3, q_2\}$$
$$\Delta'(q_0, b) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_0), b)] = \epsilon\text{-c}\,[d(\{q_0, q_1, q_2\}, b)] = \epsilon\text{-c}\,(\emptyset) = \emptyset$$

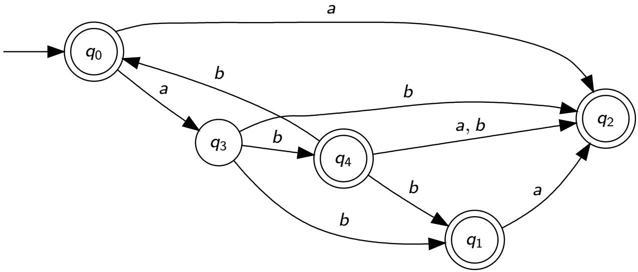
$$\Delta'(q_1, a) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_1), a)] = \epsilon\text{-c}\,[d(\{q_1, q_2\}, a)] = \epsilon\text{-c}\,(\{q_2\}) = \{q_2\}$$
$$\Delta'(q_1, b) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_1), b)] = \epsilon\text{-c}\,[d(\{q_1, q_2\}, b)] = \epsilon\text{-c}\,(\emptyset) = \emptyset$$

$$\Delta'(q_2, a) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_2), a)] = \epsilon\text{-c}\,[d(\{q_2\}, a)] = \epsilon\text{-c}\,(\emptyset) = \emptyset$$
$$\Delta'(q_2, b) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_2), b)] = \epsilon\text{-c}\,[d(\{q_2\}, b)] = \epsilon\text{-c}\,(\emptyset) = \emptyset$$

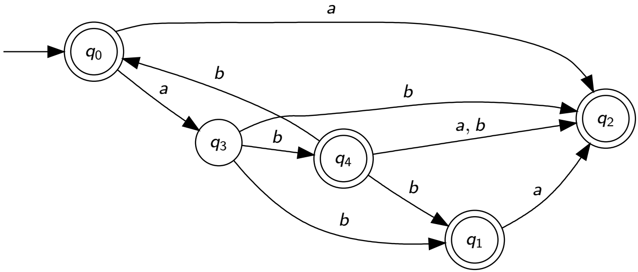
$$\Delta'(q_3, a) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_3), a)] = \epsilon\text{-c}\,[d(\{q_3\}, a)] = \epsilon\text{-c}\,(\emptyset) = \emptyset$$
$$\Delta'(q_3, b) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_3), b)] = \epsilon\text{-c}\,[d(\{q_3\}, b)] = \epsilon\text{-c}\,(\{q_4\}) = \{q_4, q_1, q_2\}$$

$$\Delta'(q_4, a) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_4), a)] = \epsilon\text{-c}\,[d(\{q_4, q_1, q_2\}, a)] = \epsilon\text{-c}\,(\{q_2\}) = \{q_2\}$$
$$\Delta'(q_4, b) = \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_4), b)] = \epsilon\text{-c}\,[d(\{q_4, q_1, q_2\}, b)] = \epsilon\text{-c}\,(\{q_0\}) = \{q_0, q_1, q_2\}$$

Por tanto, el AFN equivalente al AFN-ϵ visto anteriormente sería el siguiente:

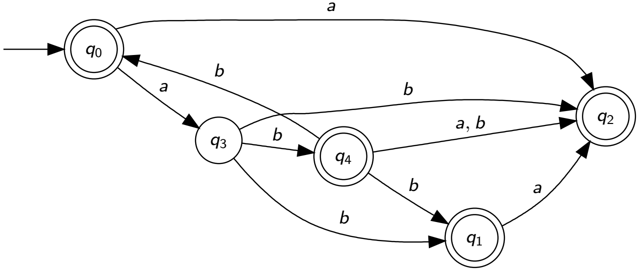


Por tanto, el AFN equivalente al AFN-ϵ visto anteriormente sería el siguiente:

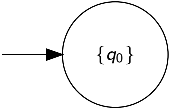


Y si lo convertimos en AFD, obtenemos:

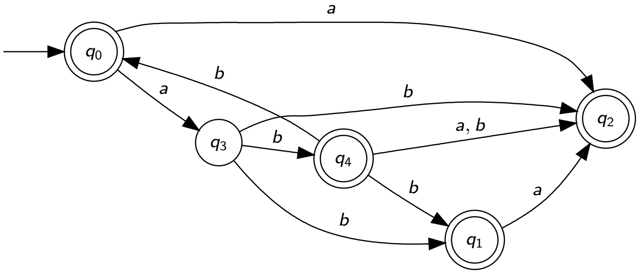
Por tanto, el AFN equivalente al AFN-ϵ visto anteriormente sería el siguiente:



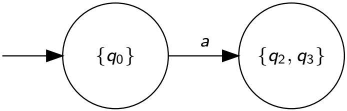
Y si lo convertimos en AFD, obtenemos:



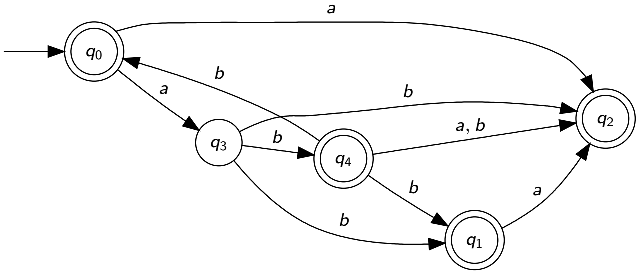
Por tanto, el AFN equivalente al AFN-ϵ visto anteriormente sería el siguiente:



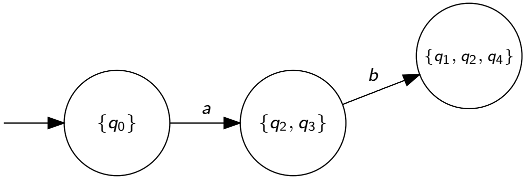
Y si lo convertimos en AFD, obtenemos:



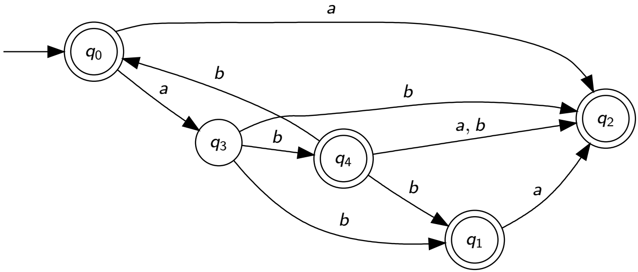
Por tanto, el AFN equivalente al AFN-ϵ visto anteriormente sería el siguiente:



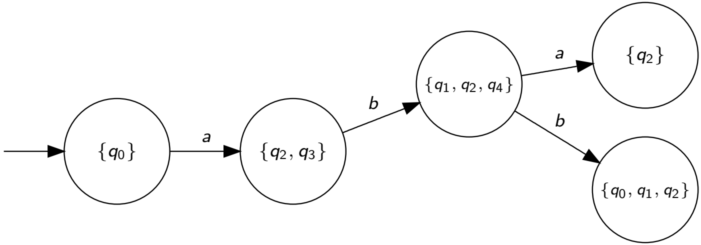
Y si lo convertimos en AFD, obtenemos:



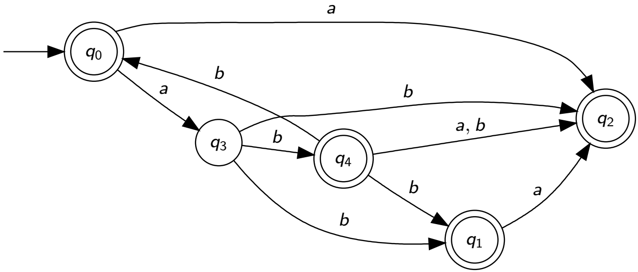
Por tanto, el AFN equivalente al AFN-ϵ visto anteriormente sería el siguiente:



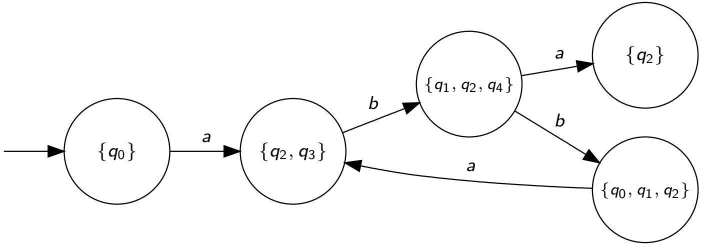
Y si lo convertimos en AFD, obtenemos:



Por tanto, el AFN equivalente al AFN-ϵ visto anteriormente sería el siguiente:

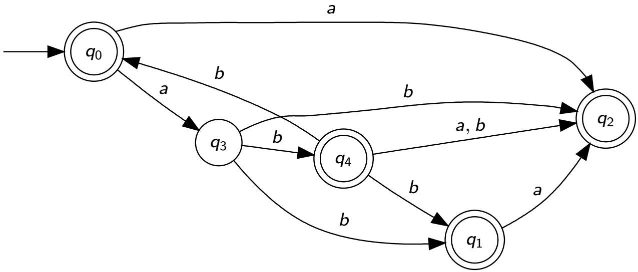


Y si lo convertimos en AFD, obtenemos:

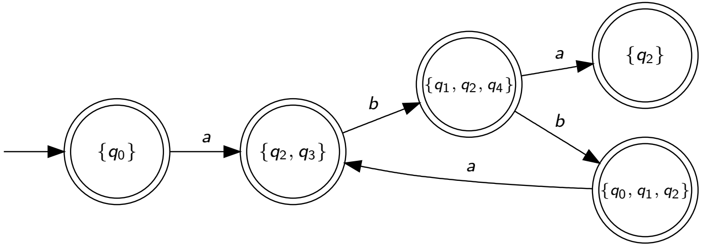


6 Autómata finito con ϵ-transiciones (AFN-ϵ)

Por tanto, el AFN equivalente al AFN-ϵ visto anteriormente sería el siguiente:

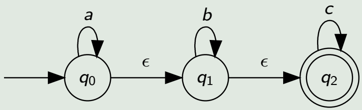


Y si lo convertimos en AFD, obtenemos:



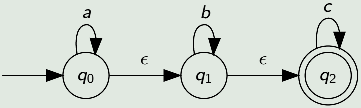
Ejercicio

Convierta en AFD el siguiente AFN-ϵ:



Ejercicio

Convierta en AFD el siguiente AFN-ϵ:



Solución:

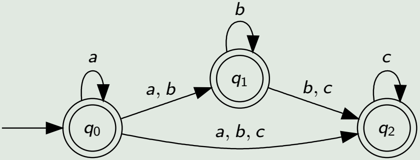
$$\epsilon\text{-c}\,(q_0) = \{q_0, q_1, q_2\} \qquad \epsilon\text{-c}\,(q_1) = \{q_1, q_2\} \qquad \epsilon\text{-c}\,(q_2) = \{q_2\}$$

$$\begin{aligned}
\Delta'(q_0, a) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_0), a)] = \epsilon\text{-c}\,[d(\{q_0, q_1, q_2\}, a)] = \epsilon\text{-c}\,(\{q_0\}) = \{q_0, q_1, q_2\} \\
\Delta'(q_0, b) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_0), b)] = \epsilon\text{-c}\,[d(\{q_0, q_1, q_2\}, b)] = \epsilon\text{-c}\,(\{q_1\}) = \{q_1, q_2\} \\
\Delta'(q_0, c) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_0), c)] = \epsilon\text{-c}\,[d(\{q_0, q_1, q_2\}, c)] = \epsilon\text{-c}\,(\{q_2\}) = \{q_2\} \\
\Delta'(q_1, a) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_1), a)] = \epsilon\text{-c}\,[d(\{q_1, q_2\}, a)] = \epsilon\text{-c}\,(\emptyset) = \emptyset \\
\Delta'(q_1, b) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_1), b)] = \epsilon\text{-c}\,[d(\{q_1, q_2\}, b)] = \epsilon\text{-c}\,(\{q_1\}) = \{q_1, q_2\} \\
\Delta'(q_1, c) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_1), c)] = \epsilon\text{-c}\,[d(\{q_1, q_2\}, c)] = \epsilon\text{-c}\,(\{q_2\}) = \{q_2\} \\
\Delta'(q_2, a) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_2), a)] = \epsilon\text{-c}\,[d(\{q_2\}, a)] = \epsilon\text{-c}\,(\emptyset) = \emptyset \\
\Delta'(q_2, b) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_2), b)] = \epsilon\text{-c}\,[d(\{q_2\}, b)] = \epsilon\text{-c}\,(\emptyset) = \emptyset \\
\Delta'(q_2, c) &= \epsilon\text{-c}\,[d(\epsilon\text{-c}\,(q_2), c)] = \epsilon\text{-c}\,[d(\{q_2\}, c)] = \epsilon\text{-c}\,(\{q_2\}) = \{q_2\}
\end{aligned}$$

.../...

