Teoría de Lenguajes

Gianfranco Zambonni

23 de enero de 2023

Teoría de Lenguajes

Gianfranco Zambonni

23 de enero de 2023

Índice

1. Introducción

1.1. Relaciones

Dados dos conjuntos A y B, se llama **relación** $R : A \to B$ de A en B a todo subconjutno de $A \times B$, es decir $R \subset A \times B$.

Dos elementos $a \in A$ y $b \in B$ están relacionados si $(a, b) \in R$ y lo notamos aRb.

Si $A = B$, se dice que R es una relación sobre A y se dice que:

- es **reflexica** cuando $\forall a,\ aRa$.
- es **simétrica** cuando $\forall a, b \in A,\ aRb \implies bRa$.
- es **transitiva** cuando $a, b, c \in A,\ aRb \land bRc \implies aRc$.

Relación de equivalencia: Una relación $R : A \to A$ es de **equivalencia** cuando es reflexiva, simétrica y transitiva. Este tipo de relaciones particiona a A en subconjuntos disjuntos llamados **clases de equivalencia**.

1.1.1. Operaciones

Composición de relaciones: Si $R : A \to B$ y $S : B \to C$ son relaciones, entonces la composición de R y S es la relación $S \circ R : A \to C$ definida por:

$$S \circ R = \{(a, c) \mid a \in A,\ c \in C : \exists b \in B,\ aRb \land bSc\}$$

Relación de identidad: La relación de identidad sobre A es la relación $id_A : A \to A$ definida por: $id_A = \{(a, a) \mid a \in A\}$.

- La relación de identidad el el elemento neutro de la composición de relaciones.

Relación de potencia: Dado $R : A \to A$ se define la relación de potencia $R^k : A \to A$ como la composición de k copias de R:

$$R^n = \begin{cases} id_A & \text{si } n = 0 \\ R \circ R^{n-1} & \text{si } n > 0 \end{cases}$$

Clausura transitiva/positiva: Dada una relación $R : A \to A$ se define la clausura transitiva de R como la relación R^+ definida por:

$$R^+ = \bigcup_{n=1}^{\infty} R^n$$

La clausura transitiva de R cumple las siguientes propiedades:

1. $R \subseteq R^+$

2. R^+ es transitiva

DEMOSTRACIÓN

Si aR^+b entonces existe una secuencia de elementos $a = a_0, a_1, \ldots, a_n = b$ tales que a_iRa_{i+1} para todo $i \in [0, n-1]$.

Análogamente, como bR^+c existe una secuencia de elementos

$$b = b_0, b_1, \ldots, b_m = c$$

tales que b_iRb_{i+1} para todo $i \in [0, m-1]$.

Entonces $aR^{n+m}c$ pues puedo armar la secuencia

$$a = a_0, a_1, \ldots, a_n, b_1 \ldots b_m = c$$

.

Luego como $R^{n+m} \subseteq R^+$ vale que aR^+c.

3. Para toda relación $G : A \to A$ tal que $R \subseteq G \wedge G$ es transitiva, entonces $R^+ \subseteq G$, es decir R^+ es la relación transitiva más pequeña que contiene a R.

DEMOSTRACIÓN

Si aR^+b entonces existe una secuencia de elementos $a = a_0, a_1, \ldots, a_n = b$ tales que a_iRa_{i+1} para todo $i \in [0, n-1]$.

Como $R \subseteq G$ entonces a_iGa_{i+1} para todo $i \in [0, n-1]$. Como G es transitiva entonces la aplicación repetida de la transitividad nos lleva a que a_1Ga_n, por lo que aGb.

Clausura transitiva reflexiva:

$$R^* = R^+ \cup id_A = \bigcup_{n=0}^{\infty} R^n$$

Observaciones:

- Si A es un conjunto finito, entonces todas las relaciones $R : A \to A$ son finitas.
- Si R es reflexiva, entonces $R^* = R^+$.

1.2. Alfabetos

Alfabeto: Un alfabeto es un conjunto finito de símbolos.

Cadena: Una cadena sobre un alfabeto Σ es una secuencia finita de símbolos de Σ. Los símbolos son notados respetando el orden de la secuencia.

Concatenación: Es una operación entre un símbolo del alfabeto Σ y una cadena sobre dicho alfabeto:

$$\circ : \Sigma \times \{\text{cadenas sobre } \Sigma\} \to \{\text{cadenas de } \Sigma\}$$

- La cadena nula λ es el elemento neutro de la concatenación.

Clausura de Kleene de Σ: Σ^*

- $\lambda \in \Sigma^*$
- $\alpha \in \Sigma^* \implies \forall\, a \in \Sigma,\ a \circ \alpha \in \Sigma^*$

Clausura positiva de Σ: $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$

1.3. Lenguajes

Lenguaje: Un lenguaje es un conjunto de cadenas sobre un alfabeto Σ.

Concatenación de lenguajes: Si L_1 y L_2 son lenguajes definidos sobre los alfabetos Σ_1 y Σ_2 respectivamente, entonces la concatenación de L_1 y L_2 es un lenguaje L_1L_2 sobre el alfabeto $\Sigma_1 \cup \Sigma_2$ dfinido de la siguiente manera:

$$L_1L_2 = \{\alpha\beta \mid \alpha \in L_1,\ \beta \in L_2\}$$

Clausura de Kleene L^*:

$L^0 = \{\lambda\}$

$L^n = LL^{n-1}$ para $n >= 1$

$L^* = \bigcup\limits_{n=0}^{\infty} L^n$

Clausura positiva L^+:

$L^+ = \bigcup\limits_{n=1}^{\infty} L^n$

Observaciones:

- $L^+ = LL^*$
- $L^* = L^+ \cup \{\lambda\}$
- Si L es un lenguaje definido sobre Σ entonces $L \subseteq \Sigma^*$

1.4. Gramáticas

Una gramática es una 4-tupla (V_N, V_T, P, S) donde:

- V_N es un conjunto finito de símbolos no terminales.
- V_T es un conjunto finito de símbolos terminales.
- P es un conjunto finito de reglas de producción: Son pares ordenados $\alpha \to \beta$ donde

$$\alpha \in (V_N \cup V_T)^* V_N (V_N \cup V_T)^* \text{ y } \beta \in (V_N \cup V_T)^*$$

- $S \in V_N$ es el símbolo inicial.

Dada una producción $A \to \alpha \in P$, se denomina a A como **cabeza** de la producción y a α como su **cuerpo**.

Derivación: El el proceso por el cual se obtiene una cadena a partir de un símbolo inicial remplazando recursivamente símbolos no terminales por cuerpos de producciones en P cuya cabeza coincida con los símbolos que están siendo remplazados.

Forma setencial de una grámatica: Se llama forma sentencial a cualquier derivación de la grámatica:

- S es una forma setencial de G
- Si $\alpha\beta\gamma$ es una forma setencial de G y $\beta \to \delta \in P$ entonces $\alpha\delta\gamma$ es una forma setencial de G.

Derivación directa en G: Si $\alpha\beta\gamma \in (V_N \cup V_T)^*$ y $\beta \to \delta \in P$ entonces $\alpha\delta\gamma$ es una derivación directa de G de $\alpha\beta\gamma$ y se denota como $\alpha\beta\gamma \underset{G}{\Longrightarrow} \alpha\delta\gamma$.

- $\overset{+}{\underset{G}{\Longrightarrow}}$ es la clausura positiva.
- $\overset{*}{\underset{G}{\Longrightarrow}}$ es la clausura transitiva y reflexiva.
- $\overset{k}{\underset{G}{\Longrightarrow}}$ será la potencia k-ésima.

Lenguaje de una grámatica $\mathcal{L}(G)$: Es el conjunto de todas las cadenas de símbolos terminales que son formas setenciales de G.

$$\mathcal{L}(G) = \{\alpha \in V_T^* : \ S \overset{+}{\underset{G}{\Longrightarrow}} \alpha\}$$

1.4.1. Clasificación de grámaticas (Chomsky)

Gramáticas regulares (tipo 3): Son aquellas gramáticas que cumplen alguna de las siguientes condiciones:

- Todas sus producciones son de la forma $A \to aB$ ó $A \to a$ ó $A \to \lambda$ donde $A, B \in V_N$ y $a \in V_T$. En este caso se dice que es una gramática lineal a derecha.
- Todas sus producciones son de la forma $A \to Ba$ ó $A \to a$ ó $A \to \lambda$ donde $A, B \in V_N$ y $a \in V_T$. En este caso se dice que es una gramática lineal a izquierda.

Gramáticas libres de contexto (tipo 2): Son aquellas gramáticas en las que cada producción es de la forma $A \to \alpha$ donde $A \in V_N$ y $\alpha \in (V_N \cup V_T)^*$.

De la definición anterior puede inferirse que toda grámatica regular es libre de contexto.

Gramáticas sensibles al contexto (tipo 1): Son aquellas gramáticas en las que cada producción es de la forma $\alpha \to \beta$ donde $\alpha, \beta \in (V_N \cup V_T)^*$ y $|\alpha| \leq |\beta|$. Se puede inferir que toda gramática independiente del contexto que no posea regla borradoraas (es decir, que no posea producciones de la forma $A \to \lambda$) es sensible al contexto.

Gramáticas sin restricciones (tipo 0): Son aquellas gramáticas que no poseen ninguna restricción sobre la forma de sus producciones. El conjunto de las grámaticas tipo 0 es el conjunto de todas las grámaticas y permite generar todos los lenguajes aceptados por una máquina de Turing.

