

Introducción a la Lógica y la Computación

Mariana Badano Héctor Gramaglia Pedro Sánchez Terraf
M. Clara Gorín Matías Steinberg

FaMAF, 11 de noviembre de 2020



1 Repaso

- Expresiones regulares
- Operaciones con lenguajes
- *Regex* definen lenguajes regulares

2 Teorema de Kleene

- Algoritmo

Operaciones con lenguajes

vacío	\emptyset	\emptyset
singulete	$\{x\} \quad (x \in \Sigma)$	x
complemento	$L^c = \Sigma^* \setminus L$	
intersección	$L \cap L' = \{\alpha \mid \alpha \in L \text{ y } \alpha \in L'\}$	
unión	$L \cup L' = \{\alpha \mid \alpha \in L \text{ ó } \alpha \in L'\}$	$r + r'$
concatenación	$LL' = \{\alpha\beta \mid \alpha \in L \text{ y } \beta \in L'\}$	rr'
potencias	$L^n = \begin{cases} \{\epsilon\} & \text{si } n = 0 \\ LL^k & \text{si } n = k + 1 \end{cases}$	$r^0 = \epsilon$ r^n
clausura	$L^* = \bigcup_{n \in \mathbb{N}} L^n$	r^*



UNC
Universidad
Nacional
de Córdoba



Expresiones regulares

Fijado un alfabeto Σ , son el menor conjunto *Regex* que cumple:

- vacío $\emptyset \in \text{Regex}$.
- épsilon $\epsilon \in \text{Regex}$.
- símbolo $x \in \Sigma \implies x \in \text{Regex}$.
- unión $r_1, r_2 \in \text{Regex} \implies r_1 + r_2 \in \text{Regex}$.
- concatenación $r_1, r_2 \in \text{Regex} \implies r_1 r_2 \in \text{Regex}$.
- clausura $r \in \text{Regex} \implies r^* \in \text{Regex}$.



UNC
Universidad
Nacional
de Córdoba



Expresiones regulares

Fijado un alfabeto Σ , son el menor conjunto *Regex* que cumple:

- vacío $\emptyset \in \text{Regex}$.
- épsilon $\epsilon \in \text{Regex}$.
- símbolo $x \in \Sigma \implies x \in \text{Regex}$.
- unión $r_1, r_2 \in \text{Regex} \implies r_1 + r_2 \in \text{Regex}$.
- concatenación $r_1, r_2 \in \text{Regex} \implies r_1 r_2 \in \text{Regex}$.
- clausura $r \in \text{Regex} \implies r^* \in \text{Regex}$.

$$L(\emptyset) := \emptyset$$

$$L(\epsilon) := \{\epsilon\}$$

$$L(x) := \{x\}$$

$$L(r_1 + r_2) := L(r_1) \cup L(r_2)$$

$$L(r_1 r_2) := L(r_1) L(r_2)$$

$$L(r^*) := (L(r))^*$$



UNC
Universidad
Nacional
de Córdoba



Teorema

Para toda expresión regular r , existe un ϵ -NFA $\mathbb{A}(r)$ con un único estado final tal que $L(r) = L(\mathbb{A}(r))$.

Teorema

Para toda expresión regular r , existe un ϵ -NFA $\mathbb{A}(r)$ con un único estado final tal que $L(r) = L(\mathbb{A}(r))$.

Ahora, vemos la recíproca.

Teorema (Kleene)

Todo lenguaje regular es de la forma $L(r)$ para alguna $r \in \text{Regex}$.

Teorema

Para toda expresión regular r , existe un ϵ -NFA $\mathbb{A}(r)$ con un único estado final tal que $L(r) = L(\mathbb{A}(r))$.

Ahora, vemos la recíproca.

Teorema (Kleene)

Todo lenguaje regular es de la forma $L(r)$ para alguna $r \in \text{Regex}$.

Daremos un método **recursivo** para obtener a partir de un ϵ -NFA $\mathbb{A} = (\{q_0, \dots, q_r\}, \Sigma, \delta, q_0, \{q_f\})$ con un único estado final, una expresión regular que denota el lenguaje $L(\mathbb{A})$.

Teorema

Para toda expresión regular r , existe un ϵ -NFA $\mathbb{A}(r)$ con un único estado final tal que $L(r) = L(\mathbb{A}(r))$.

Ahora, vemos la recíproca.

Teorema (Kleene)

Todo lenguaje regular es de la forma $L(r)$ para alguna $r \in \text{Regex}$.