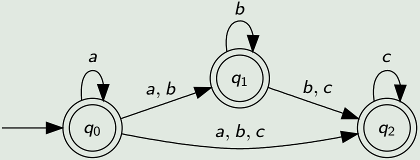
Ejercicio (continuación)

Por tanto, el AFN equivalente sería el siguiente:



Ejercicio (continuación)

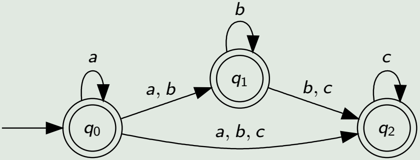
Por tanto, el AFN equivalente sería el siguiente:



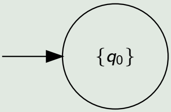
Y si lo convertimos en AFD, obtenemos:

Ejercicio (continuación)

Por tanto, el AFN equivalente sería el siguiente:

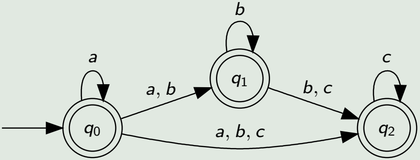


Y si lo convertimos en AFD, obtenemos:

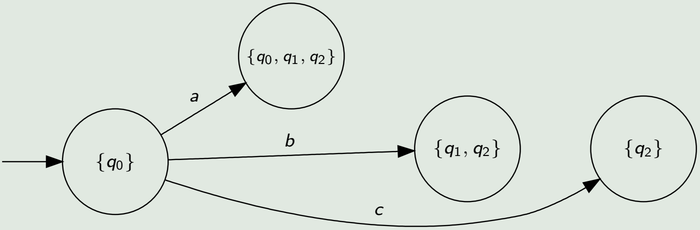


Ejercicio (continuación)

Por tanto, el AFN equivalente sería el siguiente:

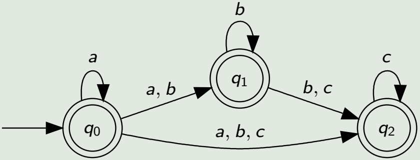


Y si lo convertimos en AFD, obtenemos:

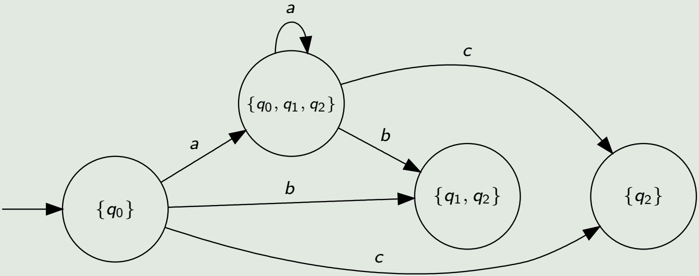


Ejercicio (continuación)

Por tanto, el AFN equivalente sería el siguiente:

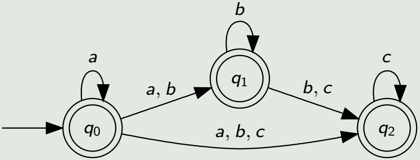


Y si lo convertimos en AFD, obtenemos:

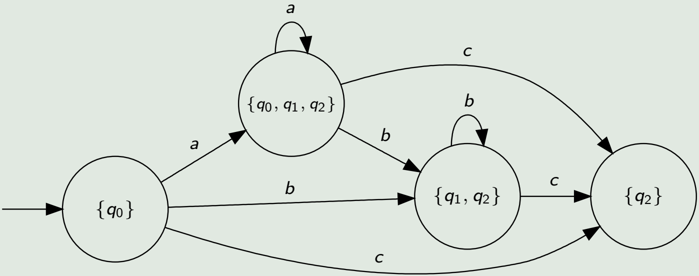


Ejercicio (continuación)

Por tanto, el AFN equivalente sería el siguiente:

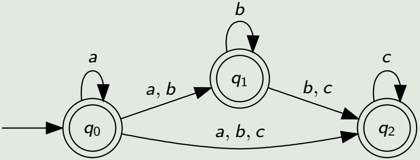


Y si lo convertimos en AFD, obtenemos:

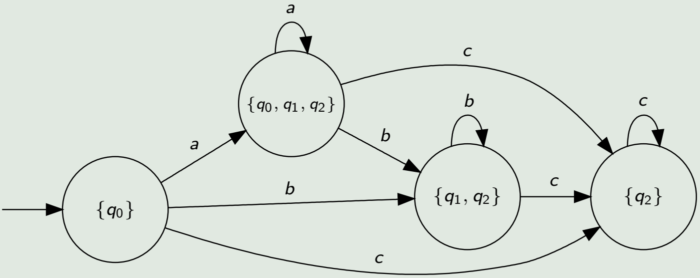


Ejercicio (continuación)

Por tanto, el AFN equivalente sería el siguiente:

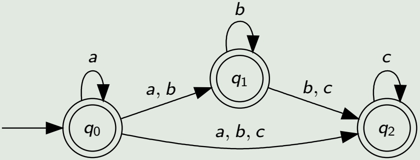


Y si lo convertimos en AFD, obtenemos:

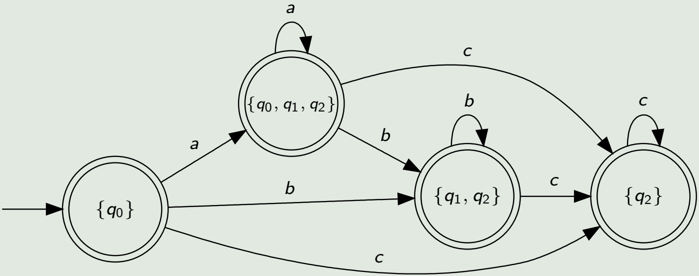


Ejercicio (continuación)

Por tanto, el AFN equivalente sería el siguiente:



Y si lo convertimos en AFD, obtenemos:



Contenidos

7 Autómatas finitos y expresiones regulares

Hasta ahora hemos tratado la relación entre autómatas finitos y expresiones regulares de una manera intuitiva. En esta sección, formalizaremos dicha relación.

7 Autómatas finitos y expresiones regulares

Hasta ahora hemos tratado la relación entre autómatas finitos y expresiones regulares de una manera intuitiva. En esta sección, formalizaremos dicha relación.

Primeramente, veremos cómo, **dada cualquier expresión regular r, se puede construir un AF que acepte** $L(r)$.

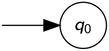
7 Autómatas finitos y expresiones regulares

Hasta ahora hemos tratado la relación entre autómatas finitos y expresiones regulares de una manera intuitiva. En esta sección, formalizaremos dicha relación.

Primeramente, veremos cómo, **dada cualquier expresión regular** r**, se puede construir un AF que acepte** $L(r)$.

Para los **casos base de una expresión regular**, tenemos que:

- Un AF que acepte la expresión regular $\emptyset$ podría ser el siguiente:



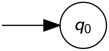
7 Autómatas finitos y expresiones regulares

Hasta ahora hemos tratado la relación entre autómatas finitos y expresiones regulares de una manera intuitiva. En esta sección, formalizaremos dicha relación.

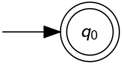
Primeramente, veremos cómo, **dada cualquier expresión regular r, se puede construir un AF que acepte $L(r)$.**

Para los **casos base de una expresión regular**, tenemos que:

- Un AF que acepte la expresión regular $\emptyset$ podría ser el siguiente:



- Un AF que acepte la expresión regular ϵ podría ser el siguiente:



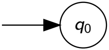
7 Autómatas finitos y expresiones regulares

Hasta ahora hemos tratado la relación entre autómatas finitos y expresiones regulares de una manera intuitiva. En esta sección, formalizaremos dicha relación.

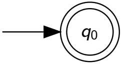
Primeramente, veremos cómo, **dada cualquier expresión regular r, se puede construir un AF que acepte** $L(r)$.

Para los **casos base de una expresión regular**, tenemos que:

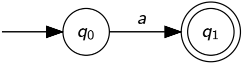
- Un AF que acepte la expresión regular $\emptyset$ podría ser el siguiente:



- Un AF que acepte la expresión regular ϵ podría ser el siguiente:



- Un AF que acepte la expresión regular a, para cualquier símbolo a del alfabeto de entrada Σ, podría ser el siguiente:



7 Autómatas finitos y expresiones regulares

Para los **casos recursivos de una expresión regular**, supongamos que tenemos dos espresiones regulares r y s, tales que

$$L(r) = L(M_1) \text{ donde } M_1 = (Q_1, \Sigma_1, s_1, \Delta_1, F_1)$$
$$L(s) = L(M_2) \text{ donde } M_2 = (Q_2, \Sigma_2, s_2, \Delta_2, F_2)$$

entonces:

Para los **casos recursivos de una expresión regular**, supongamos que tenemos dos espresiones regulares r y s, tales que

$$L(r) = L(M_1) \text{ donde } M_1 = (Q_1, \Sigma_1, s_1, \Delta_1, F_1)$$
$$L(s) = L(M_2) \text{ donde } M_2 = (Q_2, \Sigma_2, s_2, \Delta_2, F_2)$$

entonces:

- Un AF M que acepte la expresión regular $r \cup s$, es decir, tal que

$$L(M) = L(M_1) \cup L(M_2) = L(r) \cup L(s)$$

  podría construirse como sigue:

$$M = (Q_1 \cup Q_2 \cup \{s\},\ \Sigma_1 \cup \Sigma_2,\ s,\ \Delta,\ F_1 \cup F_2)$$

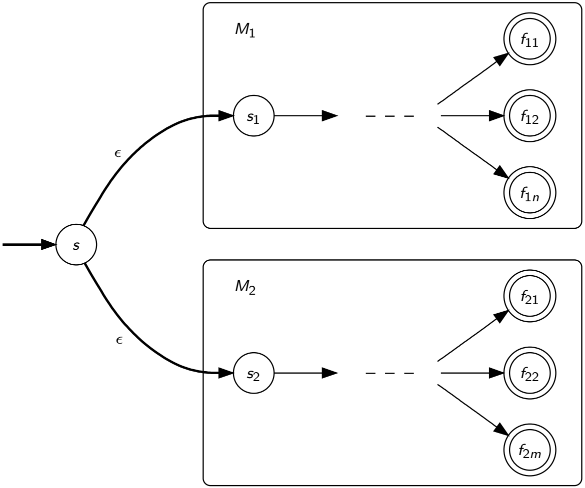
  donde:

$$\Delta(q, a) = \begin{cases} \Delta_1(q, a) & \text{si } q \in Q_1, a \in \Sigma_1 \\ \Delta_2(q, a) & \text{si } q \in Q_2, a \in \Sigma_2 \end{cases}$$

  y además

$$\Delta(s, \epsilon) = \{s_1, s_2\}$$

Gráficamente sería así:



- Un AF M que acepte la expresión regular $r \cdot s$, es decir, tal que

$$L(M) = L(M_1) \cdot L(M_2) = L(r) \cdot L(s)$$

podría construirse como sigue:

$$M = (Q_1 \cup Q_2,\ \Sigma_1 \cup \Sigma_2,\ s_1,\ \Delta,\ F_2)$$

donde $\Delta(q, a)$ es lo mismo que antes y además $\Delta(q, \epsilon) = \{s_2\},\ \forall q \in F_1$.