Definición: Un lenguaje generado por una grámatica tipo t es llamado **lenguaje tipo** t.

2. Autómatas finitos

2.1. Autómatas finitos deterministicos (AFD)

Un autómata finito determinista es una 5-tupla $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$ donde:

- Q es un conjunto finito de estados.
- Σ es un conjunto finito de símbolos de entrada.
- $\delta : Q \times \Sigma \to Q$ es una función de transición.
- $q_0 \in Q$ es el estado inicial.
- $F \subseteq Q$ es el conjunto de estados finales.

Función de transición generalizada $\hat{\delta}$: La función de transición δ está definida para que tome como parámetro un único símbolo de Σ. Se puede extender para que tome como parámetro una cadena de símbolos de Σ, es decir $\hat{\delta} : Q \times \Sigma^* \to Q$:

- $\hat{\delta}(q, \lambda) = q$
- $\hat{\delta}(q, \beta a) = \delta(\hat{\delta}(q, \beta), a)$ con $\beta \in \Sigma^*$ y $a \in \Sigma$

Cadena aceptada por un AFD: Una cadena $\beta \in \Sigma^*$ es aceptada por un AFD $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$ si y solo si $\hat{\delta}(q_0, \beta) \in F$.

Lenguaje aceptado por un AFD: El lenguaje aceptado por un AFD $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$ es el conjunto de todas las cadenas $\beta \in \Sigma^*$ que son aceptadas por $\mathcal{M}$:

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

2.2. Autómatas finitos no deterministas (AFND)

Un autómata finito no determinista es una 5-tupla $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$ donde:

- Q es un conjunto finito de estados.
- Σ es un conjunto finito de símbolos de entrada.
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ es una función de transición.

  A diferencia de los AFD, la función δ devuelve un conjunto de estados en lugar de un solo estado.
- $q_0 \in Q$ es el estado inicial.
- $F \subseteq Q$ es el conjunto de estados finales.

Función de transición generalizada $\hat{\delta}$: Primero vamos a definir $\delta_P : \mathcal{P}(Q) \times \Sigma \to \mathcal{P}(Q)$ de la siguiente manera:

$$\delta_P(P, a) = \bigcup_{p \in P} \delta(p, a)$$

La función $\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$ se define de manera recursiva como:

- $\hat{\delta}(q, \lambda) = \{q\}$
- $\hat{\delta}(q, \beta a) = \{p : \ \exists r \in \hat{\delta}(q, \beta) \text{ tal que } p \in \delta(r, a)\} = \delta_P(\hat{\delta}(q, \beta), a)$ con $\beta \in \Sigma^*$ y $a \in \Sigma$

Para generalizar a un más podemos definir $\hat{\delta}_P : \mathcal{P}(Q) \times \Sigma^* \to \mathcal{P}(Q)$ de la siguiente manera:

$$\hat{\delta}_P(P, \beta) = \bigcup_{q \in P} \hat{\delta}(q, \beta)$$

Cadena aceptada por un AFND: Una cadena $\beta \in \Sigma^*$ es aceptada por un AFND $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$ si y solo si $\hat{\delta}(q_0, \beta) \cap F \neq \emptyset$. Es decir, si alguno de los estados alcanzados por $\hat{\delta}(q_0, \beta)$ es un estado final.

Lenguaje aceptado por un AFND: El lenguaje aceptado por un AFND $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$ es el conjunto de todas las cadenas $\beta \in \Sigma^*$ que son aceptadas por $\mathcal{M}$:

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

2.2.1. Equivalencia entre AFD y AFND

Es trivial ver que para todo AFD existe un AFND que acepte el mismo lenguaje.

Teorema 2.1. Dado una AFND $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$, existe un AFD $\mathcal{M}' = \langle Q', \Sigma, \delta', q_0', F' \rangle$ tal que $L(\mathcal{M}) = L(\mathcal{M}')$.

Vamos a demostrar este teorema construyendo una AFD $\mathcal{M}'$ a partir de $\mathcal{M}$. Una vez constuido deberemos demostrar que $\mathcal{M}'$ acepta el mismo lenguaje que $\mathcal{M}$.

Construcción de $\mathcal{M}'$:

- Q' será el conjunto de partes $\mathcal{P}(Q)$. Vamos a denotar cada estado $s \in Q'$ con etiquetas del estilo $[q_1, \ldots, q_k]$ donde $q_1, \ldots, q_k \in Q$. Entonces:

$$Q' = \mathcal{P}(Q)$$

- $\delta'([q_1, \ldots, q_k], a) = [p_1, \ldots, p_m] \iff \delta_P(\{q_1, \ldots, q_k\}, a) = \{p_1, \ldots, p_m\}$
- $q_0' = [q_0]$
- $F' = \{[q_1, \ldots, q_n] \in Q' : \{q_1, \ldots, q_n\} \cap F \neq \emptyset\}$

Equivalencia entre funciones de transición: Antes de demostrar que ambos automátas aceptan el mismo lenguaje, vamos a demostrar que las funciones de transición generalizadas de ambos automátas son equivalentes cuando las llamamos con el estado inicial como primer parámetro. Es decir, queremos ver que $\hat{\delta}'(q_0', \beta) = [p_1, \ldots, p_k] \iff \hat{\delta}(q_0, \beta) = \{p_1, \ldots, p_k\}$.

Lo vamos a hacer por inducción. Recordemos que $q_0' = [q_0]$:

- Caso base: $\beta = \lambda$:
  - $\hat{\delta}'([q_0], \lambda) = [q_0]$ por definición de $\hat{\delta}'$.
  - $\hat{\delta}(q_0, \lambda) = \{q_0\}$ por definición de $\hat{\delta}$.

  Luego $\hat{\delta}'([q_0], \lambda) = [q_0] \iff \hat{\delta}(q_0, \lambda) = \{q_0\}$.

- Caso inductivo: $\beta \implies \beta a$:

  Nuestra hipotesis inductiva es $\hat{\delta}'(q_0', \beta) = [r_1, \ldots, r_m] \iff \hat{\delta}(q_0, \beta) = \{r_1, \ldots, r_m\}$.

  Queremos ver que $\hat{\delta}'(q_0', \beta a) = [p_1, \ldots, p_k] \iff \hat{\delta}(q_0, \beta a) = \{p_1, \ldots, p_k\}$

$$\begin{aligned}
\hat{\delta}'(q_0', \beta a) = [p_1, \ldots, p_k] &\underset{\text{def.}}{\iff} \delta'(\hat{\delta}'(q_0', \beta), a) = [p_1, \ldots, p_k] \\
&\underset{\text{def.}}{\iff} \exists [r_1, \ldots, r_m] \in Q' \text{ tal que } \delta'(q_0', \beta) = [r_1, \ldots, r_m] \\
&\qquad \wedge \delta'([r_1, \ldots, r_m], a) = [p_1, \ldots, p_k] \\
&\underset{\substack{\text{H.I} \\ \text{constr.}}}{\iff} \exists \{r_1, \ldots, r_m\} \in Q \text{ tal que } \hat{\delta}(q_0, \beta) = \{r_1, \ldots, r_m\} \\
&\qquad \wedge \delta_P(\{r_1, \ldots, r_m\}, a) = \{p_1, \ldots, p_k\} \\
&\underset{\text{def.}}{\iff} \delta_P(\hat{\delta}(q_0, \beta), a) = \{p_1, \ldots, p_k\} \underset{\text{def.}}{\iff} \hat{\delta}(q_0, \beta a) = \{p_1, \ldots, p_k\}
\end{aligned}$$

Demostración de la equivalencia: Ahora que hemos demostrado que las funciones de transición generalizadas de ambos automátas son equivalentes, vamos a demostrar que ambos automátas aceptan el mismo lenguaje:

$$\begin{aligned}
\beta \in \mathcal{L}(\mathcal{M}) &\underset{\text{def.}}{\iff} \hat{\delta}(q_0, \beta) = \{q_1, \ldots, q_n\} \wedge \{q_1, \ldots, q_n\} \cap F \neq \emptyset \\
&\underset{\substack{\text{equiv.} \\ \text{constr.}}}{\iff} \hat{\delta}(q_0', \beta) = [q_1, \ldots, q_n] \wedge [q_1, \ldots, q_n] \in F' \\
&\underset{\text{def.}}{\iff} x \in \mathcal{L}(M')
\end{aligned}$$

2.3. Autómatas finitos no deterministico con transiciones λ

Un autómata finito no determinista con transiciones λ es un autómata finito no determinista que tiene transiciones λ. Estas transacciones nos permiten ir de un estado a otro sin consumir ningún símbolo de entrada.

Los definimos con una 5-upla $(Q, \Sigma, \delta, q_0, F)$ donde:

- Q es un conjunto finito de estados.
- Σ es un conjunto finito de símbolos de entrada.
- $\delta : Q \times (\Sigma \cup \{\lambda\}) \to \mathcal{P}(Q)$ es una función de transición.
- $q_0 \in Q$ es el estado inicial.
- $F \subseteq Q$ es el conjunto de estados finales.

Clausura λ de un estado q: Se denota $Cl_\lambda(q)$ es el conjunto de estados que se pueden alcanzar desde q siguiendo solo transiciones λ. Es decir,

$$Cl_\lambda(q) = \delta(q, \lambda)$$

Además $q \in Cl_\lambda(q)$.

