Tema 8: Autómatas, lenguajes y gramáticas

Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

Ciencias de la Computación
(4º curso, Grado en Matemáticas)
Curso 2021–22

Contenido

Autómatas y complejidad

- Hasta ahora no hemos fijado límites a los recursos disponibles para resolver algorítmicamente un problema.
- Nuestra descripción de las máquinas de Turing no impone ninguna restricción sobre los movimientos en la cinta, el número de celdas disponible o la longitud de las computaciones. Ahora impondremos limitaciones a los recursos disponibles.
- Los autómatas finitos presentan uno de los escenarios más restrictivos. Son dispositivos muy simples, sin memoria:
  - Como las máquinas de Turing disponen de una cinta **finita** dividida en celdas. Cada celda contiene un símbolo de un cierto lenguaje Σ.
  - La cabeza lectora del autómata inspecciona la celda en la que se encuentra y a continuación, se mueve a la derecha y cambia de estado.
  - Cuando se para, el autómata acepta o rechaza la palabra leída.

Autómatas finitos (deterministas)

Un autómata finito (determinista), consta de

1. Un conjunto finito, Q, cuyos elementos se denominan **estados**.
2. Un alfabeto Σ.
3. $q_0 \in Q$ (se denomina **estado inicial**).
4. $F \subseteq Q$ (sus elementos son los **estados finales o de aceptación**).
5. Una función $\delta : Q \times \Sigma \to Q$ (llamada **función de transición**).

Autómatas finitos no deterministas

- Si en la definición anterior sustituimos la última condición por

  5. Una función $\delta : Q \times \Sigma \to \mathcal{P}(Q)$

  decimos que se trata de un autómata finito **no determinista**.
- Un autómata finito no determinista con ε–movimientos, ε–AFND, se define de manera similar a un AFND, salvo que en la definición anterior sustituimos la última condición por

  5. Una función $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$

Lenguaje aceptado por un AFD

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

- Se define $\hat{\delta} : Q \times \Sigma^* \to Q$ por recursión como sigue:

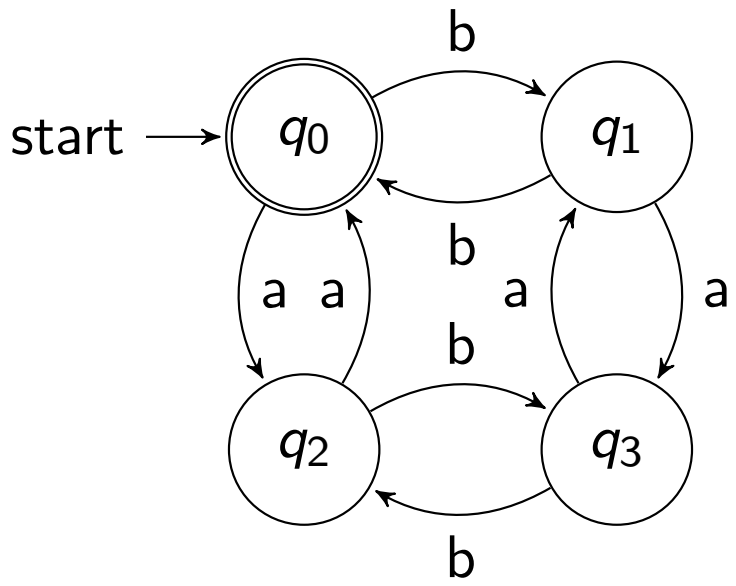
$$\hat{\delta}(q, \varepsilon) = q$$
$$\hat{\delta}(q, ws) = \delta(\hat{\delta}(q, w), s) \text{ con } w \in \Sigma^* \text{ y } s \in \Sigma$$

- $\hat{\delta}(q, v)$ determina el estado en que se encuentra el autómata después de leer la palabra v empezando en el estado q.
- Decimos que M acepta $v \in \Sigma^*$, si $\hat{\delta}(q_0, v) \in F$.
- El lenguaje aceptado por M es

$$L(M) = \{v \in \Sigma^* : \ \hat{\delta}(q_0, v) \in F\}$$

Ejemplo AFD

- Diagrama de transiciones:



- Tabla de transiciones:

		δ	a	b
$\rightarrow$	$\star$	q_0	q_2	q_1
		q_1	q_3	q_0
		q_2	q_0	q_3
		q_3	q_1	q_2

Lenguaje aceptado por un AFND

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

- Se define $\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$ por recursión como sigue:

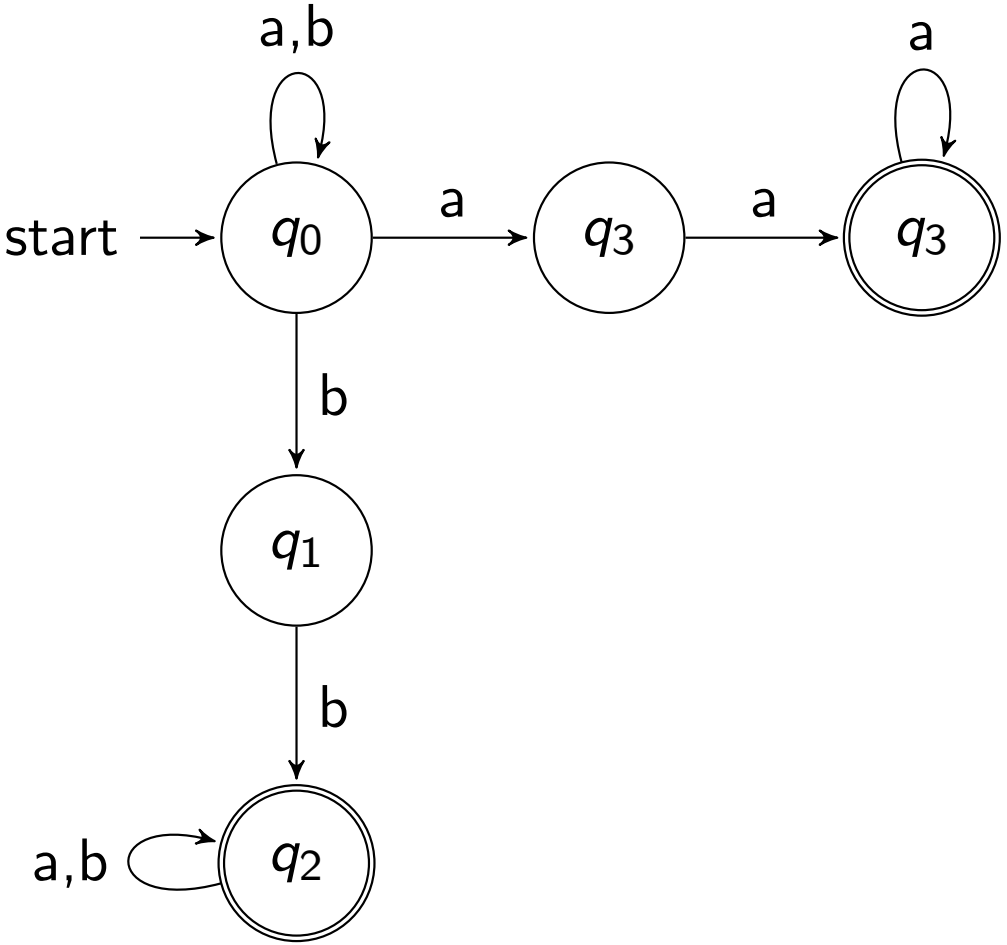
$$\hat{\delta}(q, \varepsilon) = \{q\}$$
$$\hat{\delta}(q, ws) = \{p \in Q : \text{ Existe } r \in \hat{\delta}(q, w) \text{ tal que } p \in \delta(r, s)\}$$

- $\hat{\delta}(q, v)$ determina el conjunto de estados en que puede encontrarse el autómata después de leer la palabra v, empezando en el estado q.
- Decimos que M acepta $v \in \Sigma^*$, si $\hat{\delta}(q_0, v) \cap F \neq \emptyset$.
- El lenguaje aceptado por M es

$$L(M) = \{v \in \Sigma^* : \hat{\delta}(q_0, v) \cap F \neq \emptyset\}$$

Ejemplo AFND

- Diagrama de transiciones:



Ejemplo AFND

- Tabla de transiciones:

	δ	a	b
$\rightarrow$	q_0	$\{q_0, q_3\}$	$\{q_0, q_1\}$
	q_1	$\emptyset$	$\{q_2\}$
	$\star q_2$	$\{q_2\}$	$\{q_2\}$
	q_3	$\{q_4\}$	$\emptyset$
	$\star q_4$	$\{q_4\}$	$\{q_4\}$

- Lenguaje aceptado, $L(M)$:

$$\{w \in \Sigma^* : \ \exists v_1,\ v_2\,(w = v_1 aav_2 \text{ o bien } w = v_1 bbv_2)\}$$

Lenguaje aceptado por un ε–AFND

Sea $M = (Q, \Sigma, \delta, q_0, F)$ un ε–AFND.