- Un AF M que acepte la expresión regular $r \cdot s$, es decir, tal que

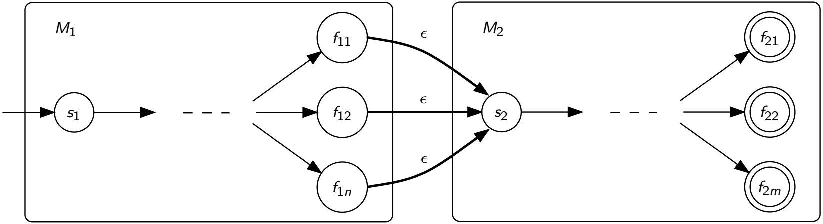
$$L(M) = L(M_1) \cdot L(M_2) = L(r) \cdot L(s)$$

podría construirse como sigue:

$$M = (Q_1 \cup Q_2,\ \Sigma_1 \cup \Sigma_2,\ s_1,\ \Delta,\ F_2)$$

donde $\Delta(q, a)$ es lo mismo que antes y además $\Delta(q, \epsilon) = \{s_2\},\ \forall q \in F_1$.

Gráficamente sería así:



7 Autómatas finitos y expresiones regulares

- Un AF M que acepte la expresión regular r^*, es decir, tal que

$$L(M) = (L(M_1))^* = (L(r))^*$$

podría construirse como sigue:

$$M = (Q_1 \cup \{s\},\ \Sigma_1,\ s,\ \Delta,\ \{s\})$$

donde $\Delta = \Delta_1$ y además $\Delta(s, \epsilon) = \{s_1\}$ y $\Delta(q, \epsilon) = \{s\}$, $\forall q \in F_1$.

- Un AF M que acepte la expresión regular r^*, es decir, tal que

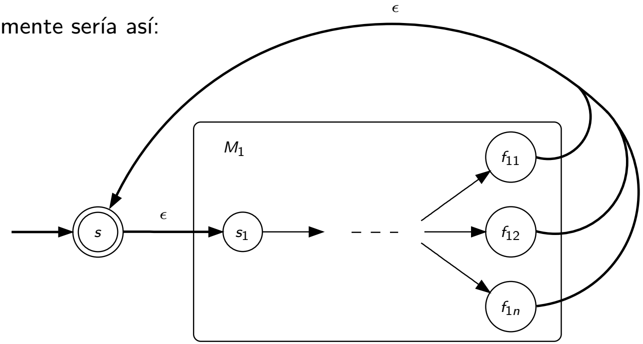
$$L(M) = (L(M_1))^* = (L(r))^*$$

podría construirse como sigue:

$$M = (Q_1 \cup \{s\},\ \Sigma_1,\ s,\ \Delta,\ \{s\})$$

donde $\Delta = \Delta_1$ y además $\Delta(s, \epsilon) = \{s_1\}$ y $\Delta(q, \epsilon) = \{s\}$, $\forall q \in F_1$.

Gráficamente sería así:



Un ejemplo completo:

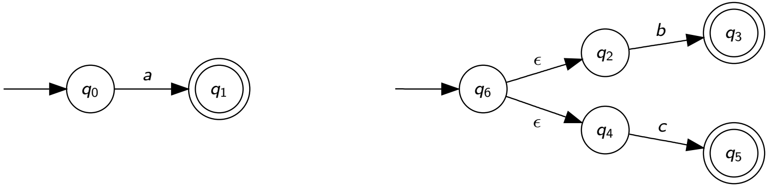
- El AFN-ϵ que acepta el lenguaje denotado por la expresión regular $a \cdot (b \cup c)^*$ podría construirse así:

Un ejemplo completo:

- El AFN-ϵ que acepta el lenguaje denotado por la expresión regular $a \cdot (b \cup c)^*$ podría construirse así:

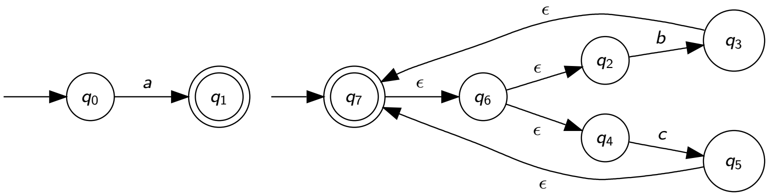
Un ejemplo completo:

- El AFN-ϵ que acepta el lenguaje denotado por la expresión regular $a \cdot (b \cup c)^*$ podría construirse así:



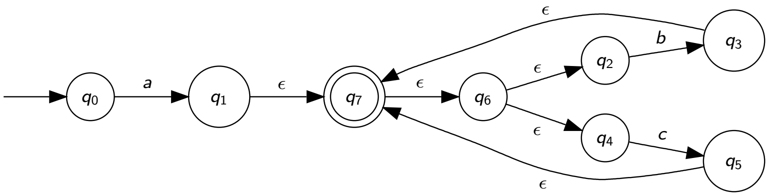
Un ejemplo completo:

- El AFN-ϵ que acepta el lenguaje denotado por la expresión regular $a \cdot (b \cup c)^*$ podría construirse así:



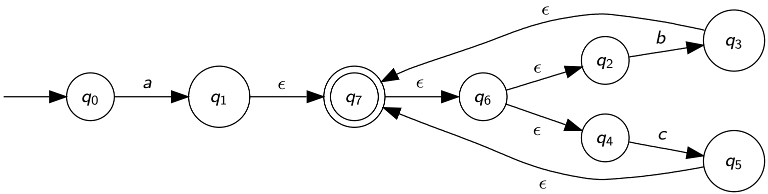
Un ejemplo completo:

- El AFN-ϵ que acepta el lenguaje denotado por la expresión regular $a \cdot (b \cup c)^*$ podría construirse así:



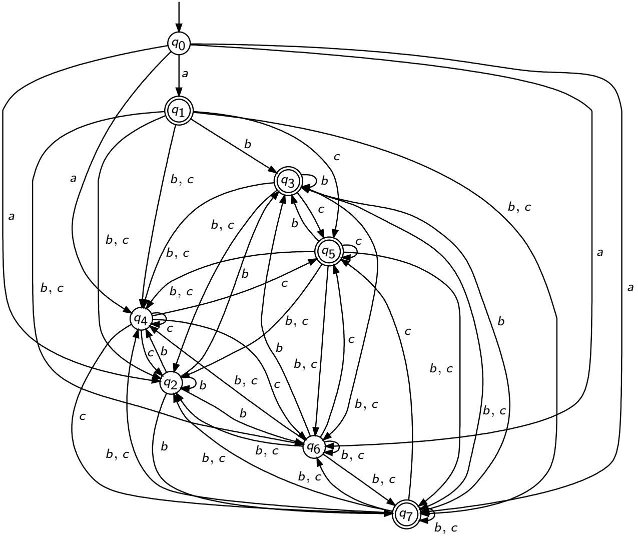
Un ejemplo completo:

- El AFN-ϵ que acepta el lenguaje denotado por la expresión regular $a \cdot (b \cup c)^*$ podría construirse así:

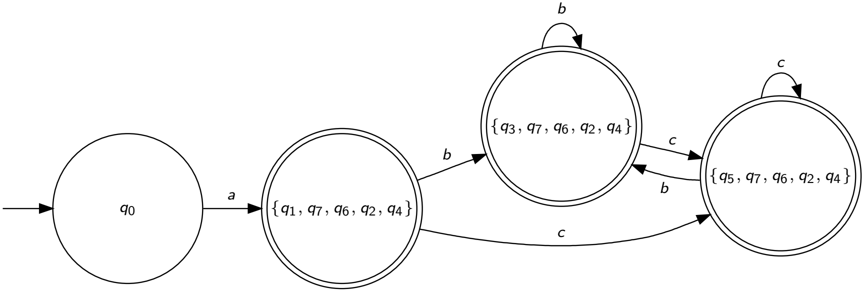


(8 estados y 9 arcos)

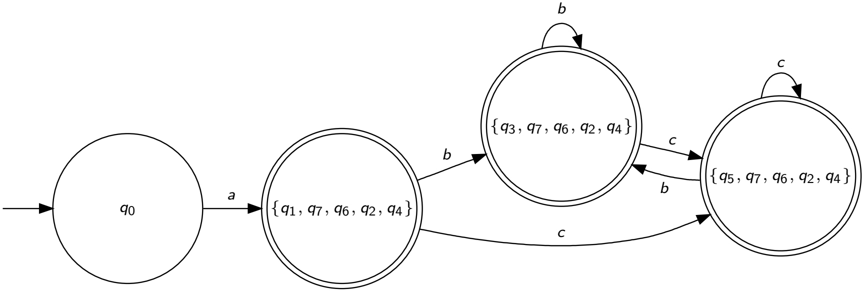
- El AFN equivalente al AFN-ϵ anterior es éste (8 estados y 65 arcos):



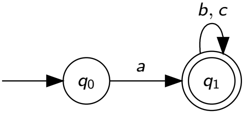
- El AFD equivalente al AFN anterior es éste (4 estados y 7 arcos):



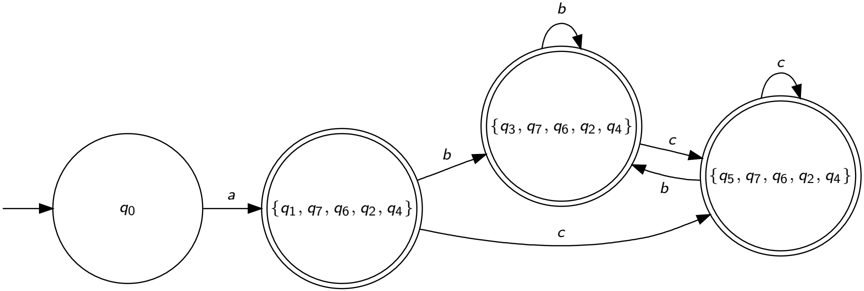
- El AFD equivalente al AFN anterior es éste (4 estados y 7 arcos):



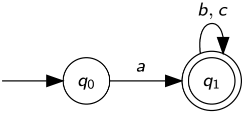
- Y el AFD mínimo equivalente al AFD anterior es éste (2 estados y 3 arcos):



- El AFD equivalente al AFN anterior es éste (4 estados y 7 arcos):



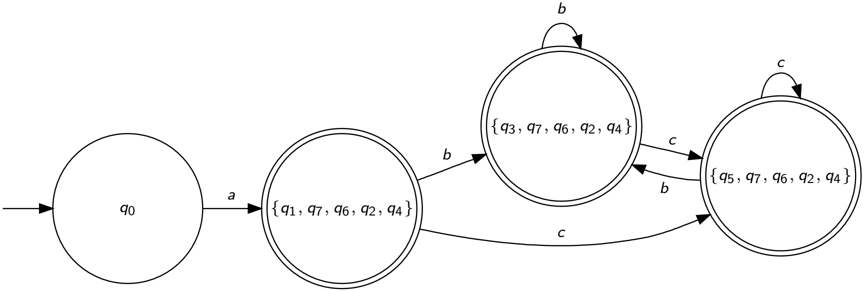
- Y el AFD mínimo equivalente al AFD anterior es éste (2 estados y 3 arcos):



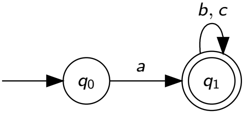
  – Un AFD M es mínimo cuando no existe ningún otro AFD M' con menos estados o menos arcos tal que $L(M') = L(M)$.