Clausura λ de un conjunto de estados P:

$$Cl_{P\lambda}(P) = \bigcup_{p \in P} Cl_\lambda(p)$$

Generalización de la función de transición: Podemos extender δ a conjunto de estados:

$$\delta_P : \mathcal{P}(Q) \times (\Sigma \cup \{\lambda\}) \to \mathcal{P}(Q)$$
$$\delta_P(P, a) = \bigcup_{p \in P} \delta(p, a)$$

Entonces podemos definir:

$$\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$$
$$\hat{\delta}(q_0, \lambda) = Cl_\lambda(q_0)$$
$$\hat{\delta}(q_0, \beta a) = Cl_{P\lambda}\left(\delta_P(\hat{\delta}(q_0, \beta), a)\right) = Cl_{P\lambda}\left(\left\{p : \exists q \in \hat{\delta}(q_0, \beta) \text{ tal que } p \in \delta(q, a)\right\}\right)$$

Tambien extendemos $\hat{\delta}$ a conjuntos de estados:

$$\hat{\delta}_P : \mathcal{P}(Q) \times \Sigma^* \to \mathcal{P}(Q)$$
$$\hat{\delta}_P(P, \beta a) = \bigcup_{p \in P} \hat{\delta}(p, \beta a)$$

Cadena aceptada por un AFND-λ: Una cadena β es aceptada por un AFND-λ $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ si y solo si $\hat{\delta}(q_0, \beta) \cap F \neq \emptyset$.

Lenguaje aceptado por un AFND-λ: El lenguaje aceptado por un AFND-λ $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ es el conjunto de todas las cadenas aceptadas por M:

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

2.3.1. Equivalencia entre AFND y AFND-λ

Dado un AFND-λ $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ podemos construir un AFND $M' = \langle Q, \Sigma, \delta', q_0, F'\rangle$ tal que M acepte el mismo lenguaje que M'.

Construcción de M': Notemos que ambos autómatas tiene el mismo conjunto de estados Q, el mismo conjunto de símbolos de entrada Σ y el mismo estado inicial q_0. Por lo que solo debemos definir δ' y F'.

- $\delta'(q,a) = \hat{\delta}(q,a) = Cl_{P\lambda}\left(\delta_P(\hat{\delta}(q,\lambda),a)\right)$
- $F' = \begin{cases} F \cup \{q_0\} & \text{si } Cl_\lambda(q_0) \cap F \neq \emptyset \\ F & \text{si no} \end{cases}$

Equivalencia de funciones de transición generalizada: Vamos a demostrar por inducción que $\hat{\delta}'(q_0,\beta) = \hat{\delta}(q_0,\beta)$ para todo $|\beta| \geq 1$:

- Caso base: $|\beta| = 1$. Sea $\beta = a$, entonces $\hat{\delta}'(q_0,\beta) = \hat{\delta}'(q_0,a) = \hat{\delta}(q_0,a)$ por como definimos δ'.
- Caso inductivo: Supongamos que $\hat{\delta}'(q_0,\beta) = \hat{\delta}(q_0,\beta)$ para todo $|\beta| \leq n$. Sea $\omega = \beta a$. Entonces:

$$\hat{\delta}'(q_0,\omega) = \hat{\delta}'(q_0,\beta a) \underset{\text{def.}}{=} \delta'_P(\hat{\delta}'(q_0,\beta),a) \underset{\text{H.I}}{=} \delta'_P(\hat{\delta}(q_0,\beta),a) \tag{1}$$

  Por otro lado, dado $P \subseteq Q$ tenemos que:

$$\delta'_P(P,a) \underset{\text{def.}}{=} \bigcup_{p\in P} \delta'(p,a) \underset{\text{constr.}}{=} \bigcup_{p\in P} \hat{\delta}(p,a) \underset{\text{def}}{=} \hat{\delta}_P(P,a)$$

  Entonces, remplazando el último término en (1), nos queda:

$$\delta'_P(\hat{\delta}(q_0,\beta),a) = \hat{\delta}_P(\hat{\delta}(q_0,\beta),a) \underset{\text{def.}}{=} \hat{\delta}(q_0,\beta a) \underset{\text{def.}}{=} \hat{\delta}(q_0,\omega)$$

Demostración de equivalencia: Veamos ahora que M acepta el mismo lenguaje que M', vamos a separar la demostración en dos casos: $\beta = \lambda$ y $\beta \neq \lambda$.

- $\beta = \lambda$
  - $\lambda \in \mathcal{L}(M) \implies \lambda \in \mathcal{L}(M')$

$$\begin{aligned} \lambda \in \mathcal{L}(M) &\underset{\text{def.}}{\iff} \hat{\delta}(q_0,\lambda) \cap F \neq \emptyset \\ &\underset{\text{def.}}{\iff} Cl_\lambda(q_0) \cap F \neq \emptyset \\ &\underset{\text{constr.}}{\implies} q_0 \in F' \underset{\text{def.}}{\iff} \lambda \in \mathcal{L}(M') \end{aligned}$$

- $\lambda \in \mathcal{L}(M') \implies \lambda \in \mathcal{L}(M)$.

$$
\begin{aligned}
\lambda \in \mathcal{L}(M') &\underset{\text{def.}}{\implies} q_0 \in F' \\
&\underset{\text{constr.}}{\implies} q_0 \in F \vee Cl_\lambda(q_0) \cap F \neq \emptyset \\
&\underset{\text{def. F}}{\implies} Cl_\lambda(q_0) \cap F \neq \emptyset \vee Cl_\lambda(q_0) \cap F \neq \emptyset \\
&\underset{\text{def.}}{\implies} Cl_\lambda(q_0) \cap F \neq \emptyset \\
&\underset{\text{def.}}{\iff} \lambda \in \mathcal{L}(M)
\end{aligned}
$$

- $\beta \neq \lambda$

  - $\beta \in \mathcal{L}(M) \implies \beta \in \mathcal{L}(M')$

$$
\begin{aligned}
\beta \in \mathcal{L}(M) &\underset{\text{def.}}{\implies} \hat{\delta}(q_0, \beta) \cap F \neq \emptyset \\
&\underset{\text{equiv. tran.}}{\implies} \hat{\delta}'(q_0, \beta) \cap F \neq \emptyset \\
&\underset{\text{constr. M'}}{\implies} \hat{\delta}'(q_0, \beta) \cap F' \neq \emptyset \underset{\text{def.}}{\implies} \beta \in \mathcal{L}(M')
\end{aligned}
$$

  - $\beta \in \mathcal{L}(M') \implies \beta \in \mathcal{L}(M)$

$$
\begin{aligned}
\beta \in \mathcal{L}(M') &\underset{\text{def.}}{\implies} \hat{\delta}'(q_0, \beta) \cap F' \neq \emptyset \\
&\underset{\text{equiv. trans}}{\implies} \hat{\delta}(q_0, \beta) \cap F \neq \emptyset \vee \hat{\delta}(q_0, \beta) \cap (F \cup \{q_0\}) \neq \emptyset \hat{\delta}(q_0, \beta) \cap \\
&\underset{\text{constr.}M'}{\implies} \hat{\delta}(q_0, \beta) \cap F \neq \emptyset
\end{aligned}
$$

  Si vale la primera parte de la última expresión $\delta(q_0, \beta) \cap F \neq \emptyset$ entonces $\beta \in \mathcal{L}(M)$ por definición.
  Veamos que pasa si vale $\hat{\delta}(q_0, \beta) \cap (F \cup \{q_0\}) \neq \emptyset$, es decir hay un camino de transiciones λ desde q_0 hasta algún estado estado $q' \in F$:

$$\hat{\delta}(q_0, \beta) \cap (F \cup \{q_0\}) \neq \emptyset \implies \hat{\delta}(q_0, \beta) \cap F \neq \emptyset \vee \hat{\delta}(q_0, \beta) \cap \{q_0\} \neq \emptyset$$

  La primer parte es lo mismo que arriba, analizemos la segunda:

$$\hat{\delta}(q_0, \beta) \cap \{q_0\} \neq \emptyset \implies Cl_\lambda(q_0) \cap F \neq \emptyset \implies \lambda \in \mathcal{L}(M)$$

Queda demostrada la equivalencia de lenguajes.

3. Expresiones regulares

Una expresión regular es una expresión que describe un lenguaje de forma compacta y sencilla:

- $\varnothing$ es una expresión regular que describe el lenguaje vacío $\emptyset$.
- λ es una expresión regular que describe el lenguaje unitario $\{\lambda\}$.
- Para cada $a \in \sigma$, a es una expresión regular que describe el lenguaje $\{a\}$.
- Si r y s son dos expresiones que denotan los lenguajes R y S entonces:
    - $r|s$ ó $r+s$ es una expresión regular que describe el lenguaje $R \cup S$.
    - rs es una expresión regular que describe el lenguaje RS.
    - r^* es una expresión regular que describe el lenguaje R^*.
    - r^+ es una expresión regular que describe el lenguaje R^+.

Expresiones regulares recursivas: Si $r = \alpha r + \beta$, entonces $r = \alpha^* \beta$. Además, si $\alpha^* = \alpha^+$, entonces $r = \alpha^*(\beta + \gamma)$ para culquier expresión regular γ.

3.1. Expresiones regulares a AFND-λ

Dada una expresión regular r, existe una AFND-λ M con un solo estado final y sin trancisiones a partir del mismo tal que $\mathcal{L}(M) = \mathcal{L}(r)$.

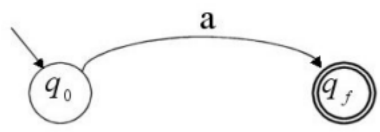
Vamosa a demostrar por inducción sobre los operadores de las expresiones regulares.