- Se define $\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$ por recursión como sigue:

$$\hat{\delta}(q, \varepsilon) = \varepsilon\text{–cl}(\{q\})$$
$$\hat{\delta}(q, ws) = \varepsilon\text{–cl}(\{p \in Q : \text{ Existe } r \in \hat{\delta}(q, w) \text{ tal que } p \in \delta(r, s)\})$$

- $\varepsilon\text{–cl}(q) = \bigcup_{p \in \mathbb{N}} C_n(q)$, donde

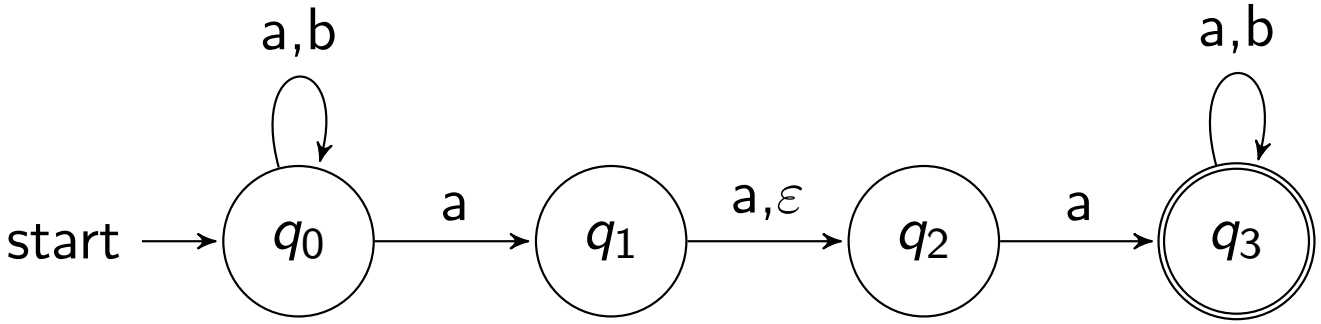
$$C_0(q) = \{q\} \cup \delta(q, \varepsilon)$$
$$C_{n+1}(q) = \bigcup_{p \in C_n(q)} \delta(p, \varepsilon)$$

- El lenguaje aceptado por M es

$$L(M) = \{v \in \Sigma^* : \ \hat{\delta}(q_0, v) \cap F \neq \emptyset\}$$

Ejemplo ε–AFND (I)

- Diagrama de transiciones:



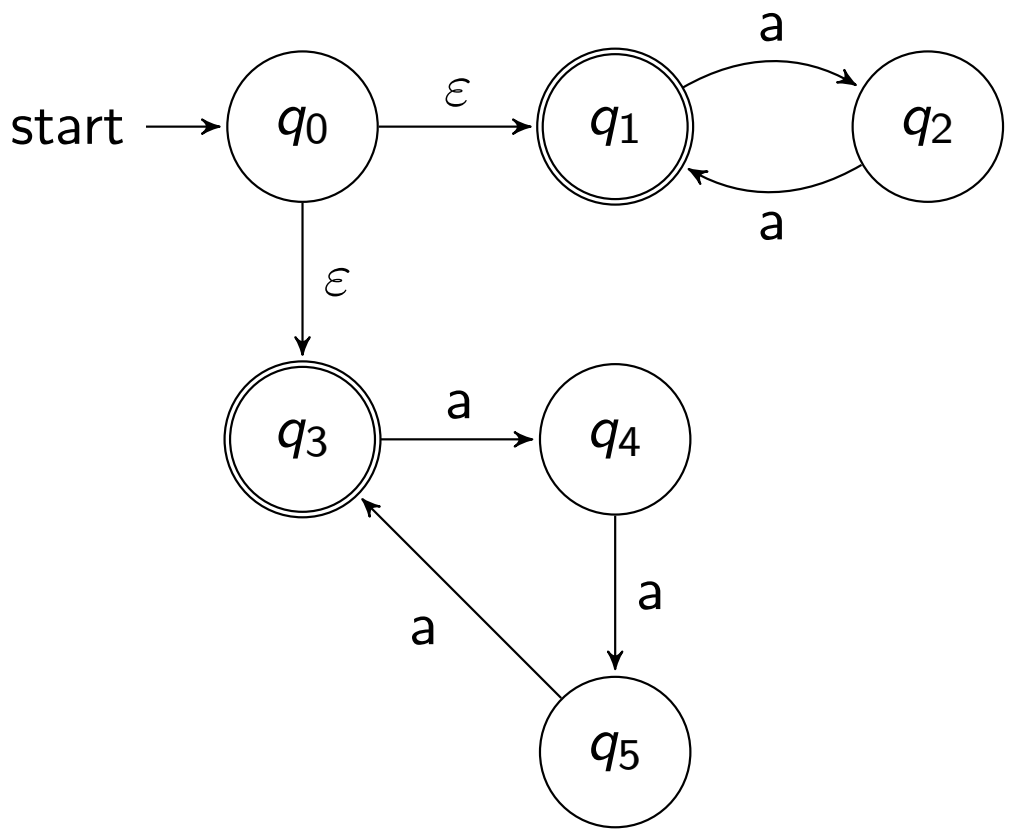
- Tabla de transiciones:

	δ	a	b	ε
$\rightarrow$	q_0	$\{q_0, q_1\}$	$\{q_0\}$	$\emptyset$
	q_1	$\{q_2\}$	$\emptyset$	$\{q_2\}$
	q_2	$\emptyset$	$\{q_3\}$	$\emptyset$
	$\star q_3$	$\{q_3\}$	$\{q_3\}$	$\emptyset$

- Lenguaje aceptado: $L(M) = ??$

Ejemplo ε–AFND (II)

- Diagrama de transiciones:



- Lenguaje aceptado:

$$\{a^k : \ k \text{ es divisible por 2 o por 3}\}$$

Equivalencia entre AFND y AFD

- **Teorema**. Si M es un AFND entonces $L(M)$ es aceptado por un AFD.
- Demostración: Si $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ es un AFND $L = L(M)$ definimos el siguiente AFD

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

  - $Q' = \mathcal{P}(Q)$ y $q_0' = \{q_0\}$.
  - $\delta'(\{q_1, \ldots, q_r\}, a) = \bigcup_{j=1}^{r} \delta(q_j, a)$.
  - $F' = \{B \subseteq Q : \ B \cap F \neq \emptyset\}$.

Por inducción en la longitud de $v \in \Sigma^*$ se prueba

$$\forall v \in \Sigma^*, \quad \hat{\delta}'(q_0', v) = \hat{\delta}(q_0, v)$$

Luego $L(M) = L(M')$:

$$\begin{array}{lcll} v \in L(M) & \Leftrightarrow & \hat{\delta}(q_0, v) \cap F \neq \emptyset & \Leftrightarrow \quad \hat{\delta}'(q_0', v) \cap F \neq \emptyset \\ & \Leftrightarrow & \hat{\delta}'(q_0', v) \in F' & \Leftrightarrow \quad v \in L(M') \end{array}$$

Equivalencia entre ε–AFND y AFD

- **Teorema**. Si M es un ε–AFND entonces $L(M)$ es aceptado por un AFD.
- Demostración: Si $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ es un ε–AFND $L = L(M)$ definimos el siguiente AFD

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

  - $Q' = \mathcal{P}(Q)$ y $q_0' = \varepsilon\text{–cl}(\{q_0\})$.
  - $\delta'(\{q_1, \ldots, q_r\}, a) = \bigcup_{j=1}^{r} \varepsilon\text{–cl}(\delta(q_j, a))$.
  - $F' = \{B \subseteq Q : \ B \cap F \neq \emptyset\}$.

Por inducción en la longitud de $v \in \Sigma^*$ se prueba

$$\forall v \in \Sigma^*, \quad \hat{\delta}'(q_0', v) = \hat{\delta}(q_0, v)$$

Luego $L(M) = L(M')$:

$$\begin{array}{rcccl} v \in L(M) & \Leftrightarrow & \hat{\delta}(q_0, v) \cap F \neq \emptyset & \Leftrightarrow & \hat{\delta}'(q_0', v) \cap F \neq \emptyset \\ & \Leftrightarrow & \hat{\delta}'(q_0', v) \in F' & \Leftrightarrow & v \in L(M') \end{array}$$

Operaciones con lenguajes

Sea Σ un alfabeto. Utilizaremos las siguientes notaciones:

- $\emptyset$ denota al lenguaje vacío.
- $\underline{\varepsilon} = \{\varepsilon\}$.
- Para cada $a \in \Sigma$, $\underline{a} = \{a\}$.

Si L_1 y L_2 son lenguajes sobre Σ, entonces

- (**Concatenación**) $L_1 \cdot L_2 = \{vw : \ v \in L_1 \text{ y } w \in L_2\}$.
- (**Unión**) $L_1 + L_2 = L_1 \cup L_2$.

Si L es un lenguaje sobre Σ, se define la **estrella (o cierre) de Kleene** de L como el lenguaje $L^* = \bigcup_{n \in \mathbb{N}} L^n$ donde

$$L^n = \begin{cases} \underline{\varepsilon} & \text{si } n = 0 \\ L \cdot L^k & \text{si } n = k + 1 \end{cases}$$

Autómatas y operaciones con lenguajes (I)

- Si L_1 y L_2 son lenguajes sobre Σ aceptados por AFDs entonces $L_1 + L_2$ también es aceptado por un AFD.
- Demostración: Supongamos que L_1 y L_2 son aceptados (respectivamente) por los ε–AFNDs:

$$M_1 = \langle Q_1, \Sigma, \delta_1, q_1, F_1 \rangle \text{ y } M_2 = \langle Q_2, \Sigma, \delta_2, q_2, F_2 \rangle$$

Podemos suponer $Q_1 \cap Q_2 = \emptyset$. Sea M el ε–AFND definido por:

- $Q = \{q_0\} \cup Q_1 \cup Q_2$, siendo $q_0 \notin Q_1 \cup Q_2$.
- q_0 es el estado inicial y $F = F_1 \cup F_2$.
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ se define para cada $q \in Q$ y $s \in \Sigma \cup \{\varepsilon\}$ por

$$\delta(q,s) = \begin{cases} \delta_1(q,s) & \text{si } q \in Q_1 \\ \delta_2(q,s) & \text{si } q \in Q_2 \\ \{q_1,\ q_2\} & \text{si } q = q_0 \text{ y } s = \varepsilon \\ \emptyset & \text{si } q = q_0 \text{ y } s \neq \varepsilon \end{cases}$$

Autómatas y operaciones con lenguajes (II)

- Si L_1 y L_2 son lenguajes sobre Σ aceptados por AFDs entonces $L_1 L_2$ también es aceptado por un AFD.
- Demostración: Supongamos que L_1 y L_2 son aceptados (respectivamente) por los ε–AFNDs:

$$M_1 = \langle Q_1, \Sigma, \delta_1, q_1, F_1 \rangle \text{ y } M_2 = \langle Q_2, \Sigma, \delta_2, q_2, F_2 \rangle$$

Podemos suponer $Q_1 \cap Q_2 = \emptyset$. Sea M el ε–AFND definido por:

- $Q = Q_1 \cup Q_2$.
- q_1 es el estado inicial y $F = F_2$.
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ se define para cada $q \in Q$ y $s \in \Sigma \cup \{\varepsilon\}$ por

$$\delta(q,s) = \begin{cases} \delta_1(q,s) & \text{si } q \in Q_1 \text{ y } q \notin F_1 \\ \delta_1(q,s) & \text{si } q \in F_1 \text{ y } s \neq \varepsilon \\ \delta_1(q,s) \cup \{q_2\} & \text{si } q \in F_1 \text{ y } s = \varepsilon \\ \delta_2(q,s) & \text{si } q \in Q_2 \end{cases}$$

Autómatas y operaciones con lenguajes (III)

- Si L_1 es un lenguaje sobre Σ aceptado por un AFD, entonces L_1^* también es aceptado por un AFD.
- Demostración: Supongamos que L_1 es aceptado por el ε–AFND: $M_1 = \langle Q_1, \Sigma, \delta_1, q_1, F_1 \rangle$. Entonces L_1^* es aceptado por el ε–AFND, M definido por:
  - $Q = \{q_0\} \cup Q_1$, siendo $q_0 \notin Q_1$.
  - q_0 es el estado inicial y $F = \{q_0\} \cup F_1$.
  - $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ se define para cada $q \in Q$ y $s \in \Sigma \cup \{\varepsilon\}$ por

$$\delta(q,s) = \begin{cases} \delta_1(q,s) & \text{si } q \in Q_1 \text{ y } q \notin F_1 \\ \delta_1(q,s) & \text{si } q \in F_1 \text{ y } s \neq \varepsilon \\ \delta_1(q,s) \cup \{q_1\} & \text{si } q \in F_1 \text{ y } s = \varepsilon \\ F_1 \cup \{q_1\} & \text{si } q = q_0 \text{ y } s = \varepsilon \\ \emptyset & \text{si } q = q_0 \text{ y } s \neq \varepsilon \end{cases}$$

Autómatas y operaciones con lenguajes (IV)

Otras operaciones:

- Si L_1 y L_2 son lenguajes sobre Σ aceptados por AFDs entonces $L_1 \cap L_2$ también es aceptado por un AFD.
- Si L_1 es un lenguaje sobre Σ aceptado por un AFD entonces el complementario de L_1, $\bar{L_1}$, también es aceptado por un AFD, siendo

$$\bar{L_1} = \{v \in \Sigma^* : \ v \notin L_1\}$$

- Demostración: Ejercicio.

Lenguajes regulares

- La clase de los lenguajes regulares sobre un alfabeto Σ es la menor clase de subconjuntos de Σ^* que
  1. Contiene los lenguajes $\emptyset$ y $\underline{a}$, para cada $a \in \Sigma$; y
  2. Es cerrada bajo unión, concatenación y cierre de Kleene, es decir
     - Si L_1 y L_2 son lenguajes regulares entonces $L_1 + L_2$ y $L_1 \cdot L_2$ son regulares.
     - Si L es un lenguaje regular entonces L^* también lo es.
- Ejemplos:
  - $\underline{\varepsilon}$ es un lenguaje regular, ya que $\underline{\varepsilon} = \emptyset^*$.
  - Todo lenguaje finito es regular.

Expresiones regulares

Dado un alfabeto Σ, el conjunto de las **expresiones regulares** sobre Σ es el menor conjunto tal que:

1. $\emptyset$ es una expresión regular.
2. ε es una expresión regular.
3. Para cada $a \in \Sigma$, a es una expresión regular.
4. Si $\mathbf{e}$ y $\mathbf{e}$ son expresiones regulares entonces $(\mathbf{e} + \mathbf{d})$ y $(\mathbf{e} \cdot \mathbf{d})$ también lo son.
5. Si $\mathbf{e}$ es una expresión regular entonces $\mathbf{e}^*$ también lo es.