7 Autómatas finitos y expresiones regulares

- El AFD equivalente al AFN anterior es éste (4 estados y 7 arcos):



- Y el AFD mínimo equivalente al AFD anterior es éste (2 estados y 3 arcos):



  – Un AFD M es mínimo cuando no existe ningún otro AFD M' con menos estados o menos arcos tal que $L(M') = L(M)$.
  – Los algoritmos de minimización identifican clases de estados equivalentes y, para cada clase, mantienen únicamente su estado representante.

A continuación veremos cómo, **dado cualquier AF** M**, se puede construir una expresión regular** r**, tal que** $L(r) = L(M)$.

A continuación veremos cómo, **dado cualquier AF M, se puede construir una expresión regular r, tal que** $L(r) = L(M)$.

Dado cualquier AF $M = (Q, \Sigma, q_0, \Delta, F)$, definimos:

$$A_i = \{w \in \Sigma^* \mid \hat{\Delta}(q_i, w) \cap F \neq \emptyset\}$$

es decir, A_i contiene todas las cadenas que desde q_i nos llevan a algún estado final.

7 Autómatas finitos y expresiones regulares

A continuación veremos cómo, **dado cualquier AF M, se puede construir una expresión regular r, tal que** $L(r) = L(M)$.

Dado cualquier AF $M = (Q, \Sigma, q_0, \Delta, F)$, definimos:

$$A_i = \{w \in \Sigma^* \mid \hat{\Delta}(q_i, w) \cap F \neq \emptyset\}$$

es decir, A_i contiene todas las cadenas que desde q_i nos llevan a algún estado final.

Por lo tanto, obsérvese que:

- $A_0 = L(M)$.

7 Autómatas finitos y expresiones regulares

A continuación veremos cómo, **dado cualquier AF M, se puede construir una expresión regular r, tal que** $L(r) = L(M)$.

Dado cualquier AF $M = (Q, \Sigma, q_0, \Delta, F)$, definimos:

$$A_i = \{w \in \Sigma^* \mid \hat{\Delta}(q_i, w) \cap F \neq \emptyset\}$$

es decir, A_i contiene todas las cadenas que desde q_i nos llevan a algún estado final.

Por lo tanto, obsérvese que:

- $A_0 = L(M)$.
- Es posible que algún $A_i = \emptyset$.

A continuación veremos cómo, **dado cualquier AF M, se puede construir una expresión regular r, tal que** $L(r) = L(M)$.

Dado cualquier AF $M = (Q, \Sigma, q_0, \Delta, F)$, definimos:

$$A_i = \{w \in \Sigma^* \mid \hat{\Delta}(q_i, w) \cap F \neq \emptyset\}$$

es decir, A_i contiene todas las cadenas que desde q_i nos llevan a algún estado final.

Por lo tanto, obsérvese que:

- $A_0 = L(M)$.
- Es posible que algún $A_i = \emptyset$.
- Si $q_i \in F$, entonces $\epsilon \in A_i$.

7 Autómatas finitos y expresiones regulares

A continuación veremos cómo, **dado cualquier AF M, se puede construir una expresión regular r, tal que** $L(r) = L(M)$.

Dado cualquier AF $M = (Q, \Sigma, q_0, \Delta, F)$, definimos:

$$A_i = \{w \in \Sigma^* \mid \hat{\Delta}(q_i, w) \cap F \neq \emptyset\}$$

es decir, A_i contiene todas las cadenas que desde q_i nos llevan a algún estado final.

Por lo tanto, obsérvese que:

- $A_0 = L(M)$.
- Es posible que algún $A_i = \emptyset$.
- Si $q_i \in F$, entonces $\epsilon \in A_i$.
- En general, $A_i = \bigcup_j \{\sigma\, A_j \mid q_j \in \Delta(q_i, \sigma)\}$.

7 Autómatas finitos y expresiones regulares

A continuación veremos cómo, **dado cualquier AF M, se puede construir una expresión regular r, tal que $L(r) = L(M)$.**

Dado cualquier AF $M = (Q, \Sigma, q_0, \Delta, F)$, definimos:

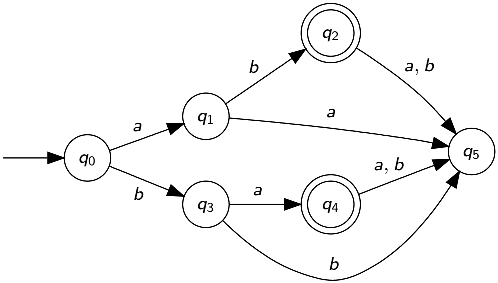
$$A_i = \{w \in \Sigma^* \mid \hat{\Delta}(q_i, w) \cap F \neq \emptyset\}$$

es decir, A_i contiene todas las cadenas que desde q_i nos llevan a algún estado final.

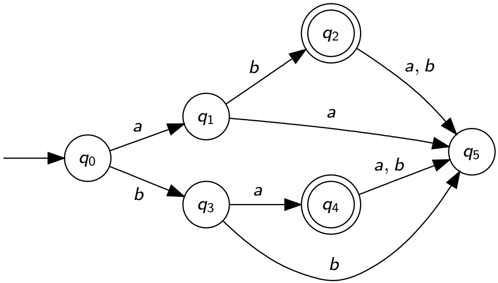
Por lo tanto, obsérvese que:

- $A_0 = L(M)$.
- Es posible que algún $A_i = \emptyset$.
- Si $q_i \in F$, entonces $\epsilon \in A_i$.
- En general, $A_i = \bigcup_j \{\sigma\, A_j \mid q_j \in \Delta(q_i, \sigma)\}$.
- Lo que haremos entonces será calcular A_0 como una expresión regular, a partir del resto de los conjuntos A_i.

Ejemplo:

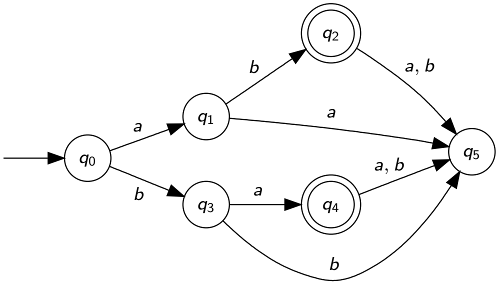


Ejemplo:



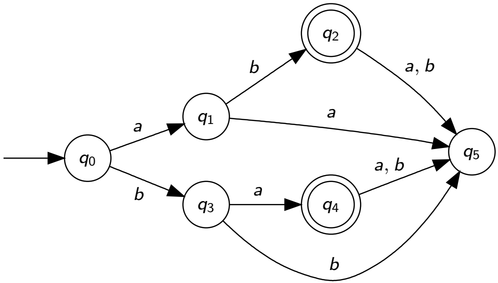
$$A_0 \;=\; a\,A_1 \cup b\,A_3$$

Ejemplo:



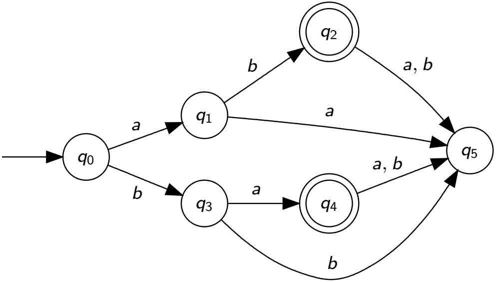
$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5
\end{aligned}
$$

Ejemplo:



$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5 \\
A_2 &= (a \cup b)\,A_5 \cup \epsilon
\end{aligned}
$$

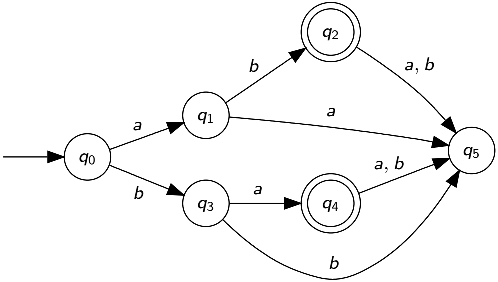
Ejemplo:



$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5 \\
A_2 &= (a \cup b)\,A_5 \cup \epsilon \\
A_3 &= a\,A_4 \cup b\,A_5
\end{aligned}
$$

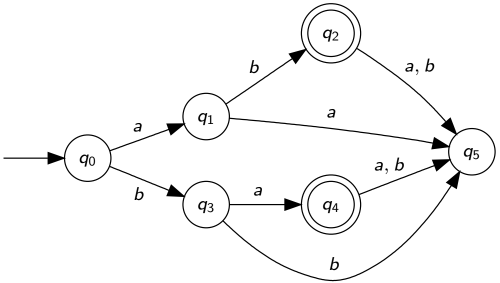
7 Autómatas finitos y expresiones regulares

Ejemplo:



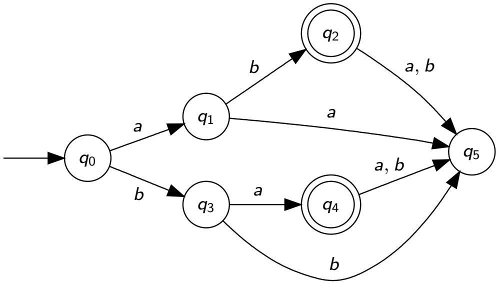
$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5 \\
A_2 &= (a \cup b)\,A_5 \cup \epsilon \\
A_3 &= a\,A_4 \cup b\,A_5 \\
A_4 &= (a \cup b)\,A_5 \cup \epsilon
\end{aligned}
$$

Ejemplo:



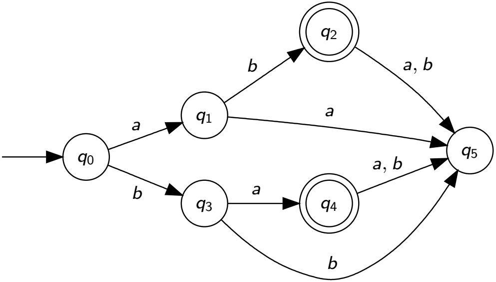
$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5 \\
A_2 &= (a \cup b)\,A_5 \cup \epsilon \\
A_3 &= a\,A_4 \cup b\,A_5 \\
A_4 &= (a \cup b)\,A_5 \cup \epsilon \\
A_5 &= \emptyset
\end{aligned}
$$

Ejemplo:



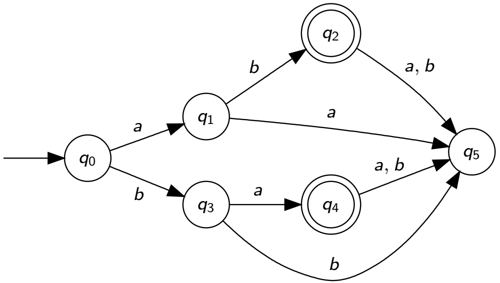
$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5 \\
A_2 &= (a \cup b)\,A_5 \cup \epsilon \\
A_3 &= a\,A_4 \cup b\,A_5 \\
A_4 &= (a \cup b)\,A_5 \cup \epsilon \quad = \quad \epsilon \\
A_5 &= \emptyset
\end{aligned}
$$

Ejemplo:



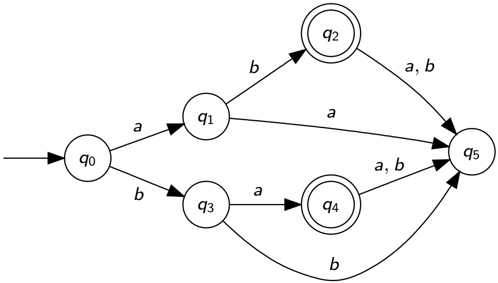
$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5 \\
A_2 &= (a \cup b)\,A_5 \cup \epsilon \\
A_3 &= a\,A_4 \cup b\,A_5 && = \quad a \\
A_4 &= (a \cup b)\,A_5 \cup \epsilon && = \quad \epsilon \\
A_5 &= \emptyset
\end{aligned}
$$

