Lenguajes regulares:

Sea $M = (Q, \Sigma, \delta, q_0, F)$ un AFD

y sea $w = w_1 \dots w_n \in \Sigma^*$.

Def: M acepta a w si existe una sucesión $r_0, r_1, r_2, \dots, r_n \in Q$ con las siguientes propiedades:

(1) $r_0 = q_0$

(2) $r_{i+1} = \delta(r_i, w_i)$

(3) $r_n \in F$.

Si M es un AFD definimos

Def: $L(M) = \{w : w \text{ es aceptado por } M\}$ "lenguaje reconocido ó aceptado por M"

Def: Un lenguaje $S \subseteq \Sigma^*$ es un lenguaje regular si existe un AFD M con $L(M) = S$.

2 Lenguajes regulares:

Recuerde que un lenguaje es simplemente un conjunto $T \subseteq \Sigma^*$

Hay varios operaciones naturales entre lenguajes, por ejemplo

$$A \cup B = \{x : x \in A \text{ ó } x \in B\}$$ ← unión

$$A \circ B = \{x \cdot y : x \in A, y \in B\}$$ ← Concatenación

$$A^* = \bigcup_{k=0}^{\infty} \{x_1 x_2 \dots x_k : x_i \in A\}$$ ← Kleene star

Teorema: La colección de LENGUAJES REGULARES es cerrada bajo unión, concatenación y $*$

(es decir, si A, B regulares $\Rightarrow$ $A \cup B$, $A \circ B$, A^* regulares)

Si $\underbrace{A}_{M_A}$ regular y $\underbrace{B}_{M_B}$ regular $\Rightarrow$ $A \cup B$ regular

Dem. La idea es que vamos a ejecutar simultáneamente M_A y M_B a lo largo de la palabra $w_1 w_2 \dots w_n$ y aceptar si al menos una de las dos acepta. Para esto

Definimos un nuevo autómata $M = (Q, \Sigma, \delta, s, F)$ con

$$Q := Q_A \times Q_B$$

$$\delta\left(\left(q_A^{(i)}, q_B^{(j)}\right), t\right) := \left(\delta_A\left(q_A^{(i)}, t\right), \delta_B\left(q_B^{(j)}, t\right)\right)$$

$$s := \left(q_A^{(0)}, q_B^{(0)}\right)$$

NOTAR: $\neq F_A \times F_B$.

$$F := \{(q_A, q_B) : q_A \in F_A \text{ ó } q_B \in F_B\}$$

$L(M) = A \cup B$ así que $A \cup B$ es regular.

Se demuestra por inducción en la longitud.

b) A, B regulares $\Rightarrow$ $A \circ B$ regular

M_A, M_B máquinas testigo

Querríamos ejecutar M_A y luego B empezando en un estado de aceptación de A. La dificultad es que no sabemos donde partir la palabra $w = w_A | w_B$

Ejemplo: A – # par de a's

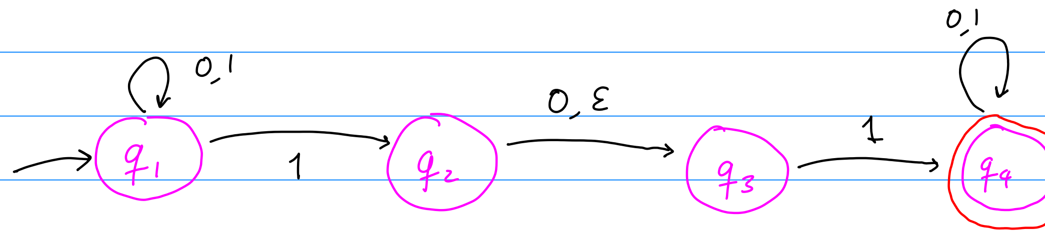
B – # par de b's

a a | b a a a b ✓ ✓ ← ACEPTA

a a b a a | a b ✓ ✗ ← NO, porque esta partida MAL.

Lo que queremos preguntarnos es si existe alguna forma de partir la palabra en dos fragmentos aceptables AB.

IDEA: Introduciremos autómatas no deterministas (NFA)



Hay tres cosas nuevas en el diagrama de estados:

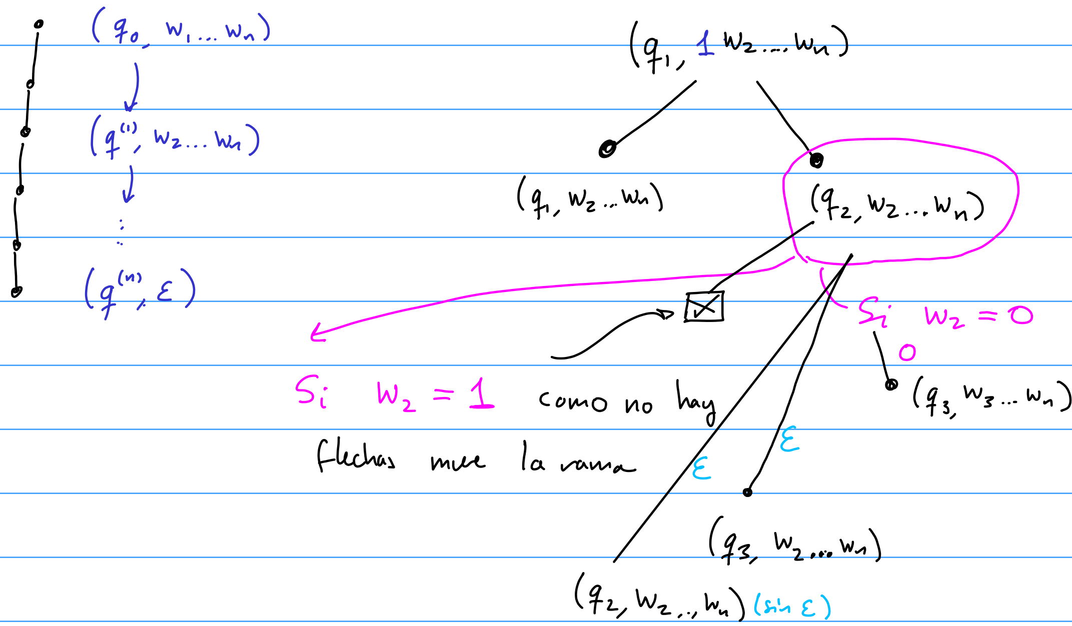
(i) Del mismo estado pueden salir VARIAS flechas con el mismo símbolo de Σ