Observación: Cada expresión regular es una palabra sobre el alfabeto

$$\tilde{\Sigma} = \Sigma \cup \{\varepsilon, \emptyset, +, \cdot, *, (,)\}$$

Lenguajes y expresiones regulares

Dada un expresión regular $\mathbf{e}$ sobre Σ, el lenguaje generado por $\mathbf{e}$, denotado por $L(\mathbf{e})$, se define recursivamente como sigue:

1. $L(\emptyset) = \emptyset$.
2. Si $\mathbf{e} = \varepsilon$ entonces $L(\mathbf{e}) = \underline{\varepsilon}$.
3. Para cada $a \in \Sigma$, si $\mathbf{e} = a$ entonces $L(\mathbf{e}) = \underline{a}$.
4. Si $\mathbf{e} = (\mathbf{e}_1 + \mathbf{e}_2)$, entonces $L(\mathbf{e}) = L(\mathbf{e}_1) + L(\mathbf{e}_2)$.
5. Si $\mathbf{e} = (\mathbf{e}_1 \cdot \mathbf{e}_2)$, entonces $L(\mathbf{e}) = L(\mathbf{e}_1) \cdot L(\mathbf{e}_2)$.
6. si $\mathbf{e} = \mathbf{d}^*$, entonces $L(\mathbf{e}) = L(\mathbf{d})^*$.

- **Lema**. Un lenguaje L sobre Σ es regular si y sólo si existe una expresión regular $\mathbf{e}$ sobre Σ tal que $L = L(\mathbf{e})$.

Autómatas y lenguajes regulares

- **Teorema**. Todo lenguaje regular es aceptado por algún AFD.
- Demostración: Si L es un lenguaje regular, basta probar que existe un ε–AFND, M tal que $L(M) = L$. Puesto que los
  - Ya hemos visto que los lenguaje básicos $\emptyset$, $\underline{\varepsilon}$ y $\underline{a}$ ($a \in \Sigma$) son aceptados por autómatas finitos.
  - Puesto que los lenguajes regulares se obtienen a partir de los básicos mediante unión, concatenación y cierre de Kleene, el resultado se sigue de las propiedades de cierre de la clases de los lenguajes aceptados por AFDs.
- El recíproco de este teorema es cierto (es un teorema de Kleene). Para probarlo introducimos una noción más general de AF capaz de leer su entrada por bloques.

Autómatas finitos no deterministas generalizados

- Un autómata finito no determinista generalizado, AFNDG, es una tupla $M = (Q, \Sigma, \delta, q_I, q_A)$, donde
  1. Q es un conjunto finito de **estados**, Σ es un alfabeto y q_I, $q_A \in Q$, se denominan, respectivamente, **estado inicial** y **estado de aceptación**.
  2. Una función de transición $\delta : (Q - \{q_A\}) \times (Q - \{p_I\}) \to \mathbf{R}$.
     - $\mathbf{R}$ es el conjunto de las expresiones regulares sobre Σ.
- El lenguaje aceptado por M es el conjunto $L(M)$ formado por los $w \in \Sigma^*$ tales que existe una sucesión de palabras $w_1, \ldots, w_k \in \Sigma^*$ y una sucesión de estados de M, $q_0, q_1, \ldots, q_k$ tales que:
  - $w = w_1 \cdots w_k$.
  - $q_0 = q_I$ y $q_r = q_A$.
  - Para cada $i : 1 \leq i \leq k$, $w_i \in L(\mathbf{r}_i)$, siendo $\mathbf{r}_i = \delta(q_{i-1}, q_i)$.

ε–AFND vs AFNDG

- Dado un ε-AFND $M = (Q, \Sigma, \delta, q_0, F)$ existe un AFNDG $M^g = (Q', \Sigma, \delta^g, q_I, q_A)$, tal que $L(M) = L(M^g)$.

  Basta tomar $Q' = Q \cup \{q_I, q_A\}$ y definir δ^g como

$$\delta^g(q, q') = \begin{cases} \underline{\varepsilon} & \text{si } q = q_I \text{ y } q' = q_0 \\ \underline{\varepsilon} & \text{si } q \in F \text{ y } q' = q_A \\ \biguplus_{q \in \delta(a, q')} \underline{a} & \text{si } q, q' \in Q \\ \emptyset & \text{en otro caso} \end{cases}$$