Ejemplo:



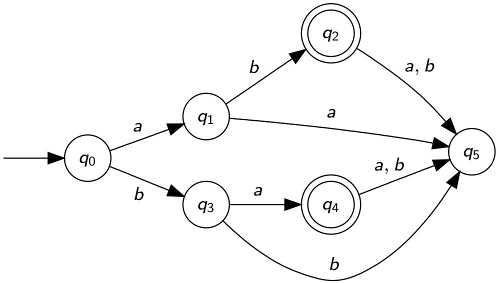
$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5 \\
A_2 &= (a \cup b)\,A_5 \cup \epsilon &&= \epsilon \\
A_3 &= a\,A_4 \cup b\,A_5 &&= a \\
A_4 &= (a \cup b)\,A_5 \cup \epsilon &&= \epsilon \\
A_5 &= \emptyset
\end{aligned}
$$

Ejemplo:



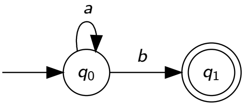
$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_3 \\
A_1 &= b\,A_2 \cup a\,A_5 &&= b \\
A_2 &= (a \cup b)\,A_5 \cup \epsilon &&= \epsilon \\
A_3 &= a\,A_4 \cup b\,A_5 &&= a \\
A_4 &= (a \cup b)\,A_5 \cup \epsilon &&= \epsilon \\
A_5 &= \emptyset
\end{aligned}
$$

Ejemplo:



$$
\begin{array}{rclcl}
A_0 & = & a\,A_1 \cup b\,A_3 & = & \boxed{ab \cup ba} \\
A_1 & = & b\,A_2 \cup a\,A_5 & = & b \\
A_2 & = & (a \cup b)\,A_5 \cup \epsilon & = & \epsilon \\
A_3 & = & a\,A_4 \cup b\,A_5 & = & a \\
A_4 & = & (a \cup b)\,A_5 \cup \epsilon & = & \epsilon \\
A_5 & = & \emptyset & &
\end{array}
$$

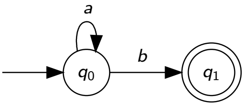
Otro ejemplo:



$$\left.\begin{array}{l} A_0 = a\,A_0 \cup b\,A_1 \\ A_1 = \epsilon \end{array}\right\} \Rightarrow\ A_0 = a\,A_0 \cup b$$

7 Autómatas finitos y expresiones regulares

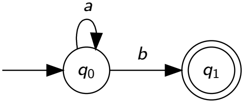
Otro ejemplo:



$$\left.\begin{array}{l} A_0 = a\,A_0 \cup b\,A_1 \\ A_1 = \epsilon \end{array}\right\} \Rightarrow \; A_0 = a\,A_0 \cup b$$

Realmente, esta expresión debería ser $a^* b$. Pero, en este caso, no podemos simplificar más, a no ser que introduzcamos el siguiente resultado.

Otro ejemplo:



$$\left.\begin{array}{l} A_0 = a\,A_0 \cup b\,A_1 \\ A_1 = \epsilon \end{array}\right\} \Rightarrow\ A_0 = a\,A_0 \cup b$$

Realmente, esta expresión debería ser $a^* b$. Pero, en este caso, no podemos simplificar más, a no ser que introduzcamos el siguiente resultado.

Lema de Arden

Una ecuación de la forma $X = A \cdot X \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^* \cdot B$. $\square$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

<u>Demostración</u>:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^* B$.

<u>Demostración</u>:

- Primero hay que ver que $A^* B$ es solución:
  $\boxed{A^* B} = (A^+ \cup \epsilon)B = A^+ B \cup B = (AA^*)B \cup B = A\boxed{(A^* B)} \cup B$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

<u>Demostración</u>:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

<u>Demostración</u>:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

<u>Demostración</u>:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$

- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$

- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$

- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC =$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$

- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC = \boxed{A^*B \cup AC}$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

<u>Demostración</u>:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$

- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC = \boxed{A^*B \cup AC}$

  Ahora intersecamos con C en ambos miembros, recordando que $A^*B \cap C = \emptyset$:

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

<u>Demostración</u>:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC = \boxed{A^*B \cup AC}$

  Ahora intersecamos con C en ambos miembros, recordando que $A^*B \cap C = \emptyset$:
  $(A^*B \cup C) \cap C = (A^*B \cup AC) \cap C \quad \Rightarrow$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

<u>Demostración</u>:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$

- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC = \boxed{A^*B \cup AC}$

  Ahora intersecamos con C en ambos miembros, recordando que $A^*B \cap C = \emptyset$:
  $(A^*B \cup C) \cap C = (A^*B \cup AC) \cap C \;\Rightarrow\; C = AC \cap C \;\Rightarrow$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC = \boxed{A^*B \cup AC}$

  Ahora intersecamos con C en ambos miembros, recordando que $A^*B \cap C = \emptyset$:
  $(A^*B \cup C) \cap C = (A^*B \cup AC) \cap C \;\Rightarrow\; C = AC \cap C \;\Rightarrow\; C \subseteq AC$

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$
- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC = \boxed{A^*B \cup AC}$

  Ahora intersecamos con C en ambos miembros, recordando que $A^*B \cap C = \emptyset$:
  $(A^*B \cup C) \cap C = (A^*B \cup AC) \cap C \;\Rightarrow\; C = AC \cap C \;\Rightarrow\; C \subseteq AC$

  Pero dado que $\epsilon \notin A$, AC es una operación que produce cadenas más largas que las de C.

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

<u>Demostración</u>:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$

- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC = \boxed{A^*B \cup AC}$

  Ahora intersecamos con C en ambos miembros, recordando que $A^*B \cap C = \emptyset$:
  $(A^*B \cup C) \cap C = (A^*B \cup AC) \cap C \;\Rightarrow\; C = AC \cap C \;\Rightarrow\; C \subseteq AC$

  Pero dado que $\epsilon \notin A$, AC es una operación que produce cadenas más largas que las de C.

  Por tanto, la única forma de que $C \subseteq AC$ es que $C = \emptyset$.

Lema de Arden (demostración)

Una ecuación de la forma $X = AX \cup B$, donde $\epsilon \notin A$, tiene una solución única $X = A^*B$.

Demostración:

- Primero hay que ver que A^*B es solución:
  $\boxed{A^*B} = (A^+ \cup \epsilon)B = A^+B \cup B = (AA^*)B \cup B = A\boxed{(A^*B)} \cup B$

- Ahora veamos que es la única solución:
  Para ello, supongamos que hay otra mayor $X = A^*B \cup C$ tal que $A^*B \cap C = \emptyset$.

  Si ahora en la ecuación $X = AX \cup B$ substituimos X por $A^*B \cup C$, obtenemos:
  $\boxed{A^*B \cup C} = A(A^*B \cup C) \cup B = AA^*B \cup AC \cup B = A^+B \cup AC \cup B = A^+B \cup B \cup AC =$
  $= (A^+ \cup \epsilon)B \cup AC = \boxed{A^*B \cup AC}$

  Ahora intersecamos con C en ambos miembros, recordando que $A^*B \cap C = \emptyset$:
  $(A^*B \cup C) \cap C = (A^*B \cup AC) \cap C \;\Rightarrow\; C = AC \cap C \;\Rightarrow\; C \subseteq AC$

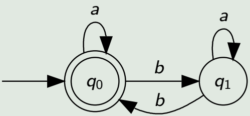
  Pero dado que $\epsilon \notin A$, AC es una operación que produce cadenas más largas que las de C.

  Por tanto, la única forma de que $C \subseteq AC$ es que $C = \emptyset$.

  Por tanto, $X = A^*B$ es la solución única. $\square$

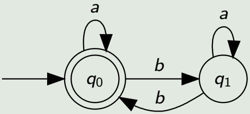
Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:

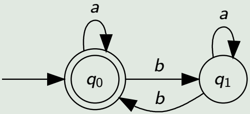


Solución:

$$A_0 \quad = \quad a\,A_0 \cup b\,A_1 \cup \epsilon$$

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:

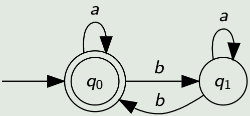


Solución:

$$
\begin{aligned}
A_0 &= a\,A_0 \cup b\,A_1 \cup \epsilon \\
A_1 &= a\,A_1 \cup b\,A_0
\end{aligned}
$$

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:

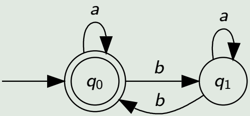


Solución:

$$
\begin{aligned}
A_0 &= a\,A_0 \cup b\,A_1 \cup \epsilon \\
A_1 &= a\,A_1 \cup b\,A_0 \\
&= a^*\,b\,A_0
\end{aligned}
$$

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



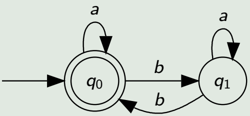
Solución:

$$
\begin{aligned}
A_0 &= a\,A_0 \cup b\,A_1 \cup \epsilon \\
A_1 &= a\,A_1 \cup b\,A_0 \\
&= a^*\,b\,A_0
\end{aligned}
$$

$$
A_0 = a\,A_0 \cup b\,a^*\,b\,A_0 \cup \epsilon
$$

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



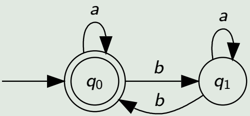
Solución:

$$
\begin{aligned}
A_0 &= a\,A_0 \cup b\,A_1 \cup \epsilon \\
A_1 &= a\,A_1 \cup b\,A_0 \\
&= a^*\,b\,A_0
\end{aligned}
$$

$$
\begin{aligned}
A_0 &= a\,A_0 \cup b\,a^*\,b\,A_0 \cup \epsilon \\
&= (a \cup b\,a^*\,b)\,A_0 \cup \epsilon
\end{aligned}
$$

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



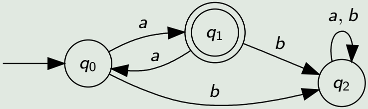
Solución:

$$
\begin{aligned}
A_0 &= a\,A_0 \cup b\,A_1 \cup \epsilon \\
A_1 &= a\,A_1 \cup b\,A_0 \\
&= a^*\,b\,A_0
\end{aligned}
$$

$$
\begin{aligned}
A_0 &= a\,A_0 \cup b\,a^*\,b\,A_0 \cup \epsilon \\
&= (a \cup b\,a^*\,b)\,A_0 \cup \epsilon \\
&= \boxed{(a \cup b\,a^*\,b)^*}
\end{aligned}
$$

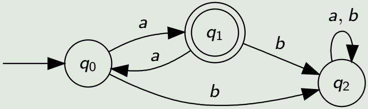
Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:

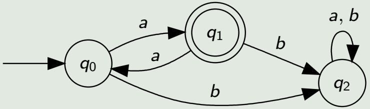


Solución:

$$A_0 \;=\; a\,A_1 \cup b\,A_2$$

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



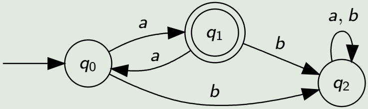
Solución:

$$A_0 = a\,A_1 \cup b\,A_2$$

$$A_1 = a\,A_0 \cup b\,A_2 \cup \epsilon$$

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



Solución:

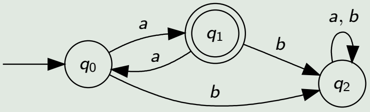
$$A_0 = a\,A_1 \cup b\,A_2$$

$$A_1 = a\,A_0 \cup b\,A_2 \cup \epsilon$$

$$A_2 = (a \cup b)\,A_2$$

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



Solución:

$$A_0 = a\,A_1 \cup b\,A_2$$

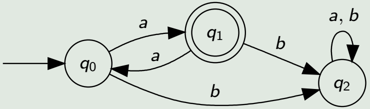
$$A_1 = a\,A_0 \cup b\,A_2 \cup \epsilon$$

$$A_2 = (a \cup b)\,A_2 \quad = \quad (a \cup b)^* \cdot \emptyset \quad = \quad \emptyset$$

Atención al Lema de Arden cuando en $X = AX \cup B$ no existe B.