Daremos un método **recursivo** para obtener a partir de un ϵ -NFA $\mathbb{A} = (\{q_0, \dots, q_r\}, \Sigma, \delta, q_0, \{q_f\})$ con un único estado final, una expresión regular que denota el lenguaje $L(\mathbb{A})$.

La recursión será en $|Q| = |\{q_0, \dots, q_r\}|$, así que iremos quitando estados para pasar a autómatas más chicos.

Teorema

Para toda expresión regular r , existe un ϵ -NFA $\mathbb{A}(r)$ con un único estado final tal que $L(r) = L(\mathbb{A}(r))$.

Ahora, vemos la recíproca.

Teorema (Kleene)

Todo lenguaje regular es de la forma $L(r)$ para alguna $r \in \text{Regex}$.

Daremos un método **recursivo** para obtener a partir de un ϵ -NFA $\mathbb{A} = (\{q_0, \dots, q_r\}, \Sigma, \delta, q_0, \{q_f\})$ con un único estado final, una expresión regular que denota el lenguaje $L(\mathbb{A})$.

La recursión será en $|Q| = |\{q_0, \dots, q_r\}|$, así que iremos quitando estados para pasar a autómatas más chicos.

Algoritmo del Teorema de Kleene

- $\mathbb{A} = (\{q_0, \dots, q_r\}, \Sigma, \delta, q_0, \{q_f\})$.
- $L(\mathbb{A}) =: L_{of}(Q)$

Algoritmo del Teorema de Kleene

- $\mathbb{A} = (\{q_0, \dots, q_r\}, \Sigma, \delta, q_0, \{q_f\})$.
- $L(\mathbb{A}) =: L_{0f}(Q)$  palabras que arrancan en q_0 y terminan en q_f .

Algoritmo del Teorema de Kleene

■ $\mathbb{A} = (\{q_0, \dots, q_r\}, \Sigma, \delta, q_0, \{q_f\})$.

■ $L(\mathbb{A}) =: L_{0f}(Q)$  palabras que arrancan en q_0 y terminan en q_f .

En general, supongamos $R \subseteq Q$.

Definición

$L_{nm}(R) :=$ palabras que arrancan en q_n y terminan en q_m , involucrando sólo estados en R .

Algoritmo del Teorema de Kleene

■ $\mathbb{A} = (\{q_0, \dots, q_r\}, \Sigma, \delta, q_0, \{q_f\})$.

■ $L(\mathbb{A}) =: L_{0f}(Q)$  palabras que arrancan en q_0 y terminan en q_f .

En general, supongamos $R \subseteq Q$.

Definición

$L_{nm}(R) :=$ palabras que arrancan en q_n y terminan en q_m , involucrando sólo estados en R .

Pueden pasar por q_n varias veces

$\alpha = \alpha_1\alpha_2\alpha_3 \in L_{nm}(R)$  $q_n \xrightarrow{\alpha_1} q_n \xrightarrow{\alpha_2} q_n \xrightarrow{\alpha_3} q_m$

Algoritmo del Teorema de Kleene

■ $\mathbb{A} = (\{q_0, \dots, q_r\}, \Sigma, \delta, q_0, \{q_f\})$.

■ $L(\mathbb{A}) =: L_{0f}(Q)$  palabras que arrancan en q_0 y terminan en q_f .

En general, supongamos $R \subseteq Q$.

Definición

$L_{nm}(R) :=$ palabras que arrancan en q_n y terminan en q_m , involucrando sólo estados en R .

Pueden pasar por q_n varias veces

$\alpha = \alpha_1\alpha_2\alpha_3 \in L_{nm}(R)$  $q_n \xrightarrow{\alpha_1} q_n \xrightarrow{\alpha_2} q_n \xrightarrow{\alpha_3} q_m$

Definición

$I_n(R) :=$ palabras que salen y vuelven a q_n **sin pasar** en el medio por él (e involucrando sólo estados en R).

$F_{nm}(R) :=$ palabras que salen de q_n y llegan a q_m **sin pasar** nuevamente por q_n (e involucrando sólo estados en R).

Algoritmo del Teorema de Kleene, primera capa

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

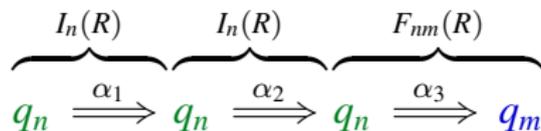
Algoritmo del Teorema de Kleene, primera capa

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$\begin{array}{ccccccc} & \overbrace{I_n(R)} & & \overbrace{I_n(R)} & & \overbrace{F_{nm}(R)} & \\ q_n & \xrightarrow{\alpha_1} & q_n & \xrightarrow{\alpha_2} & q_n & \xrightarrow{\alpha_3} & q_m \end{array}$$

Algoritmo del Teorema de Kleene, primera capa

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).