- **Lema**. Dado un AFNDG $M_1 = (Q_1, \Sigma, \delta_1, q_I, q_A)$ con $|Q_1| > 2$, existe otro AFNDG $M_2 = (Q_2, \Sigma, \delta_2, q_I, q_A)$, tal que $L(M_1) = L(M_2)$, $Q_1 \subseteq Q_2$ y $|Q_2| = |Q_1| - 1$.
  - Basta fijar $\hat{q} \in Q_1 - \{q_I, q_A\}$ y definir $Q_2 = Q_1 - \{\hat{q}\}$ y $\delta_2 : (Q_2 - \{q_A\}) \times (Q_2 - \{q_I\}) \to \mathbf{R}$ como

$$\delta_2(q, q') = \mathbf{r}_1(\mathbf{r}_2)^* \mathbf{r}_3 + \mathbf{r}_4$$

  para cada $q, q' \in Q_2$ con $q \neq q_A$ y $q' \neq q_I$, siendo $\mathbf{r}_1 = \delta_1(q, \hat{q})$, $\mathbf{r}_2 = \delta_1(\hat{q}, \hat{q})$, $\mathbf{r}_3 = \delta_1(\hat{q}, q')$ y $\mathbf{r}_4 = \delta_1(q, q')$.

Teorema de Kleene

- **Teorema** (Kleene) Sea M un ε-AFND. Entonces $L(M)$ es regular.
- Demostración: Dado un ε-AFND, M, existe un AFNDG, M^g tal que $L(M) = L(M^g)$ y M^g tiene más de 2 estados. Aplicando el lema anterior reiteradamente para eliminar uno a uno los estados distintos de q_I y q_A, construimos un AFNDG, $M' = (\{q_I, q_A\}, \Sigma, \delta', q_I, q_A)$ con solo dos estados (q_I y q_A) y tal que $L(M') = L(M^g)$. Pero entonces, si $\mathbf{r} = \delta'(q_I, q_A)$ se tiene que

$$L(M') = L(\mathbf{r})$$

Luego $L(M) = L(M^g) = L(M') = L(\mathbf{r})$ es regular.

Gramáticas

- Hasta ahora sólo hemos considerado el problema de **reconocer** si una palabra sobre un alfabeto σ pertenece o no a un lenguaje regular L.
- Los autómatas proporcionan algoritmos de decisión para este problema.
- A través de las expresiones regulares hemos encontrado una forma de hacer explícita la estructura "sintáctica" común a todas las palabras de un lenguaje regular.
- Ahora pasaremos a considerar el problema de **generar** automáticamente palabras de un lenguaje dado.
- Para ello necesitaremos algún modo de hacer explícita la forma "sintáctica" común a todas estas palabras.
- Recordemos que las gramáticas formales proporcionan una herramienta para esta tarea.

Procesos semi–Thue

- Una **producción** (o **regla de reescritura**) sobre un alfabeto Σ es un par de palabras sobre Σ, $(g, \bar{g})$.
  - La regla $(g, \bar{g})$ se representará por $g \to \bar{g}$.
- Si P es un producción $g \to \bar{g}$, y $u, v \in \Sigma^*$, escribiremos $u \Rightarrow_P v$ para expresar que

$$\text{Existen } r, s \in \Sigma^* \text{ tales que } u = rgs \text{ y } v = r\bar{g}s$$

- Un **proceso semi–Thue** es un conjunto finito de producciones.
- Si Φ es un proceso semi–Thue y $u, v \in \Sigma^*$ escribimos:
  - $u \Rightarrow_\Phi v$ si existe $P \in \Phi$ tal que $u \Rightarrow_P v$.
  - $u \overset{\star}{\Rightarrow}_\Phi v$ si existe una sucesión $w_1, \ldots, w_n \in \Sigma^*$ tal que $w_1 = u$, $w_n = v$ y

$$\text{para cada } j = 1, \ldots, n-1, \quad w_j \Rightarrow_\Phi w_{j+1}$$

(decimos que $w_1, \ldots, w_n$ es una derivación de v a partir de u.

Gramáticas

- Una **gramática** es un proceso semi–Thue en el que los símbolos del alfabeto se han dividido en dos clases: **terminales** y **no terminales** (o variables), y existe un símbolo no terminal distinguido que llamamos **símbolo inicial**.
- Una gramática $\Gamma = \langle V, T, \Phi, S \rangle$ está determinada por:
  - Un alfabeto $\Sigma = V \cup T$ con $V \cap T = \emptyset$ (V es el conjunto de variables, T el de símbolos terminales)
  - Un símbolo $S \in V$ (símbolo inicial).
  - Un conjunto de reglas de reescritura, Φ.
- El **lenguaje generado** por la gramática $\Gamma = \langle V, T, \Phi, S \rangle$ es

$$L(\Gamma) = \{u \in T^* : \ S \stackrel{\star}{\Rightarrow}_{\Phi} u\}$$

Gramáticas independientes del contexto

- Una gramática $\Gamma = \langle V, T, \Phi, S \rangle$ es **independiente del contexto** si para cada producción $u \to v$ de Φ, $u \in V$.
  - Es decir, la cabeza de cada regla de reescritura está formada por un único símbolo no terminal.
- Un lenguaje L se denomina **independiente del contexto** si existe una gramática Γ independiente del contexto, $L(\Gamma) = L$.