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



<u>Solución</u>:

$$A_0 = a\,A_1 \cup b\,A_2$$

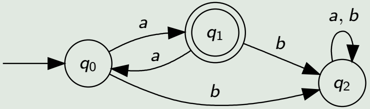
$$A_1 = a\,A_0 \cup b\,A_2 \cup \epsilon = a\,A_0 \cup b \cdot \emptyset \cup \epsilon = a\,A_0 \cup \epsilon$$

$$A_2 = (a \cup b)\,A_2 = (a \cup b)^* \cdot \emptyset = \emptyset$$

Atención al Lema de Arden cuando en $X = AX \cup B$ no existe B.

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



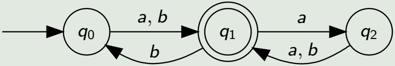
Solución:

$$
\begin{aligned}
A_0 &= a\,A_1 \cup b\,A_2 &&= a\,(a\,A_0 \cup \epsilon) \cup b \cdot \emptyset = a\,a\,A_0 \cup a = \boxed{(aa)^*\,a} \\
A_1 &= a\,A_0 \cup b\,A_2 \cup \epsilon &&= a\,A_0 \cup b \cdot \emptyset \cup \epsilon = a\,A_0 \cup \epsilon \\
A_2 &= (a \cup b)\,A_2 &&= (a \cup b)^* \cdot \emptyset = \emptyset
\end{aligned}
$$

Atención al Lema de Arden cuando en $X = AX \cup B$ no existe B.

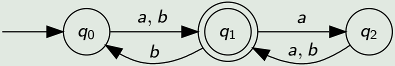
Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



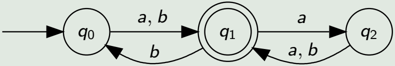
Solución:

$$
\begin{aligned}
A_0 &= (a \cup b)\, A_1 &&= (a \cup b)\, [a\,(a \cup b)]^*(b\, A_0 \cup \epsilon) \\
& &&= (a \cup b)\, [a\,(a \cup b)]^*\, b\, A_0 \cup (a \cup b)\, [a\,(a \cup b)]^* \\
& &&= \boxed{[(a \cup b)\, [a\,(a \cup b)]^*\, b]^*\, (a \cup b)\, [a\,(a \cup b)]^*} \\
A_1 &= b\, A_0 \cup a\, A_2 \cup \epsilon &&= b\, A_0 \cup a\,(a \cup b)\, A_1 \cup \epsilon \quad = \quad [a\,(a \cup b)]^*(b\, A_0 \cup \epsilon) \\
A_2 &= (a \cup b)\, A_1
\end{aligned}
$$

7 Autómatas finitos y expresiones regulares

Ejercicio

Obtenga la expresión regular que denota el lenguaje aceptado por el siguiente autómata:



Solución:

$$\begin{array}{lclcl}
A_0 & = & (a \cup b)\, A_1 & = & (a \cup b)\, [a\,(a \cup b)]^*(b\, A_0 \cup \epsilon) \\
 & & & = & (a \cup b)\, [a\,(a \cup b)]^*\, b\, A_0 \cup (a \cup b)\, [a\,(a \cup b)]^* \\
 & & & = & \boxed{[(a \cup b)\, [a\,(a \cup b)]^*\, b]^*\, (a \cup b)\, [a\,(a \cup b)]^*} \\
A_1 & = & b\, A_0 \cup a\, A_2 \cup \epsilon & = & b\, A_0 \cup a\,(a \cup b)\, A_1 \cup \epsilon \quad = \quad [a\,(a \cup b)]^*(b\, A_0 \cup \epsilon) \\
A_2 & = & (a \cup b)\, A_1 & &
\end{array}$$

Obsérvese que este método no siempre obtiene las expresiones regulares más compactas. En este caso, el lenguaje aceptado por el autómata podría denotarse también mediante la expresión $(a \cup b)\, [a\,(a \cup b) \cup b\,(a \cup b)]^*$, la cual se puede simplificar en $(a \cup b)\, [(a \cup b)\,(a \cup b)]^*$.

Resumen de lo vista hasta ahora:

- Son posibles todas estas transformaciones:

$$LR \rightarrow ER \rightarrow AFN\text{-}\epsilon \rightarrow AFN \rightarrow AFD \rightarrow AFD\ mínimo$$

7 Autómatas finitos y expresiones regulares

Resumen de lo vista hasta ahora:

- Son posibles todas estas transformaciones:

  $LR \rightarrow ER \rightarrow AFN\text{-}\epsilon \rightarrow AFN \rightarrow AFD \rightarrow AFD\ mínimo$

- Adicionalmente, también es posible:

  $AF \rightarrow ER$

7 Autómatas finitos y expresiones regulares

Resumen de lo vista hasta ahora:

- Son posibles todas estas transformaciones:

  $LR \rightarrow ER \rightarrow AFN\text{-}\epsilon \rightarrow AFN \rightarrow AFD \rightarrow AFD\ mínimo$

- Adicionalmente, también es posible:

  $AF \rightarrow ER$

- Todos estos formalismos son equivalentes:

  Dado un LR cualquiera, se puede construir una ER que lo denota o un AF que lo acepta.

  Y viceversa: las ER,s sólo denotan lenguajes regulares y los AF,s sólo aceptan lenguajes regulares.

7 Autómatas finitos y expresiones regulares

Resumen de lo vista hasta ahora:

- Son posibles todas estas transformaciones:

  LR $\rightarrow$ *ER* $\rightarrow$ *AFN-*ϵ $\rightarrow$ *AFN* $\rightarrow$ *AFD* $\rightarrow$ *AFD mínimo*

- Adicionalmente, también es posible:

  AF $\rightarrow$ *ER*

- Todos estos formalismos son equivalentes:

  Dado un LR cualquiera, se puede construir una ER que lo denota o un AF que lo acepta.

  Y viceversa: las ER,s sólo denotan lenguajes regulares y los AF,s sólo aceptan lenguajes regulares.

Estamos entonces en condiciones de enunciar el siguiente resultado:

Teorema de Kleene

Un lenguaje es regular si y sólo si es aceptado por un autómata finito. □

Contenidos

8 Propiedades de los lenguajes regulares

Por definición, los lenguajes regulares son cerrados para las operaciones de unión, concatenación y cierre de Kleene.

8 Propiedades de los lenguajes regulares

Por definición, los lenguajes regulares son cerrados para las operaciones de unión, concatenación y cierre de Kleene.

Ejercicio

¿Los lenguajes regulares son cerrados también para la operación de complementario?

8 Propiedades de los lenguajes regulares

Por definición, los lenguajes regulares son cerrados para las operaciones de unión, concatenación y cierre de Kleene.

Ejercicio

¿Los lenguajes regulares son cerrados también para la operación de complementario?

Solución:

La respuesta es afirmativa.

Dado un lenguaje regular L, siempre existe un AFD M tal que $L(M) = L$.

Si $M = (Q, \Sigma, s, \delta, F)$, siempre podemos construir $\overline{M} = (Q, \Sigma, s, \delta, \mathbf{Q} - \mathbf{F})$:

- Es decir, donde M aceptaba, ahora $\overline{M}$ rechaza.
- Y viceversa, donde M rechazaba, ahora $\overline{M}$ acepta.

Por tanto, $L(\overline{M}) = \Sigma^* - L(M) = \overline{L(M)} = \overline{L}$.

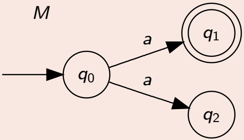
Y si $\overline{L}$ es aceptado por un AFD, entonces $\overline{L}$ es regular.

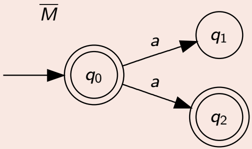
Así pues, los lenguajes regulares son cerrados para la operación de complementario.

Atención

El razonamiento del ejercicio anterior no sería válido si M no fuera un AFD.

Ejemplo:





Dado M podemos construir $\overline{M}$, pero $\overline{M}$ sigue aceptando la cadena a.

Es decir, en este caso, $L(\overline{M}) \neq \overline{L(M)}$.

Atención

El razonamiento del ejercicio anterior no sería válido si M no fuera un AFD.

Ejemplo:

Dado M podemos construir $\overline{M}$, pero $\overline{M}$ sigue aceptando la cadena a.

Es decir, en este caso, $L(\overline{M}) \neq \overline{L(M)}$.

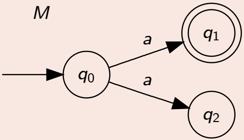
Ejercicio

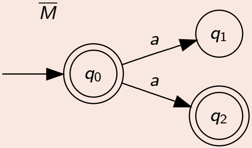
Demuestre que si L_1 y L_2 son lenguajes regulares, entonces $L_1 \cap L_2$ también lo es.

Atención

El razonamiento del ejercicio anterior no sería válido si M no fuera un AFD.

Ejemplo:





Dado M podemos construir $\overline{M}$, pero $\overline{M}$ sigue aceptando la cadena a.

Es decir, en este caso, $L(\overline{M}) \neq \overline{L(M)}$.

Ejercicio

Demuestre que si L_1 y L_2 son lenguajes regulares, entonces $L_1 \cap L_2$ también lo es.

Solución:

$L_1 \cap L_2$ puede escribirse como $\overline{(\overline{L_1} \cup \overline{L_2})}$.

Por tanto, los lenguajes regulares también son cerrados para la operación de intersección.

8 Propiedades de los lenguajes regulares

No obstante, hay que recordar que existen otras clases de lenguajes que no son regulares. Por tanto, resultaría útil tener un método para saber si un lenguaje es regular o no.

8 Propiedades de los lenguajes regulares

No obstante, hay que recordar que existen otras clases de lenguajes que no son regulares. Por tanto, resultaría útil tener un método para saber si un lenguaje es regular o no.

¿Cuándo un lenguaje L es regular?

- Si L es finito o ya está especificado por una ER o por un AF, entonces es regular.
- Pero si no es el caso, la búsqueda exhaustiva de una ER o de un AF para L puede resultar a veces totalmente infructuosa.

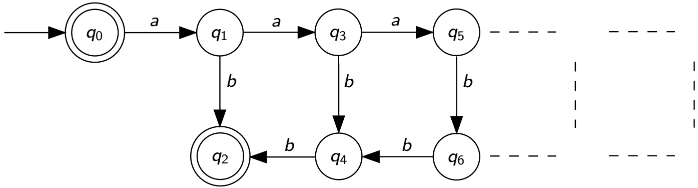
8 Propiedades de los lenguajes regulares

No obstante, hay que recordar que existen otras clases de lenguajes que no son regulares. Por tanto, resultaría útil tener un método para saber si un lenguaje es regular o no.

¿Cuándo un lenguaje L es regular?

- Si L es finito o ya está especificado por una ER o por un AF, entonces es regular.
- Pero si no es el caso, la búsqueda exhaustiva de una ER o de un AF para L puede resultar a veces totalmente infructuosa.

Por ejemplo, si intentáramos construir un AF que acepte el lenguaje no regular $L = \{a^n b^n \mid n \geq 0\} = \{\epsilon, ab, aabb, aaabbb, \ldots\}$, podríamos hacerlo hasta un cierto n finito, pero no para todo $n \geq 0$:



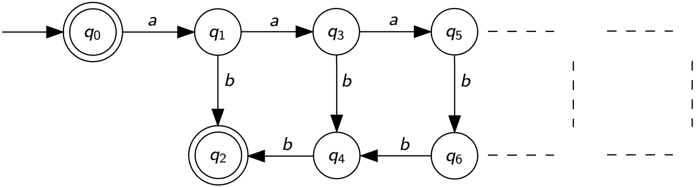
8 Propiedades de los lenguajes regulares

No obstante, hay que recordar que existen otras clases de lenguajes que no son regulares. Por tanto, resultaría útil tener un método para saber si un lenguaje es regular o no.