Luego

$$L_{nn}(R) := I_n(R)^*$$

$$L_{nm}(R) := I_n(R)^* F_{nm}(R) \quad \text{si } n \neq m$$

Algoritmo del Teorema de Kleene, ciclos iniciales

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

Algoritmo del Teorema de Kleene, ciclos iniciales

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta b} q_n$$

Algoritmo del Teorema de Kleene, ciclos iniciales

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta b} q_n \rightsquigarrow q_n \xrightarrow{a} q_t \xrightarrow{\beta} q_s \xrightarrow{b} q_n$$

Algoritmo del Teorema de Kleene, ciclos iniciales

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta b} q_n \rightsquigarrow q_n \xrightarrow{a} \underbrace{q_t \xrightarrow{\beta} q_s}_{L_{ts}(R \setminus \{q_n\})} \xrightarrow{b} q_n$$

Algoritmo del Teorema de Kleene, ciclos iniciales

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta b} q_n \rightsquigarrow q_n \xrightarrow{a} \underbrace{q_t \xrightarrow{\beta} q_s}_{L_{ts}(R \setminus \{q_n\})} \xrightarrow{b} q_n$$

Luego

$$I_n(R) := \sum_{\substack{q_n \xrightarrow{a} q_t \\ q_s \xrightarrow{b} q_n}} a L_{ts}(R \setminus \{q_n\}) b$$



UNC
Universidad
Nacional
de Córdoba



Algoritmo del Teorema de Kleene, ciclos iniciales

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta b} q_n \rightsquigarrow q_n \xrightarrow{a} \underbrace{q_t \xrightarrow{\beta} q_s}_{L_{ts}(R \setminus \{q_n\})} \xrightarrow{b} q_n$$

Luego

$$I_n(R) := \sum_{\substack{q_n \xrightarrow{a} q_t \\ q_s \xrightarrow{b} q_n}} a L_{ts}(R \setminus \{q_n\}) b + \sum_{q_n \xrightarrow{c} q_n} c \quad (n \neq t, s)$$



UNC
Universidad
Nacional
de Córdoba



Algoritmo del Teorema de Kleene, camino al final

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

Algoritmo del Teorema de Kleene, camino al final

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta} q_m$$

Algoritmo del Teorema de Kleene, camino al final

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta} q_m \rightsquigarrow q_n \xrightarrow{a} q_t \xrightarrow{\beta} q_m$$

Algoritmo del Teorema de Kleene, camino al final

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta} q_m \rightsquigarrow q_n \xrightarrow{a} \underbrace{q_t \xrightarrow{\beta} q_m}_{L_{tm}(R \setminus \{q_n\})}$$

Algoritmo del Teorema de Kleene, camino al final

- $L_{nm}(R) :=$ de q_n a q_m (sólo estados en R).
- $I_n(R) :=$ de q_n a sí mismo sin repetirlo (sólo estados en R).
- $F_{nm}(R) :=$ de q_n a q_m sin repetir q_n (sólo estados en R).

$$q_n \xrightarrow[\cancel{q_n}]{a\beta} q_m \rightsquigarrow q_n \xrightarrow{a} \underbrace{q_t \xrightarrow{\beta} q_m}_{L_{tm}(R \setminus \{q_n\})}$$

De igual modo,

$$F_{nm}(R) := \sum_{q_n \xrightarrow{a} q_t} a L_{tm}(R \setminus \{q_n\}) \quad (n \neq t, m)$$



UNC
Universidad
Nacional
de Córdoba



Algoritmo recursivo de Kleene

$$L_{nm}(R) := \emptyset \quad \text{si } n \text{ ó } m \text{ no están en } R$$

$$L_{nn}(R) := I_n(R)^*$$

$$L_{nm}(R) := I_n(R)^* F_{nm}(R) \quad \text{si } n \neq m$$

$$I_n(R) := \sum_{\substack{q_n \xrightarrow{a} q_t \\ q_s \xrightarrow{b} q_n}} a L_{ts}(R \setminus \{q_n\}) b + \sum_{q_n \xrightarrow{c} q_n} c \quad (n \neq t, s)$$

$$F_{nm}(R) := \sum_{q_n \xrightarrow{a} q_t} a L_{tm}(R \setminus \{q_n\}) \quad (n \neq t, m)$$