3.1.1. Casos base

$r = \varnothing$

$r = \lambda$

$$r = a$$



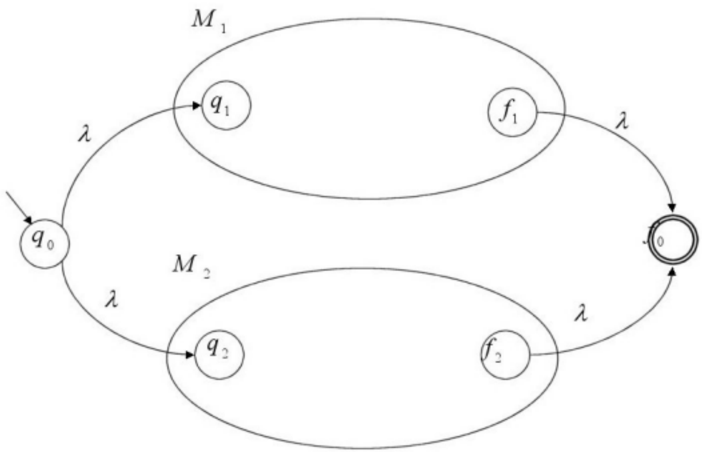
3.1.2. Pasos inductivos

Sean r_1 y r_2 dos expresiones regulares. Supongamos que existen AFND-λ $M_1 = \langle Q_1, \Sigma_1, \delta_1, q_1, \{f_1\}\rangle$ y $M_2 = \langle Q_2, \Sigma_2, \delta_2, q_2, \{f_{\}}\rangle$ tal que $\mathcal{L}(M_1) = \mathcal{L}(r_1)$ y $\mathcal{L}(M_2) = \mathcal{L}(r_2)$. Vamos a armar a partir de estos autómata uno nuevo que acepte los lenguajes generados por las expresiones $r_1|r_2$, r_1r_2, r_1^* y r^+.

$\boldsymbol{r_1|r_2}$**:** Podemos construir un automata $M_0 = \langle Q_0, \Sigma_0, \delta_0, q_0, \{f_0\}\rangle$ tal que $\mathcal{L}(M_0) = \mathcal{L}(r_1|r_2)$ de la siguiente forma:

- $Q_0 = Q_1 \cup Q_2 \cup \{q_0, f_0\}$
- $\Sigma_0 = \Sigma_1 \cup \Sigma_2$
- $\delta_0 : Q_0 \times \Sigma_0 \to \mathcal{P}(Q_0)$

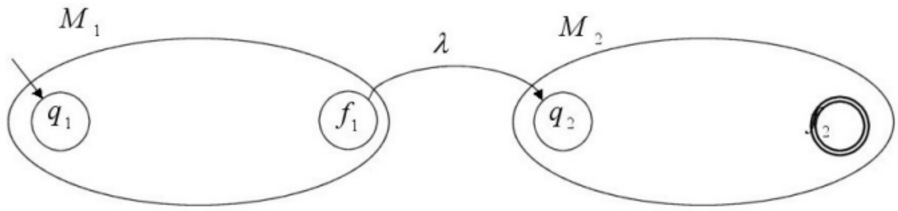
$$\delta(q_0, \lambda) = \{q_1, q_2\}$$
$$\delta(q, a) = \delta_1(q, a) \text{ para } q \in Q_1 - \{f_1\} \text{ y } a \in \Sigma_1 \cup \{\lambda\}$$
$$\delta(q, a) = \delta_2(q, a) \text{ para } q \in Q_2 - \{f_2\} \text{ y } a \in \Sigma_2 \cup \{\lambda\}$$
$$d(f_1, \lambda) = d(f_2, \lambda) = \{f_0\}$$



$\boldsymbol{r_1 r_2}$**:** $M_0 = \langle Q_0, \Sigma_0, \delta_0, q_1, \{f_2\} \rangle$:

- $Q_0 = Q_1 \cup Q_2$
- $\Sigma_0 = \Sigma_1 \cup \Sigma_2$
- $\delta_0 : Q_0 \times \Sigma_0 \to \mathcal{P}(Q_0)$

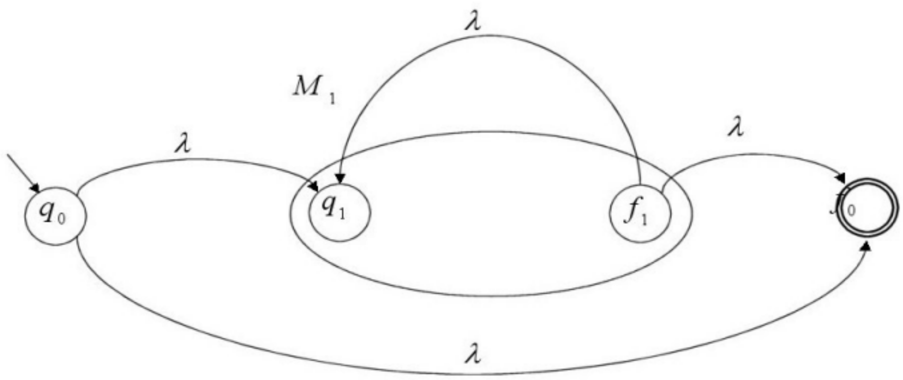
$$\delta(q, a) = \delta_1(q, a) \text{ para } q \in Q_1 - \{f_1\} \text{ y } a \in \Sigma_1 \cup \{\lambda\}$$
$$\delta(q, a) = \delta_2(q, a) \text{ para } q \in Q_2 - \{f_2\} \text{ y } a \in \Sigma_2 \cup \{\lambda\}$$
$$d(f_1, \lambda) = \{q_2\}$$



$\boldsymbol{r_1^*}$**:** $M_0 = \langle Q_0, \Sigma_1, \delta_0, q_0, \{f_0\} \rangle$:

- $Q_0 = Q_1 \cup \{f_0, q_0\}$
- $\delta_1 : Q_0 \times \Sigma_1 \to \mathcal{P}(Q_0)$

$$\delta(q_0, \lambda) = \delta(f_1, \lambda) = \{q_1, f_0\}$$
$$\delta(q, a) = \delta_1(q, a) \text{ para } q \in Q_1 - \{f_1\} \text{ y } a \in \Sigma_1 \cup \{\lambda\}$$



Para el caso r_1^+ es el mismo autómata que para este caso sin la transición $q_0 \xrightarrow{\lambda} f_0$.

3.2. AFD a expresión regular

Dado un AFD $M = \langle \{q_1, \dots, q_n\}, \Sigma, \delta, q_1, F\rangle$, que acepta el lenguaje $\mathcal{L}$, existe una expresión regular que denota el mismo lenguaje.

3.2.1. Demostración

Nombremos $R^k_{i,j}$ a la expresión regular cuyo lenguaje $\omega \subseteq \Sigma^*$ son las cadenas que llevan al autómata M desde el estado q_i al estado q_j pasando solo por estados q_l con $l \leq k$. En particular $R^n_{i,j}$ es la expresión regular que representa todas las cadenas que permiten ir del estado i al estado j.

Vamos a buscar como construir $R^k_{i,j}$ para cada $k \in \{0, \dots, n\}$ de manera inductiva. Suponiendo que demostramos la existencia de esta expresión regular, podemos concluir que la unión $R^n_{1,f_1}|R^n_{1,f_2}|...|R^n_{1,f_m}$ (con $f_1 \dots f_m \in F$) es la expresión regular que representa el lenguaje $\mathcal{L}$:

Caso base ($k = 0$): Como todos los estados están enumerados del 1 para arriba, $k = 0$ significa que no debe haber estados intermedios en el camino entre q_i y q_j, por lo que pueden ser de dos formas:

- Una arco del estado i al estado j.
- Un camino de longitud cero que solo contiene el estado i.

Si $i \neq j$, entonces solo es posible la primera opción. Debemos examinar el AFD y encontrar aquellos simbolos que nos permitan ir del estado i al estado j.

1. Si no existe tal símbolo, entonces $R^0_{i,j} = \varnothing$.
2. Si existe exactamente un símbolo a, entonces $R^0_{i,j} = a$.
3. Si existen más de un símbolo, entonces $R^0_{i,j} = a_1|a_2|...|a_n$.

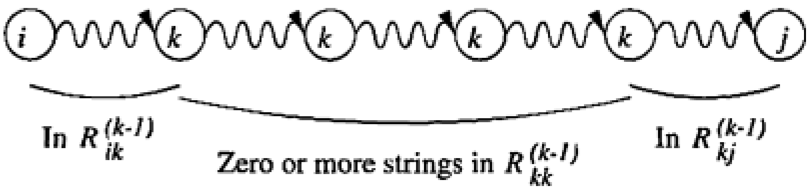
Ahora, si $i = j$ entonces los caminos de longitud cero también son posibles, por lo que habría que agregar a cada una de las expresiones recién mencionadas el simbolo λ:

1. $R^0_{i,j} = \lambda$.
2. $R^0_{i,j} = a|\lambda$.
3. $R^0_{i,j} = a_1|a_2|...|a_n|\lambda$.