Gramáticas regulares

- Una gramática $\Gamma = \langle V, T, \Phi, S \rangle$ independiente del contexto es **regular** si para cada producción $u \to v$ de Φ, $v \in T^*(V + \underline{\varepsilon})$.
  - Es decir, la cabeza de cada regla de reescritura está formada por un único símbolo no terminal y el cuerpo es una palabra sin variables, o que contiene una única variable como último símbolo.
- Dada una gramática regular $\Gamma = \langle V, T, \Phi, S \rangle$ siempre existe otra gramática regula Γ' tal que
  1. $L(\Gamma) = L(\Gamma')$, y
  2. Toda producción $u \to v$, verifica que $v \in \underline{\varepsilon} + T(V + \underline{\varepsilon})$.

Gramáticas y lenguajes regulares

- **Teorema**. Un lenguaje L es regular si y sólo si existe una gramática regular Γ que lo genera (es decir, $L = L(\Gamma)$).
- Demostración: Sea $M = (Q, \Sigma, \delta, q_o, F)$ un AFD tal que $L(M) = L$. Sea Γ la gramática regular, $\Gamma = \langle Q, \Sigma, \Phi, q_0 \rangle$, con

$$\Phi = \{q \to ap : \ \delta(q, a) = p\} \cup \{q \to \varepsilon : \ \ q \in F\}$$

Entonces, $L(\Gamma) = L(M)$.
Recíprocamente, si $L = L(\Gamma)$ para una gramática regular $\Gamma = \langle V, T, \Phi, S \rangle$, ("normalizada") definamos el siguiente ε–AFND, $M = \langle Q, \Sigma, \delta, q_0, F \rangle$, donde

- $Q = V \cup \{f\}$, siendo $f \notin V \cup T$.
- $q_0 = S$ y $F = \{f\}$.
- Para cada $a \in \Sigma \cup \{\varepsilon\}$ y $p \in Q$,

$$p \in \delta(q, a) \Leftrightarrow \begin{cases} q \to ap \in \Phi \\ \text{ o bien,} \\ (p = f \text{ y } q \to a \in \Phi) \end{cases}$$

El lema del bombeo para lenguajes regulares

- **Lema**. Sea M un AFD con n estados y alfabeto Σ. Si $x \in L = L(M)$ y $|x| \geq n$ entonces existen u, v, $w \in \Sigma^*$ tales que
  1. $x = uvw$, con $v \neq \varepsilon$ y $|uv| \leq n$.
  2. Y, para todo $k \in \mathbb{N}$, $uv^k w \in L$.
- Demostración: Puesto que $|x| \geq n$ y M tiene n estados la computación de M sobre x pasa por $n+1$ estados (incluyendo el inicial). Por tanto existe un estado q de M que se alcanza más de una vez. Entonces podemos escribir $x = uvw$ siendo u, v y w tales que

$$\hat{\delta}(q_0, u) = q = \hat{\delta}(q, v) \quad \text{y} \quad \hat{\delta}(q, w) \in F$$

Es fácil comprobar que para todo $k \in \mathbb{N}$ se tiene

$$\hat{\delta}(q_0, uv^k w) = \hat{\delta}(q_0, uvw) \in F$$

y, por tanto, $uv^k w \in L$. Si elegimos q como el estado que se repite primero y uv de modo que el estado q se alcance sólo dos veces en uv, obtenemos $|uv| \leq n$.

Consecuencias del lema del bombeo

- **Teorema**. Si M es AFD con n estados entonces:
  1. $L(M) \neq \emptyset$ si y sólo si existe $x \in L$ con $|x| < n$.
  2. $L(M)$ es infinito si y sólo si existe $x \in L$ con $n \leq |x| < 2n$.
- **Demostración:** Ejercicio.
- El lenguaje $L = \{a^i b^j : \ i, j \in \mathbb{N},\ i = j\}$ no es regular. En efecto, si L es regular, tomando n como en el lema del bombeo y $x = a^n b^n$ existen $u, v, w \in \{a, b\}^*$ tales que $v \neq \varepsilon$, $x = uvw$, $|uv| \leq n$ y

$$\text{para todo } k \in \mathbb{N},\ uv^k w \in L.$$

Puesto que $|uv| \leq n$, y $uvw = x = a^n b^n$, resulta que $v = a^m$, para cierto $m \leq n$. En consecuencia, $uv^n w \in L$, lo cual es absurdo ya que w contiene a lo sumo n b's y uv^n al menos $n \cdot m$ a's.