¿Cuándo un lenguaje L es regular?

- Si L es finito o ya está especificado por una ER o por un AF, entonces es regular.
- Pero si no es el caso, la búsqueda exhaustiva de una ER o de un AF para L puede resultar a veces totalmente infructuosa.

Por ejemplo, si intentáramos construir un AF que acepte el lenguaje no regular $L = \{a^n b^n \mid n \geq 0\} = \{\epsilon, ab, aabb, aaabbb, \dots\}$, podríamos hacerlo hasta un cierto n finito, pero no para todo $n \geq 0$:



Así pues, necesitamos estudiar algunas **nuevas propiedades** que están presentes sólo en los **lenguajes regulares infinitos**.

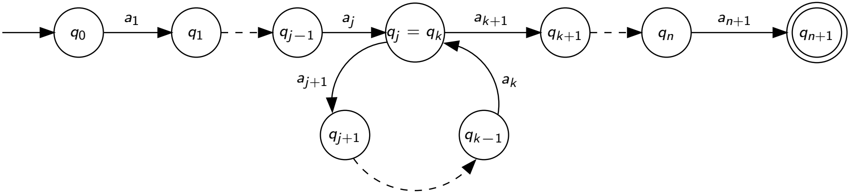
8 Propiedades de los lenguajes regulares

Supongamos un lenguaje regular aceptado por un AFD $M = (Q, \Sigma, q_0, \delta, F)$, donde Q contiene n estados. Si $L(M)$ es infinito, existirán cadenas aceptadas por M con longitud mayor que n. Supongamos $w = a_1\, a_2\, \ldots\, a_{n+1}$ una cadena de $L(M)$ de longitud $n + 1$.

8 Propiedades de los lenguajes regulares

Supongamos un lenguaje regular aceptado por un AFD $M = (Q, \Sigma, q_0, \delta, F)$, donde Q contiene n estados. Si $L(M)$ es infinito, existirán cadenas aceptadas por M con longitud mayor que n. Supongamos $w = a_1\, a_2\, \ldots\, a_{n+1}$ una cadena de $L(M)$ de longitud $n+1$.

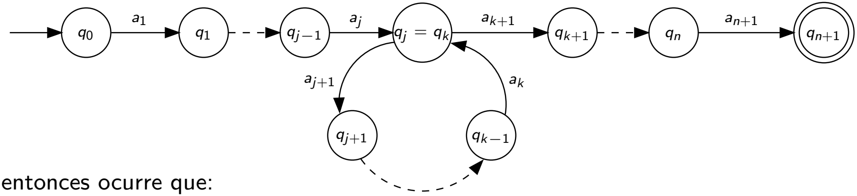
Si $\delta(q_0, a_1) = q_1$, $\delta(q_1, a_2) = q_2$, $\ldots$, $\delta(q_n, a_{n+1}) = q_{n+1}$, entonces $q_0\, q_1\, q_2\, \ldots\, q_{n+1}$ constituye el camino de aceptación de w en M. Pero dado que Q sólo tiene n estados, no todos los q_i de ese camino serán distintos. Es decir, para algunos índices j y k, $0 \leq j < k \leq n+1$, tendremos que $q_j = q_k$ y por tanto tendremos un ciclo en el camino de aceptación de w. Y puesto que $j < k$, la subcadena $a_{j+1}\, \ldots\, a_k$ procesada en el ciclo tendrá una longitud de al menos un símbolo.



8 Propiedades de los lenguajes regulares

Supongamos un lenguaje regular aceptado por un AFD $M = (Q, \Sigma, q_0, \delta, F)$, donde Q contiene n estados. Si $L(M)$ es infinito, existirán cadenas aceptadas por M con longitud mayor que n. Supongamos $w = a_1\, a_2\, \ldots\, a_{n+1}$ una cadena de $L(M)$ de longitud $n + 1$.

Si $\delta(q_0, a_1) = q_1$, $\delta(q_1, a_2) = q_2$, $\ldots$, $\delta(q_n, a_{n+1}) = q_{n+1}$, entonces $q_0\, q_1\, q_2\, \ldots\, q_{n+1}$ constituye el camino de aceptación de w en M. Pero dado que Q sólo tiene n estados, no todos los q_i de ese camino serán distintos. Es decir, para algunos índices j y k, $0 \leq j < k \leq n + 1$, tendremos que $q_j = q_k$ y por tanto tendremos un ciclo en el camino de aceptación de w. Y puesto que $j < k$, la subcadena $a_{j+1}\, \ldots\, a_k$ procesada en el ciclo tendrá una longitud de al menos un símbolo.



Y entonces ocurre que:

- La cadena $a_1\, \ldots\, a_j\, a_{k+1}\, \ldots\, a_{n+1}$ (que corresponde a la elección de ninguna iteración del ciclo) pertenecerá también a $L(M)$.
- Y todas las cadenas de la forma $a_1\, \ldots\, a_j\, (a_{j+1}\, \ldots\, a_k)^m\, a_{k+1}\, \ldots\, a_{n+1}$ (que corresponden a m iteraciones del ciclo) estarán también en $L(M)$, $\forall m \geq 0$.

8 Propiedades de los lenguajes regulares

Es decir, podemos **“bombear”** cero o más veces la subcadena procesada en el ciclo y seguiremos obteniendo cadenas aceptadas por el autómata. Esta idea se formaliza en el siguiente resultado.

Lema del Bombeo

Sea L un lenguaje regular infinito. Entonces existe una constante k asociada a L, tal que, si w es una cadena de L cuya longitud es mayor o igual que k, es decir,

$$w \in L, \; |w| \geq k,$$

w se puede descomponer de la forma

$$w = uvx, \quad \text{donde} \quad |v| \geq 1, \; |uv| \leq k,$$

y todas las cadenas de la forma $uv^i x$ pertenecerán a L, $\forall i \geq 0$. $\square$

8 Propiedades de los lenguajes regulares

Es decir, podemos **"bombear"** cero o más veces la subcadena procesada en el ciclo y seguiremos obteniendo cadenas aceptadas por el autómata. Esta idea se formaliza en el siguiente resultado.

Lema del Bombeo

Sea L un lenguaje regular infinito. Entonces existe una constante k asociada a L, tal que, si w es una cadena de L cuya longitud es mayor o igual que k, es decir,

$$w \in L, \ |w| \geq k,$$

w se puede descomponer de la forma

$$w = uvx, \quad \text{donde} \quad |v| \geq 1, \ |uv| \leq k,$$

y todas las cadenas de la forma uv^ix pertenecerán a L, $\forall i \geq 0$. □

El Lema del Bombeo representa una manera de demostrar que un determinado lenguaje **no es regular**.

La manera de utilizarlo es elegir una cadena del lenguaje, y una descomposición adecuada de la misma, de tal forma que al bombear la parte central obtengamos cadenas que no pertenezcan al lenguaje considerado.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^n b^n \mid n \geq 0\}$ no es regular.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^n b^n \mid n \geq 0\}$ no es regular.

Solución:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^n b^n \mid n \geq 0\}$ no es regular.

Solución:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^k b^k$.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^n b^n \mid n \geq 0\}$ no es regular.

Solución:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.
Elegimos la cadena $w = a^k b^k$.
Dado que $w \in L$ y $|w| = 2k \geq k$, w está en las condiciones del lema.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^n b^n \mid n \geq 0\}$ no es regular.

<u>Solución</u>:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^k b^k$.

Dado que $w \in L$ y $|w| = 2k \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo $uv^i x$ debería estar en L, $\forall i \geq 0$.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^n b^n \mid n \geq 0\}$ no es regular.

<u>Solución</u>:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^k b^k$.

Dado que $w \in L$ y $|w| = 2k \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo $uv^i x$ debería estar en L, $\forall i \geq 0$.

Si $|uv| \leq k$, entonces las subcadenas u y v están formadas sólo por $a$es, es decir:

$$u = a^r \qquad v = a^s \; (s \geq 1) \qquad x = a^{k-(r+s)} b^k$$

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^n b^n \mid n \geq 0\}$ no es regular.

<u>Solución</u>:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^k b^k$.

Dado que $w \in L$ y $|w| = 2k \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo $uv^i x$ debería estar en L, $\forall i \geq 0$.

Si $|uv| \leq k$, entonces las subcadenas u y v están formadas sólo por $a$es, es decir:

$$u = a^r \qquad v = a^s \ (s \geq 1) \qquad x = a^{k-(r+s)} b^k$$

Y si elegimos, por ejemplo, el bombeo $i = 2$, obtenemos lo siguiente:

$$uv^2 x = a^r a^{2s} a^{k-(r+s)} b^k = a^{k+s} b^k$$

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^n b^n \mid n \geq 0\}$ no es regular.

Solución:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^k b^k$.

Dado que $w \in L$ y $|w| = 2k \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo $uv^i x$ debería estar en L, $\forall i \geq 0$.

Si $|uv| \leq k$, entonces las subcadenas u y v están formadas sólo por $a$es, es decir:

$$u = a^r \qquad v = a^s \; (s \geq 1) \qquad x = a^{k-(r+s)} b^k$$

Y si elegimos, por ejemplo, el bombeo $i = 2$, obtenemos lo siguiente:

$$uv^2 x = a^r a^{2s} a^{k-(r+s)} b^k = a^{k+s} b^k$$

Pero, dado que $s \geq 1$, la cadena $a^{k+s} b^k$ no tiene el mismo número de $a$es que de $b$es.
Es decir, el bombeo $i = 2$ produce una cadena que no está en L.
Por lo tanto, L no es regular.

El hecho de que $\{a^n b^n \mid n \geq 0\}$ no sea regular es un reflejo de las propiedades de este tipo de lenguajes y de los autómatas finitos:

- La cantidad de memoria disponible para aceptar o rechazar una cadena es limitada y se reduce, en cada paso, al estado y símbolo actuales.
- En este caso, cuando analizamos las *b*es, cualquier autómata finito es incapaz de mantener información sobre cuántas *a*es han sido leídas previamente.

8 Propiedades de los lenguajes regulares

El hecho de que $\{a^n b^n \mid n \geq 0\}$ no sea regular es un reflejo de las propiedades de este tipo de lenguajes y de los autómatas finitos:

- La cantidad de memoria disponible para aceptar o rechazar una cadena es limitada y se reduce, en cada paso, al estado y símbolo actuales.
- En este caso, cuando analizamos las $b$es, cualquier autómata finito es incapaz de mantener información sobre cuántas $a$es han sido leídas previamente.

Por otra parte, es interesante comentar también lo siguiente:

- El ejercicio anterior aplica las propiedades del lema del bombeo utilizando la **forma estructural de las cadenas**: un bombeo apropiado aumenta el número de $a$es y no el de $b$es, produciendo cadenas fuera del lenguaje en cuestión.
- Sin embargo, esto no es posible con lenguajes como $\{a^{i^2} \mid i \geq 1\}$, donde todas las cadenas están formadas por el mismo símbolo a.

  No obstante, este lenguaje tampoco es regular y este hecho se puede demostrar también mediante el lema del bombeo. Para ello, lo que debemos hacer es estudiar con detalle la única información disponible en este caso: la **longitud de las cadenas** generadas por los diferentes bombeos considerados.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{i^2} \mid i \geq 1\}$ no es regular.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{i^2} \mid i \geq 1\}$ no es regular.

<u>Solución</u>:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{i^2} \mid i \geq 1\}$ no es regular.