Paso inductivo: Supongamos que hay un camino desde el estado i al estado j que no pasa por estados mas grandes k. Entonces podemos considerar las siguientes dos opciones:

1. El camino no pasa por el estado k, por lo que el lenguaje de $R^{k-1}_{i,j}$ contiene a ese camino.

2. El camino pasa por el estado k por lo menos una vez. Entonces podemos partir el camino en varias partes:



La primer parte, va desde el estado i al estado k sin pasar por k, la última parte es desde el estado k al estado j sin pasar por k, y todas las partes intermedia s van desde el estado k al estado k sin pasar por k. Cada una de estas partes ya tiene una expresión regular asociada: $R_{i,k}^{k-1}$, $R_{k,k}^{k-1}$, $R_{k,j}^{k-1}$, por lo que podemos unirlas para obtener la expresión regular que representa el camino completo de la siguiente forma:

$$R_{i,k}^{k-1}\left(R_{k,k}^{k-1}\right)^{*}R_{k,j}^{k-1}$$

Entonces $R_{i,j}^{k}$ es la unión de las expresiones de los dos tipos de caminos que acabamos de describir:

$$R_{i,j}^{k}=R_{i,j}^{k-1}\cup R_{i,k}^{k-1}\left(R_{k,k}^{k-1}\right)^{*}R_{k,j}^{k-1}$$

Finalmente, si construimos en orden todas estas expresiones regulares desde $R_{i,j}^{0}$, eventualmente llegaremos hasta $R_{i,j}^{n}$.

Y como dijimos, más arriba si calculamos $R_{1,j}^{0}$ para cada $q_j\in F$ y unimos todas las expresiones, obtendremos la expresión regular que representa el lenguaje $\mathcal{L}$.

3.3. Gramática regular a AFND

Dada una grámatica regular $G = \langle V_N, V_T, P, S\rangle$, podemos construir un AFND $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ que reconozca el lenguaje generado por G

3.3.1. Demostración

Vamos a constuir el autómata finito no determinista M y demostrar que reconoce el lenguaje generado por G.

Construcción de M: Construyamos M de la siguiente manera:

- $Q = V_N \cup \{q_f\}$. Denotarmeos q_A al estado que representa al no símbolo no terminar A.
- $\Sigma = V_T$
- $q_0 = q_S$
- Si $A, B \in V_N$ y $a \in \Sigma$, entonces:
  - $q_B \in \delta(q_A, a) \iff A \to aB \in P$
  - $q_f \in \delta(q_A, a) \iff A \to a \in P$
  - $q_A \in F \iff A \to \lambda \in P$
  - $q_f \in F$

Equivalencia clausura transitiva de producciones y δ: Vamos a probar por inducción que

$$A \overset{*}{\Rightarrow} \alpha B \iff q_B \in \hat{\delta}(q_A, \alpha)$$

- **Caso base $\alpha = \lambda$:**
  - $A \overset{*}{\Rightarrow} \alpha B$, pero las gramáticas regulares no acentan producciones que vayan de un no terminal a otro sin pasar por un terminal, por lo que $B = A$. Osea $A \overset{*}{\Rightarrow} \alpha A$.
  - Además, como es un AFND, no tiene transiciones lambda, osea que $\delta(q_A, \lambda) = \{q_A\}$, por lo que $q_A \in \hat{\delta}(q_A, \alpha)$.
- **Caso inductivo $\alpha = \beta a$:**

$$A \overset{*}{\Rightarrow} \alpha B \iff A \overset{*}{\Rightarrow} \beta a B \underset{\text{def.}}{\iff} \exists C \in V_N : A \overset{*}{\Rightarrow} \beta C \wedge C \to aB$$

$$\underset{\substack{\text{H.I} \\ \text{constr. M}}}{\iff} \exists q_C \in Q, q_c \in \hat{\delta}(q_A, \alpha) \wedge q_B \in \delta(q_C, a)$$

$$\iff q_B \in \delta(\hat{\delta}(q_A, \alpha), a)$$

$$\iff q_B \in \hat{\delta}(q_A, \beta a) \iff q_B \in \hat{\delta}(q_A, \alpha)$$

Demostración de la equivalencia: Vamos a demostrar que el lenguaje generado por G y M son iguales, osea que $\alpha a \in \mathcal{L}(M) \iff S \overset{*}{\Rightarrow} \alpha a$. Como G es una grámatica regular, hay solo dos formas de llegar desde S hasta αa:

1. $\exists A \in V_N : S \overset{*}{\Rightarrow} \alpha A \wedge A \to a \in P$
2. $\exists B \in V_N : S \overset{*}{\Rightarrow} \alpha a B \wedge B \to \lambda \in P$

Entonces:

$$\begin{aligned}
S \overset{*}{\Rightarrow} \alpha a &\underset{\text{def. G}}{\iff} (\exists A \in V_N : S \overset{*}{\Rightarrow} \alpha A \wedge A \to a \in P) \vee (\exists B \in V_N : S \overset{*}{\Rightarrow} \alpha a B \wedge B \to \lambda \in P) \\
&\underset{\text{Equiv. anterior}}{\iff} (\exists q_A \in Q, q_A \in \hat{\delta}(q_0, \alpha) \wedge q_f \in \delta(q_A, a)) \vee (\exists q_B \in Q, q_B \in \hat{\delta}(q_0, \alpha a) \wedge q_B \in F) \\
&\underset{\text{def. } \delta}{\iff} q_f \in \delta(q_S, \alpha a) \vee (\exists q_B \in Q, q_B \in \hat{\delta}(q_0, \alpha a) \wedge q_B \in F) \\
&\iff \alpha a \in \mathcal{L}(M)
\end{aligned}$$

Falta ver que pasa si $\lambda \in \mathcal{L}(G)$:

$$\lambda \in \mathcal{L}(G) \iff S \overset{*}{\Rightarrow} \lambda \iff S \to \lambda \in P \iff q_S \in F \iff \lambda \in \mathcal{L}(M)$$

3.4. AFD a gramática regular

Dado un AFD $M = \langle Q, \Sigma, \delta, q_0, F \rangle$, existe una gramática regular $G = \langle V_N, V_T, P, S \rangle$ equivalente

3.4.1. Demostración

Contrucción de G: Vamos a construir G de la siguiente forma:

- $V_N = Q$, para mayor claridad llamamos A_p al no terminal correspondiente al estado $p \in Q$
- $V_T = \Sigma$
- $S = q_0$
- Si $q \in Q \wedge q \notin F$ entonces $A_p \to a A_q \in P \iff \delta(p, a) = q$
- Si $q \in F$ entonces $A_p \to a \in P \iff \delta(p, a) = q$
- $S \to \lambda \in P \iff q_0 \in F$

Paso intermedio: Vamos a demostrar por inducción:

$$\hat{\delta}(p,\alpha) = q \iff A_p \overset{*}{\Rightarrow} \alpha A_q$$

- **Caso base:** $\alpha = \lambda$ es trivial:

$$\hat{\delta}(p,\lambda) = q \iff A_p \overset{*}{\Rightarrow} A_p$$

- **Caso inductivo** $\alpha = \beta a$**:** Queremos probar que $\hat{\delta}(p,\alpha) = q \iff A_p \overset{*}{\Rightarrow} \alpha A_q$.

  Nuestra hipotesis inductiva: $\hat{\delta}(p,\beta) = q \iff A_p \overset{*}{\Rightarrow} \beta A_q$ para todo $|\beta| \leq n$

$$\hat{\delta}(p,\alpha) = \hat{\delta}(p,\beta a) = q \underset{\text{def.}}{\iff} \exists r \in Q : \ \hat{\delta}(p,\beta) = r \wedge \delta(r,a) = q$$

$$\underset{\substack{\text{H.I} \\ \text{constr. G}}}{\iff} \exists A_r, A_p \overset{*}{\Rightarrow} \beta A_r \wedge A_r \rightarrow a A_q \in P \iff A_p \overset{*}{\Rightarrow} \beta a A_q$$

Demostración de equivalencia de lenguajes:

$$\alpha a \in \mathcal{L}(M) \underset{\text{def.}}{\iff} \hat{\delta}(q_0,\alpha a) \in F \underset{\text{def.}}{\iff} \exists q \in Q : \ \hat{\delta}(q_0,\alpha) = q \wedge \delta(q,a) \in F$$

$$\underset{\text{paso intermedio}}{\iff} \exists A_p, A_{q0} \overset{*}{\Rightarrow} \alpha A_p \wedge A_p \rightarrow a \in P \iff A_{q0} \overset{*}{\Rightarrow} \alpha a$$

$$\underset{\text{def.}}{\iff} \alpha a \in \mathcal{L}(G)$$

4. Minimización de AFD

4.1. Indistinguibilidad

Sea $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ un AFD, decimos que $p, q \in Q$, son indistinguibles ($p \equiv q$) si para toda cadena $\alpha \in \Sigma^*$ tal que $\hat{\delta}(p, \alpha) \in F$ entonces pasa que $\hat{\delta}(q, \alpha) \in F$ y viceversa. Si $p, q \in Q$ son indistinguibles, entonces decimos que p y q son equivalentes.

$$p \equiv q \iff \forall \alpha \in \Sigma^* : \ (\hat{\delta}(p, \alpha) \in F \iff \hat{\delta}(q, \alpha) \in F)$$

Teorema: Si p y q son indistinguibles, sea $\alpha \in \Sigma^*$ entonces $\hat{\delta}(p, \alpha) \equiv \hat{\delta}(q, \alpha)$

$$p \equiv q \implies \forall \alpha \in \Sigma^* : \ \hat{\delta}(p, \alpha) \equiv \hat{\delta}(q, \alpha)$$

DEMOSTRACIÓN

Sean $p, q \in Q$, $p \equiv q$.
Supogamos que existe $\alpha \in \Sigma^*$ tal que $\hat{\delta}(p, \alpha) \not\equiv \hat{\delta}(q, \alpha)$ entonces existe una cadena $\gamma \in \Sigma^*$ que distingue a $\hat{\delta}(p, \alpha)$ de $\hat{\delta}(q, \alpha)$. Osea que $\hat{\delta}(\hat{\delta}(p, \alpha), \gamma) \in F$ y $\hat{\delta}(\hat{\delta}(q, \alpha), \gamma) \notin F$ (o viceversa).
Por def: $\hat{\delta}(\hat{\delta}(p, \alpha), \gamma) = \hat{\delta}(p, \alpha\gamma)$ y $\hat{\delta}(\hat{\delta}(q, \alpha), \gamma) = \hat{\delta}(p, \alpha\gamma)$. Entonces, como $\alpha\gamma$ es una cadena que nos permite distinguir p de q, es decir $p \not\equiv q$. Absurdo.