(ii) Hay nuevas flechas con el símbolo ε.

(iii) Hay CERO flechas saliendo de q_2 con 1

Su interpretación computacional cambia completamente,

Determinista | No determinista:

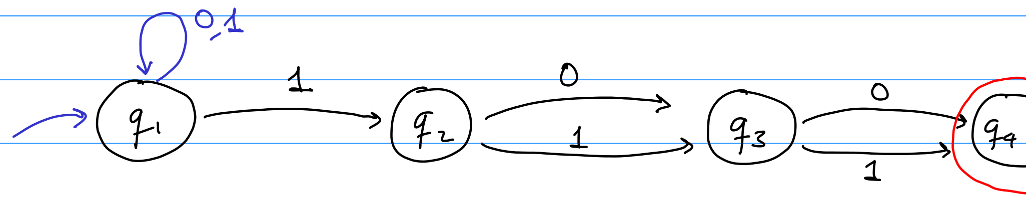


"En cada turno, después de leer el símbolo la máquina crea muchas copias de sí misma y sigue todas las posibilidades en paralelo. — PASO 2

Si en un estado que permite ε uso las flechas ε antes que nada (cambio de estado sin leer símbolo)" — PASO 1

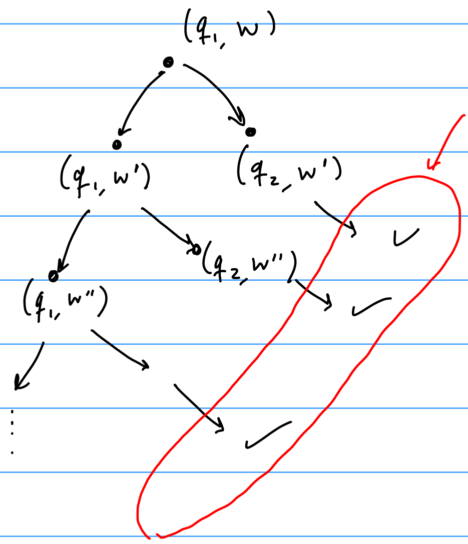
La máquina ACEPTA w si cualquiera de las ramas termina en algún estado de aceptación.

Ejemplo: Construya un NFA en $\Sigma = \{0,1\}$ que acepte palabras con 1 en la antepenúltima posición.



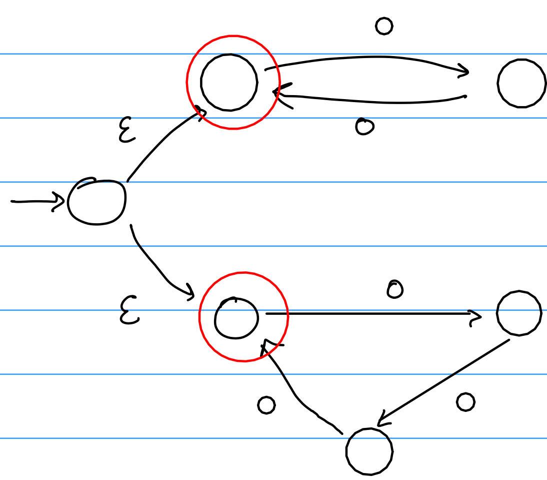
Si llegamos a q_4 al final entonces la palabra es

$1**$



Cualquiera de estas marca el final de la palabra.

Ejemplo: Qué hace este autómata en $\Sigma = \{0\}$



Def: Un autómata no determinista N es

$N = (Q, \Sigma, \delta, q_0, F)$

$\Sigma_\varepsilon := \Sigma \cup \{\varepsilon\}$ donde

$$\delta : Q \times \Sigma_\varepsilon \longrightarrow 2^Q := \{\text{subconjuntos de } Q\}$$

$$(q, s) \longmapsto \delta(q, s) \leftarrow \text{(puede ser vacío)}$$

conjunto de estados permisibles.

Este acepta a $w = w_1 \dots w_n$ si existen $y_1 \dots y_m$ en Σ_ε con $w = y_1 \dots y_m$ y una sucesión de estados $r_1, \dots, r_m$ con:

(i) $r_1 = q_0$
(ii) $r_{i+1} \in \delta(r_i, y_{i+1})$
(iii) $r_m \in F$.