Solución:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^{k^2}$.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{i^2} \mid i \geq 1\}$ no es regular.

Solución:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^{k^2}$.

Dado que $w \in L$ y $|w| = k^2 \geq k$, w está en las condiciones del lema.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{i^2} \mid i \geq 1\}$ no es regular.

Solución:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^{k^2}$.

Dado que $w \in L$ y $|w| = k^2 \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo uv^ix debería estar en L, $\forall i \geq 0$.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{i^2} \mid i \geq 1\}$ no es regular.

<u>Solución</u>:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^{k^2}$.

Dado que $w \in L$ y $|w| = k^2 \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo uv^ix debería estar en L, $\forall i \geq 0$.

Si $|uv| \leq k$, entonces la longitud de la cadena producida por el bombeo $i = 2$ verifica lo siguiente:

$$k^2 = |uvx| < |uv^2x| \leq k^2 + k < (k+1)^2$$

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{i^2} \mid i \geq 1\}$ no es regular.

<u>Solución</u>:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^{k^2}$.

Dado que $w \in L$ y $|w| = k^2 \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo uv^ix debería estar en L, $\forall i \geq 0$.

Si $|uv| \leq k$, entonces la longitud de la cadena producida por el bombeo $i = 2$ verifica lo siguiente:

$$k^2 = |uvx| < |uv^2x| \leq k^2 + k < (k+1)^2$$

Es decir, el bombeo $i = 2$ produce una cadena cuya longitud está entre dos cuadrados perfectos consecutivos.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{i^2} \mid i \geq 1\}$ no es regular.

Solución:

Suponemos que L es regular y sea k su constante asociada según el lema del bombeo.

Elegimos la cadena $w = a^{k^2}$.

Dado que $w \in L$ y $|w| = k^2 \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo uv^ix debería estar en L, $\forall i \geq 0$.

Si $|uv| \leq k$, entonces la longitud de la cadena producida por el bombeo $i = 2$ verifica lo siguiente:

$$k^2 = |uvx| < |uv^2x| \leq k^2 + k < (k+1)^2$$

Es decir, el bombeo $i = 2$ produce una cadena cuya longitud está entre dos cuadrados perfectos consecutivos.

Por lo tanto, la longitud de uv^2x no es un cuadrado perfecto.
Por lo tanto, $uv^2x \notin L$. Por lo tanto, L no es regular.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{w \mid w = w^I \text{ y } w \in \{a, b\}^*\}$ no es regular.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{w \mid w = w^I \text{ y } w \in \{a, b\}^*\}$ no es regular.

Solución:

Suponemos que lo es y tomamos $w = a^k b^k a^k$, donde k es la constante asociada a L. Dado que $w \in L$ y $|w| = 3k \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo $uv^i x$ debería estar en L, $\forall i \geq 0$.

Si $|uv| \leq k$, entonces las subcadenas u y v están formadas sólo por $a$es, es decir:

$$u = a^r \qquad v = a^s \ (s \geq 1) \qquad x = a^{k-(r+s)} b^k a^k$$

Así pues, cualquier bombeo $i \geq 2$ aumentará las primeras $a$es y no las segundas, produciendo cadenas que no estarán en L. Por lo tanto, L no es regular.

Obsérvese que, en este caso, incluso el bombeo $i = 0$ produce una cadena fuera del lenguaje:

$$uv^0 x = a^r a^{k-(r+s)} b^k a^k = a^{k-s} b^k a^k \notin L$$

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{2^n} \mid n \geq 0\}$ no es regular.

Ejercicio

Demuestre mediante el lema del bombeo que $L = \{a^{2^n} \mid n \geq 0\}$ no es regular.

Solución:

Suponemos que lo es y tomamos $w = a^{2^k}$, donde k es la constante asociada a L. Dado que $w \in L$ y $|w| = 2^k \geq k$, w está en las condiciones del lema.

Por tanto, w debería poder descomponerse de la forma $w = uvx$, donde $|v| \geq 1$ y $|uv| \leq k$, y cualquier bombeo uv^ix debería estar en L, $\forall i \geq 0$.

Pero veremos que no para cualquier descomposición se verifica esto. Para ello, obtenemos primeramente una cota superior de la longitud de la cadena producida por cualquier bombeo i:

$$|uv^ix| = |w| + (i-1)|v| = 2^k + (i-1)|v| \leq 2^k + (i-1)k$$

Y, en concreto, para el bombeo $i = 2$, tendremos lo siguiente:

$$2^k < |uv^2x| \leq 2^k + k < 2^{k+1}$$

Es decir, el bombeo $i = 2$ produce una cadena cuya longitud está entre dos potencias de 2 consecutivas. Por lo tanto, la longitud de uv^2x no es una potencia de 2. Por lo tanto, $uv^2x \notin L$. Por lo tanto, L no es regular.

8 Propiedades de los lenguajes regulares

Además de servir para demostrar que un lenguaje no es regular, las propiedades del lema del bombeo pueden ser utilizadas también para determinar si un AF acepta un lenguaje no vacío, o bien si dicho lenguaje es finito o infinito.

Teorema

Sea M un autómata finito con k estados:

1. $L(M) \neq \emptyset \Leftrightarrow M$ acepta una cadena de longitud menor que k.
2. $L(M)$ es infinito $\Leftrightarrow M$ acepta una cadena de longitud n, donde $k \leq n < 2k$. $\square$

8 Propiedades de los lenguajes regulares

Además de servir para demostrar que un lenguaje no es regular, las propiedades del lema del bombeo pueden ser utilizadas también para determinar si un AF acepta un lenguaje no vacío, o bien si dicho lenguaje es finito o infinito.

Teorema

Sea M un autómata finito con k estados:

1. $L(M) \neq \emptyset \Leftrightarrow M$ acepta una cadena de longitud menor que k.
2. $L(M)$ es infinito $\Leftrightarrow M$ acepta una cadena de longitud n, donde $k \leq n < 2k$. □

Dado que los alfabetos son conjuntos finitos de símbolos, los procedimientos propuestos por este teorema siempre terminan.

Sin embargo, esto no quiere decir que sean los más óptimos:

1. Para ver que $L(M)$ no es vacío, bastaría con comprobar que existe algún estado final accesible desde el estado inicial de M.
2. De igual manera, eliminando de M los estados no accesibles y los estados sumidero, la presencia de cualquier ciclo implicaría que $L(M)$ es infinito.

Contenidos

9 Aplicaciones prácticas de las ER,s y de los AF,s

Los **autómatas finitos** constituyen un modelo muy útil para muchos tipos de herramientas *software* que realizan tareas informáticas importantes, como por ejemplo:

9 Aplicaciones prácticas de las ER,s y de los AF,s

Los **autómatas finitos** constituyen un modelo muy útil para muchos tipos de herramientas *software* que realizan tareas informáticas importantes, como por ejemplo:

- *Software* para **diseño y verificación de circuitos digitales**.

9 Aplicaciones prácticas de las ER,s y de los AF,s

Los **autómatas finitos** constituyen un modelo muy útil para muchos tipos de herramientas *software* que realizan tareas informáticas importantes, como por ejemplo:

- *Software* para **diseño y verificación de circuitos digitales**.
- *Software* para comprobar la corrección de cualquier tipo de sistemas que tengan un número finito de estados diferentes, como los **protocolos de comunicación** o los **protocolos para el intercambio seguro de información**.

9 Aplicaciones prácticas de las ER,s y de los AF,s

Los **autómatas finitos** constituyen un modelo muy útil para muchos tipos de herramientas *software* que realizan tareas informáticas importantes, como por ejemplo:

- *Software* para **diseño y verificación de circuitos digitales**.
- *Software* para comprobar la corrección de cualquier tipo de sistemas que tengan un número finito de estados diferentes, como los **protocolos de comunicación** o los **protocolos para el intercambio seguro de información**.
- Los **analizadores léxicos de los compiladores**, esto es, la componente del compilador que descompone el texto de entrada en unidades lógicas con mayor nivel de significado tales como identificadores, cifras, palabras reservadas o signos de puntuación.

9 Aplicaciones prácticas de las ER,s y de los AF,s

Los **autómatas finitos** constituyen un modelo muy útil para muchos tipos de herramientas *software* que realizan tareas informáticas importantes, como por ejemplo:

- *Software* para **diseño y verificación de circuitos digitales**.
- *Software* para comprobar la corrección de cualquier tipo de sistemas que tengan un número finito de estados diferentes, como los **protocolos de comunicación** o los **protocolos para el intercambio seguro de información**.
- Los **analizadores léxicos de los compiladores**, esto es, la componente del compilador que descompone el texto de entrada en unidades lógicas con mayor nivel de significado tales como identificadores, cifras, palabras reservadas o signos de puntuación.
- *Software* para la **exploración de grandes volúmenes de texto**, como por ejemplo los integrados por conjuntos de páginas *web*, permitiendo así descubrir las apariciones de ciertas palabras, frases o cualquier otro tipo de patrones.

9 Aplicaciones prácticas de las ER,s y de los AF,s

Los **autómatas finitos** constituyen un modelo muy útil para muchos tipos de herramientas *software* que realizan tareas informáticas importantes, como por ejemplo:

- *Software* para **diseño y verificación de circuitos digitales**.
- *Software* para comprobar la corrección de cualquier tipo de sistemas que tengan un número finito de estados diferentes, como los **protocolos de comunicación** o los **protocolos para el intercambio seguro de información**.
- Los **analizadores léxicos de los compiladores**, esto es, la componente del compilador que descompone el texto de entrada en unidades lógicas con mayor nivel de significado tales como identificadores, cifras, palabras reservadas o signos de puntuación.
- *Software* para la **exploración de grandes volúmenes de texto**, como por ejemplo los integrados por conjuntos de páginas *web*, permitiendo así descubrir las apariciones de ciertas palabras, frases o cualquier otro tipo de patrones.
- Etc ...

9 Aplicaciones prácticas de las ER,s y de los AF,s

Las **expresiones regulares** juegan también un importante papel ya que, en general, ofrecen algo que los autómatas no proporcionan: una forma declarativa de expresar las cadenas que queremos aceptar.

9 Aplicaciones prácticas de las ER,s y de los AF,s

Las **expresiones regulares** juegan también un importante papel ya que, en general, ofrecen algo que los autómatas no proporcionan: una forma declarativa de expresar las cadenas que queremos aceptar.

Por tanto, las expresiones regulares se utilizan como lenguaje de entrada en muchos sistemas de proceso de cadenas.

9 Aplicaciones prácticas de las ER,s y de los AF,s

Las **expresiones regulares** juegan también un importante papel ya que, en general, ofrecen algo que los autómatas no proporcionan: una forma declarativa de expresar las cadenas que queremos aceptar.

Por tanto, las expresiones regulares se utilizan como lenguaje de entrada en muchos sistemas de proceso de cadenas.

En particular, en el caso de los analizadores léxicos, nos permiten especificar más cómodamente el formato de los patrones que deseamos identificar, los cuales constituyen el dato de entrada de los métodos estudiados en este capítulo para la obtención del autómata finito correspondiente a una expresión regular.

Las **expresiones regulares** juegan también un importante papel ya que, en general, ofrecen algo que los autómatas no proporcionan: una forma declarativa de expresar las cadenas que queremos aceptar.

Por tanto, las expresiones regulares se utilizan como lenguaje de entrada en muchos sistemas de proceso de cadenas.

En particular, en el caso de los analizadores léxicos, nos permiten especificar más cómodamente el formato de los patrones que deseamos identificar, los cuales constituyen el dato de entrada de los métodos estudiados en este capítulo para la obtención del autómata finito correspondiente a una expresión regular.

Dichos métodos son capaces de generar en cada momento el formalismo operativo más óptimo para realizar el proceso de reconocimiento adecuado a nuestras necesidades.

Fin del capítulo

"Lenguajes Regulares y Autómatas Finitos"