Teorema: $\equiv$ es una relación de equivalencia.

DEMOSTRACIÓN

- **Reflexividad:** $p \equiv p$:

$$\forall \alpha \in \Sigma^* : \ (\hat{\delta}(p, \alpha) \in F \iff \hat{\delta}(p, \alpha) \in F) \iff p \equiv p$$

- **Simetría:** $p \equiv q \implies q \equiv p$:

$$\begin{aligned} p \equiv q &\implies \forall \alpha \in \Sigma^* : \ (\hat{\delta}(p, \alpha) \in F \iff \hat{\delta}(q, \alpha) \in F) \\ &\iff \forall \alpha \in \Sigma^* : \ (\hat{\delta}(q, \alpha) \in F \iff \hat{\delta}(p, \alpha) \in F) \iff q \equiv p \end{aligned}$$

- **Transitividad:** $p \equiv q \land q \equiv r \implies p \equiv r$:

$$p \equiv q \implies \forall \alpha \in \Sigma^* : \ (\hat{\delta}(p, \alpha) \in F \iff \hat{\delta}(q, \alpha) \in F)$$
$$q \equiv r \implies \forall \alpha \in \Sigma^* : \ (\hat{\delta}(q, \alpha) \in F \iff \hat{\delta}(r, \alpha) \in F)$$

Entonces

$$\forall \alpha \in \Sigma^* : \ (\hat{\delta}(p, \alpha) \in F \iff \hat{\delta}(r, \alpha) \in F) \iff p \equiv r$$

4.1.1. Indestinguibilidad de orden k

$$p \overset{k}{\equiv} q \iff \forall \alpha \in \Sigma^*, (|\alpha| \leq k) \implies (\hat{\delta}(p,\alpha) \in F \iff \hat{\delta}(q,\alpha) \in F)$$

Propiedades:

1. $\overset{k}{\equiv}$ es un relación de equivalencia.

   DEMOSTRACIÓN

   Es exactamente igual a la demostración $\equiv$ es transitiva.

2. $\overset{k+1}{\equiv} \subseteq \overset{k}{\equiv}$

   DEMOSTRACIÓN

   $$p \overset{k+1}{\equiv} q \implies \forall \alpha \in \Sigma^*, (|\alpha| \leq k+1) \implies (\hat{\delta}(p,\alpha) \in F \iff \hat{\delta}(q,\alpha) \in F)$$

   Ahora como esto vale $\forall \alpha \in \Sigma^*, (|\alpha| \leq k+1)$, necesarimente vale $\forall \alpha \in \Sigma^*, (|\alpha| \leq k)$. Entonces

   $$\forall \alpha \in \Sigma^*, (|\alpha| \leq k+1) \implies (\hat{\delta}(p,\alpha) \in F \iff \hat{\delta}(q,\alpha) \in F) \implies p \overset{k}{\equiv} q$$

3. $\left(Q/ \overset{0}{\equiv}\right) = \{Q-F, F\}$ si $Q-F \neq \emptyset$ y $F \neq \emptyset$. En castellano, $\overset{0}{\equiv}$ divide al conjunto de estados en estados finales y no finales.

4. $p \overset{k+1}{\equiv} q \iff \left(p \overset{0}{\equiv} q\right) \wedge \left(\forall a \in \Sigma, \delta(p,a) \overset{k}{\equiv} \delta(q,a)\right)$

   DEMOSTRACIÓN

   $\Rightarrow$) Como $\overset{k+1}{\equiv} \subseteq \overset{k}{\equiv}$ entonces $p \overset{k+1}{\equiv} q \implies p \overset{0}{\equiv} q$.

   Por otro lado, supongamos que no vale $\left(\forall a \in \Sigma, \delta(p,a) \overset{k}{\equiv} \delta(q,a)\right)$ entonces

   $$\exists a \in \Sigma,\ \exists \alpha \in \Sigma^*, (|\alpha| \leq k) \wedge \hat{\delta}(\delta(p,a),\alpha) \in F \wedge \hat{\delta}(\delta(q,a),\alpha) \notin F$$

   o viceversa. Pero entonces $p \overset{k+1}{\not\equiv} q$ ya que $a\alpha \leq k+1$ y $a\alpha$ distinque a p y a q.

$\Leftarrow$) Supogamos que $p \overset{k}{\equiv} q$. Entonces ó $p \overset{0}{\equiv} q$ ó $\exists a\alpha,\ |a\alpha| \leq k+1$ que distingue p de q, o sea que:

$$\hat{\delta}(\delta(p,a),\alpha) \in F \wedge \hat{\delta}(\delta(q,a),\alpha) \notin F$$

o viceversa. Pero entonce $\delta(p,a) \overset{k+1}{\not\equiv} \delta(q,a)$.

5. $\left(\overset{k+1}{\equiv} = \overset{k}{\equiv}\right) \implies \forall n \geq 0, \left(\overset{k+n}{\equiv} = \overset{k}{\equiv}\right)$

DEMOSTRACIÓN

Lo vamos a demostrar por inducción:

Caso base:$n = 0$. Entonces $k \overset{k}{\equiv} = \overset{k}{\equiv}$.

Paso inductivo: Suponemos que es cierto para n, osea que vale $\overset{k+1}{\equiv} = \overset{k}{\equiv} \implies \overset{k+n}{\equiv} = \overset{k}{\equiv}$

Queremos probar $\overset{k+1}{\equiv} = \overset{k}{\equiv} \implies \overset{k+n+1}{\equiv} = \overset{k}{\equiv}$:

Sabemos que $\overset{k+n+1}{\equiv} = \overset{k}{\equiv}$ si y solo si $\forall p, q \in Q, \left(p \overset{k}{\equiv} q \iff p \overset{k+n+1}{\equiv} q\right)$

Por la propiedad (4), tenemos:

$$\begin{aligned}
p \overset{k+n+1}{\equiv} &\iff \left(p \overset{0}{\equiv} q\right) \wedge \left(\forall a \in \Sigma, \delta(p,a) \overset{k+n}{\equiv} \delta(q,a)\right) \left(p \overset{0}{\equiv} q\right) \\
&\wedge \left(\forall a \in \Sigma, \delta(p,a) \overset{k+n}{\equiv} \delta(q,a)\right) \\
&\underset{\text{prop. 2}}{\implies} \left(p \overset{0}{\equiv} q\right) \wedge \left(\forall a \in \Sigma, \delta(p,a) \overset{k}{\equiv} \delta(q,a)\right) \\
&\underset{\text{prop. 4}}{\implies} p \overset{k+1}{\equiv} q \underset{\text{prop. 2}}{\implies} q \overset{k}{p}
\end{aligned}$$

4.2. Autómatas finito determinístico mínimo

Sea $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ un AFD sin estados inaccesibles, el AFD mínimo equivalente $M' = \langle Q', \Sigma, \delta', q_0, F'\rangle$ se define de la siguiente manera:

- $Q' = Q/\equiv$. Vamos a notar $[q]$ al estado que representa a la clase de equivalencia que contiene a q.
- $\delta'([q], a) = [\delta(q,a)]$
- $q_0' = [q_0]$
- $F' = \{[q] \in Q' \ :\ q \in F\}$

Teorema:

$$\forall \alpha \in \Sigma^*, \hat{\delta}(q,\alpha) = r \implies \hat{\delta}'(q_0',\alpha) = \hat{\delta}'([q],\alpha) = [r]$$

DEMOSTRACIÓN

Va a ser por inducción en la longitud de α:

- $\alpha = \lambda$: $\hat{\delta}(q,\epsilon) = q$, por def. de $\hat{\delta}$

  $\hat{\delta}'(q_0',\epsilon) = \hat{\delta}'([q],\epsilon) = [q]$ por def. de $\hat{\delta}'$

  Entonces $\hat{\delta}(q,\lambda) = q \implies \hat{\delta}'([q],\lambda) = [q]$

- Paso inductivo: Sea $\alpha = \beta a$, queremos probar que $\hat{\delta}(q,\alpha) = r \implies \hat{\delta}'([q],\alpha) = [r]$.

  Nuestra hipotesis inductiva es $\forall \beta \in \Sigma^*,\ |\beta| \leq n,\ \hat{\delta}(q,\beta) = p \implies \hat{\delta}'([q],\beta) = [p]$

  Entonces:

  $$\hat{\delta}(q,\alpha) = \hat{\delta}(q,\beta a) = \delta(\hat{\delta}(q,\beta),a) = \hat{\delta}(p,a) = r \underbrace{\implies}_{\text{constr. } \delta'} \delta'([p],a) = [r] \ (1)$$

  Además, por hipotesis inductiva sabemos que $\hat{\delta}'([q],\beta) = [p]$, entonces remplazando en el último término de la ecuación (1) obtenemos:

  $$\delta'([p],\alpha) = \delta'(\hat{\delta}'([q],\beta),a) = \hat{\delta}(q,\beta a) = \hat{\delta}(q,\alpha) = [r]$$

4.3. Algoritmo de minimización de un AFD

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

$P \leftarrow \{Q-F,F\}$ ▷ $P = \overset{0}{\equiv}$, separamos los estados finales de los no finales
$stop \leftarrow false$
while $stop = false$ **do**
  $P' \leftarrow \emptyset$
  for $X \in P$ **do** ▷ Separamos cada clase de equivalencia en las subclases de $\overset{n+1}{\equiv}$
    while $\exists e \in X : \neg\texttt{marked}(e,X)$ **do** ▷ Elegimos un nuevo representante para cada clase
      $X_1 \leftarrow \{e\}$
      $\texttt{marked}(e,X)$
      for $e' \in X : e \neq e'$ **do** ▷ Conseguimos los elementos de esa clase
        if $\neg\texttt{marked}(e',X) \wedge (\forall a \in \Sigma,\ [\delta(e,a)] = [\delta(e',a)])$ **then**
          $X_1 \leftarrow X_1 \cup \{e'\}$
          $\texttt{mark}(e',X)$

```
            end if
        end for
        P' ← P' ∪ {X_1}
    end while
end for
if P ≠ P' then
    P ← P'
else
    stop ← true
end if
end while
```

Lema: Sean $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ y $M' = \langle Q', \Sigma, \delta', q_0', F' \rangle$ dos AFDs. Si M no poseee estados inaccesibles y todo par de cadenas que conducen a estados diferentes de M conducen a estados diferentes de M', entonces la cantidad de estados de M' es mayor o igual a la cantidad de estados de M. Es decir:

$$\left(\forall \alpha, \beta \in \Sigma^*,\ \hat{\delta}(q, \alpha) \neq \hat{\delta}(q, \beta) \implies \hat{\delta}'(q_0', \alpha) \neq \hat{\delta}'(q_0', \beta)\right) \implies |Q| \leq |Q'|$$

DEMOSTRACIÓN

Sea $g : Q \to \Sigma^*$ definida por $g(q) = \min\left\{\alpha \in \Sigma^* :\ \hat{\delta}(q_0, \alpha) = q\right\}$ donde suponemos una relación de orden en Σ^* dada por la longitud para cadenas de distinta longitud, y por el lexicográfico para las cadenas de igual longitud. Definamos $f : Q \to Q'$ con $f(q) = \hat{\delta}'(q_0', g(q))$.
Como para cualquier par de estados diferentes $p, q \in Q$ es cierto que $\hat{\delta}(q_0, g(p)) \neq \hat{\delta}(q_0, g(q))$, entonces $\hat{\delta}'(q_0', g(p)) \neq \hat{\delta}'(q_0', g(q))$. Lo que equivale a decir que $f(p) \neq f(q)$. Por lo tanto, f es una función inyectiva, es decir que $|Q| \leq |Q'|$.

Lema: Sea $M_R = \langle Q_R, \Sigma, \delta_R, q_{R0}, F_R \rangle$ el autómata reducido correspondiente a $M = \langle Q, \Sigma, \delta, q_0, F \rangle$. Entonces, cualquier autómata $M' = \langle Q', \Sigma, \delta', q_{0'}, F' \rangle$ que reconozca el mismo lenguaje que M no poseerá menos estados que M_R. Osea:

$$\forall M',\ \mathcal{L}(M') = \mathcal{L}(M) \implies |Q'| \geq |Q_R|$$

DEMOSTRACIÓN

Supongamos que $\exists M'$ tal que $|Q'| < |Q_R|$, entonces según el lema anterior deben existir dos cadenas $\alpha, \beta \in \Sigma^*$ tales que

$$\hat{\delta_R}(q_0, \alpha) \neq \hat{\delta_R}(q_0, \beta) \wedge \hat{\delta}'(q_0', \alpha) = \hat{\delta}'(q_0', \beta)$$

Pero entonces, como $\hat{\delta_R}(q_0, \alpha)$ y $\hat{\delta_R}(q_0, \beta)$ son estados diferentes, entonces $\hat{\delta}(q_0, \alpha)$ y $\hat{\delta}(q_0, \beta)$ son estados distinguibles por pertenecer al autómata reducido M_R entonces $\exists \gamma \in \Sigma^*$ tal que:

$$\hat{\delta}(q_0, \alpha\gamma) \in F \wedge \hat{\delta}(q_0, \beta\gamma) \notin F$$

o viceversa. Entonces $\alpha\gamma \in \mathcal{L}(M_R) \iff \beta\gamma \notin \mathcal{L}(M_R)$.
Por otro lado, como $\hat{\delta}'(q_0', \alpha) = \hat{\delta}'(q_0', \beta)$, es obvio que

$$\hat{\delta}'(q_0', \alpha\gamma) \in F \wedge \hat{\delta}'(q_0', \beta\gamma) \in F$$

o ninguno de los dos perteneces a F. De esto se inifiere que $\alpha\gamma \in \mathcal{L}(M') \iff \beta\gamma \in \mathcal{L}(M')$.
Pero entonces, $\mathcal{L}(M') \neq \mathcal{L}(M)$, se contradice nuestra hipotesis inicial.

5. Lenguajes regulares

5.1. Lema de pumping

Propiedad: Sea $\mathcal{L}$ un lenguaje regular, si las longitudes de las cadenas de un lenguaje $\mathcal{L}$ están acotadas superiormente, entonces $\mathcal{L}$ tiene que ser finito.

Propiedad: Si $\mathcal{L}$ es un luenguaje regular infinito, entonces el grafo de un autómata fínito que acepte $\mathcal{L}$ tiene que tener un camino desde el estado inicial hasta algún estado final que paso por algún ciclo.

Lema de pumping: Sea $\mathcal{L}$ un lenguaje regular, si $\mathcal{L}$ es infinito, entonces todas las cadenas ω de longitud mayor o igual a n (para algún $n > 1$) van a ser de la forma $\omega = xy^iz$, es decir hay una parte de ω que se repite i cantidad de veces:

$$\mathcal{L} \text{ es regular e infinito} \implies \exists n \in \mathbb{N} \text{ tal que}$$
$$\forall \omega \in \mathcal{L},\ |\omega| \geq n : \big(\exists x, y, z \in \Sigma^* : \omega = xyz \wedge |xy| \leq n \wedge |z| \geq 1 \wedge (\forall i \geq 0 :\ xy^iz \in \mathcal{L}\big)$$

DEMOSTRACIÓN

Supongamos que $\mathcal{L}$ es un lenguaje regular. Entonces existe una AFD $A = \langle Q, \Sigma, \delta, q_0, F\rangle$ que acepta $\mathcal{L}$.
Sea $n = |Q|$ la cantidad de estados de A y $\omega = a_1a_2 \dots a_m \in \Sigma^*$ de longitud $m > n$. Para cada $i = 0, \dots, m$ definamos el estado $p_i = \hat{\delta}(q_0, a_1 \dots a_i)$ (el estado en el que se encuentra A después de haber consumido los primeros i símbolos de ω.
Como el A solo tiene n estados pero hay $m > n$ estados p_i, es imposible que todos sean distintos. Por lo tanto, existen $0 \leq i < j \leq n$ tales que $p_i = p_j$.
Considerar entonces la siguiente descomposición para $\omega = xyz$:

$x = a_1 \dots a_i$

$y = a_{i+1} \dots a_j$

$z = a_{j+1} \dots a_m$

Entonces podemos concluir que:

$\hat{\delta}(q_0, x) = p_i$

$\hat{\delta}(p_i, y) = p_j$ (que como son el mismo estado implica que A tiene un ciclo)

$\hat{\delta}(p_j, z) = p_m$

Observar que x podría ser la cadena vacía cuando $i = 0$, z podría ser vacía si $j = n = m$, pero y no puede ser vacía ya que se tomó $i < j$.
Vimos entonces que si $\mathcal{L}$ es regular y $|\omega| \geq n$ entonces podemos dividirla en cadenas x, y, z tal que $|xy| \leq n$ y $|z| \geq 1$. Ahora vamos a ver que si $xyz \in \mathcal{L}$ entonces $xy^kz \in \mathcal{L}$ para todo $k \geq 0$.

- Si $k = 0$ entonces $xy^0z = xz$:

$$\hat{\delta}(q_0, xz) = \hat{\delta}(\hat{\delta}(q_0, x), z) = \hat{\delta}(p_i, z) = \hat{\delta}(p_j, z) = p_m$$

  Y p_m es un estado final pues es el mismo estado al que llegamos si la entrada fuese $xyz \in \mathcal{L}$. Entonces $xz = xy^0z \in \mathcal{L}$.

- Si $i > 0$. Entonces A consume x desde q_0 y llega hasta p_i. Luego A consume y desde p_i y llega hasta p_j (que son iguales) y repite este ciclo k veces hasta consumir todas las apariciones de y en la cadena. Finalmente A consume z desde p_j y llega hasta p_m. Entonces A llega a un estado final y por lo tanto $xy^kz \in \mathcal{L}$.

Contrarecíproco:

$\forall n \in \mathbb{N}\ \exists \omega \in \mathcal{L}$ tal que $|\omega| \geq n \wedge \forall x, y, z \in \Sigma^* : \omega \neq xyz \vee |xy| \geq n \vee |z| \leq 1 \vee \exists i \geq 0,\ xy^iz \notin \mathcal{L}$
$\implies \mathcal{L}$ no es regular

5.2. Operaciones sobre lenguajes regulares

5.2.1. Unión de lenguajes regulares

Dados M_1 y M_2 AFDs que aceptan los lenguajes L_1 y L_2, respectivamente, podemos construir un AFD $M = \langle Q_\cup, \Sigma, \delta_\cup, q_{0\cup}, F_\cup \rangle$ que acepte el lenguaje $L_1 \cup L_2$ con:

- $Q_\cup = Q_1 \times Q_2$
- $\delta_\cup((q, r), a) = (\delta_1(q, a), \delta_2(r, a))$ para $q \in Q_1$, $r \in Q_2$ y $a \in \Sigma$
- $q_{0\cup} = (q_{0_1}, q_{0_2})$
- $F_\cup = \{(q, r) \in Q_\cup \mid q \in F_1 \vee r \in F_2\}$

DEMOSTRACIÓN

$$\begin{aligned}
\alpha \in \mathcal{L}(M_\cup) &\iff \hat{\delta}_\cup((q_{0_1}, q_{0_2}), \alpha) \in F_\cup \iff (\hat{\delta}_1(q_{0_1}, \alpha), \hat{\delta}_2(q_{0_2}, \alpha)) \in F_\cup \\
&\iff \hat{\delta}_1(q_{0_1}, \alpha) \in F_1 \vee \hat{\delta}_2(q_{0_2}, \alpha) \in F_2 \\
&\iff \alpha \in \mathcal{L}(M_1) \vee \alpha \in \mathcal{L}(M_2)
\end{aligned}$$

5.2.2. Intersección de lenguajes regulares

Dados M_1 y M_2 AFDs que aceptan los lenguajes L_1 y L_2, respectivamente, podemos construir un AFD $M_\cap = \langle Q_\cap, \Sigma, \delta_\cap, q_{0\cap}, F_\cap \rangle$ que acepte el lenguaje $L_1 \cap L_2$ con:

- $Q_\cap = Q_1 \times Q_2$
- $\delta_\cap((q, r), a) = (\delta_1(q, a), \delta_2(r, a))$ para $q \in Q_1$, $r \in Q_2$ y $a \in \Sigma$
- $q_{0\cap} = (q_{0_1}, q_{0_2})$
- $F_\cap = \{(q, r) \in Q_\cap \mid q \in F_1 \wedge r \in F_2\}$

DEMOSTRACIÓN

$$\begin{aligned}
\alpha \in \mathcal{L}(M_\cap) &\iff \hat{\delta}_\cap((q_{0_1}, q_{0_2}), \alpha) \in F_\cap \iff (\hat{\delta}_1(q_{0_1}, \alpha), \hat{\delta}_2(q_{0_2}, \alpha)) \in F_\cap \\
&\iff \hat{\delta}_1(q_{0_1}, \alpha) \in F_1 \wedge \hat{\delta}_2(q_{0_2}, \alpha) \in F_2 \\
&\iff \alpha \in \mathcal{L}(M_1) \wedge \alpha \in \mathcal{L}(M_2)
\end{aligned}$$

5.2.3. Complemento de lenguajes regulares

Teorema 5.1. El conjunto de lenguajes regulares incluido en Σ^* es cerrado respecto de la complementación. Es decir, si $L \in \mathcal{L}(\Sigma^*)$ es un lenguaje regular entonces $\bar{L} \in \mathcal{L}(\Sigma^*)$ también lo es.

DEMOSTRACIÓN

Sea $\mathcal{L} = \mathcal{L}(M)$ con $M = (Q, \Sigma, \delta, q_0, F)$ un AFD cuya función de transición δ está definida para todos los elementos del alfabeto Σ. Si hay algún símbolo $a \in \Sigma$ que falta en las posibles transiciones desde un estado $q \in Q$ entonces se puede agregar un estado trampa q_{trampa} (no final) y una transición $\delta(q, a) = q_{trampa}$ para completar la definición de la función.
Si armamos el automáta $M_\neg = \langle Q, \Sigma, \delta, q_0, Q - F \rangle$, entnonces $\alpha \in \mathcal{L}M_\neg \iff \delta(q_0, w) \in Q - F \iff \alpha \in \Sigma^* - L$.

Teorema 5.2. El conjunto de lenguajes regulares es cerrado respecto de la intersección

DEMOSTRACIÓN

Como ya probamos que el conjunto de lenguajes regulares es cerrado respecto de la unión y el complemento entonces si logramos escribir la interesección como combinación de estas dos operaciones podremos decir que también es cerrada respecto de la intersección:

$$L_1 \cap L_2 = \overline{\overline{L_1 \cap L_2}} = \overline{\overline{L_1} \cup \overline{L_2}}$$

Entonces si L_1 y L_2 son lenguajes regulares, entonces sus complementos también lo son y la unión de los complementos y el complemento de la misma también lo son.

Teorema 5.3. De estos ultimos tres teoremas puede deducirse que la unión finita y la intersección finita de lenguajes regulares dan por resultado un lenguaje regular.

$$\forall n \in \mathbb{N}, \ \bigcup_{i=1}^{n} L_i \text{ es regular}$$

$$\forall n \in \mathbb{N}, \ \bigcap_{i=1}^{n} L_i \text{ es regular}$$

DEMOSTRACIÓN

Por inducción sobre n:

Caso base: $n = 0$.

$$\bigcup_{i=1}^{0} L_i = \emptyset \text{ es regular}$$

$$\bigcap_{i=1}^{0} L_i = \emptyset \text{ es regular}$$

- **Caso inductivo:** $n \geq 1$. Supongamos que $L_1, \ldots, L_n$ son lenguajes regulares y que $\bigcup_{i=1}^{n-1} L_i$ y $\bigcap_{i=1}^{n-1} L_i$ son lenguajes regulares. Entonces:

$$\bigcup_{i=1}^{n} L_i = \bigcup_{i=1}^{n-1} L_i \cup L_n \text{ es regular}$$

$$\bigcap_{i=1}^{n} L_i = \bigcap_{i=1}^{n-1} L_i \cap L_n \text{ es regular}$$

Teorema 5.4. Todo lenguaje finito es regular

DEMOSTRACIÓN

Sea L un lenguaje finito tal que $|L| = n$ y $x_i \in L$ con $1 \leq i \leq n$.
Entonces podemos definir n lenguajes $L_i = \{x_i\}$ regulares. Por el teorema anterior, la unión finita de lenguajes regulares es un lenguaje regular. Entonces $L = \bigcup_{i=1}^{n} L_i$ es un lenguaje regular.

5.3. Problemas decicibles acerca de lenguajes regulares

1. **Pertenencia:** Dado un lenguaje regular L y una cadena w, determinar si $w \in L$.

   Se construye un AFD M que reconozca L y se verifica si w es aceptada por M.

2. **Finitud:** Dado un lenguaje regular L, determinar si L es finito.

   Un lenguaje regular L es finito, si en su AFD ningún ciclo que es alcanzable desde el estado inicial puede, a su ve, alcanzar algún estado final.

   $$L \text{ finito} \iff \left(\forall q \in Q, \left(q_0 \overset{*}{\Rightarrow} q \wedge q \overset{*}{\Rightarrow} f \in F \implies \left(\nexists q \overset{+}{\to} q\right)\right)\right)$$

   Escrito en función de δ:

   $$L \text{ finito si y solo si: } \forall q \in Q \text{ tal que } \left(\exists \alpha, \omega \in \Sigma^*,\ \hat{\delta}(q_0, \alpha) = q \wedge \hat{\delta}(q, \omega) = f \in F\right)$$
   $$\text{vale que} \left(\nexists \beta \in \Sigma^+,\ \hat{\delta}(q, \beta) = q\right)$$

3. **Vacuidad:** Dado un lenguaje regular L, determinar si L es vacío.

   Se construye el AFD M que reconozca L y se verifica si el conjunto A de estados alcanzables de M tiene un estado final. Si $F \cap A = \emptyset$, entonces L es vacío.

4. **Equivalencia:** Dados dos lenguajes regulares L_1 y L_2, determinar si L_1 y L_2 son equivalentes.

   Si el lenguaje $(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2)$ es vacío, entonces L_1 y L_2 son equivalentes, si no no lo son.

6. Autómatas de Pila

Los autómatas de pila (AP) son autómatas fínitos que además tienen la capacidad de almacenar información en una pila con dos operaciones básicas: apilar y desapilar. La operación apilar consiste en agregar un elemento a la pila, mientras que la operación desapilar consiste en eliminar el elemento que se encuentra en la cima de la misma.

Cuando se va a realiar una transición, se consume un elemento de la cadena de entrada y un elemento del tope de la pila. Al moverse al nuevo estado, el automáta pushea (apila) una cadena de símbolos en la pila (que puede ser vacía).

Definición: Un autómata de pila (AP) es una 7-tupla $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ donde:

- Q es un conjunto finito de estados.
- Σ es el alfabeto de entrada y es finito.
- Γ es un alfabeto de la pila y es finito.
- δ es la función de transición, que es un conjunto de reglas de la forma:

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \to \mathcal{P}(Q \times \Gamma^*)$$

$$\delta(q, a, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \ldots, (p_n, \gamma_n)\}$$

  δ indica el estado que p_i al que se transiciona y la cadena γ_i que se apila en la pila.

- $q_0 \in Q$ es el estado inicial.
- $Z_0 \in \Gamma$ es el símbolo inicial de la pila.
- $F \subseteq Q$ es el conjunto de estados finales.

6.1. Configuración instantanea

Una configuración instantánea de un autómata de pila es una 3-tupla $(q, w, \gamma) \in Q \times \Sigma \times \Gamma^*$ donde:

- $q \in Q$ es el estado actual.
- $w \in \Sigma^*$ es la parte de la cadena de entrada que no ha sido consumida.
- $\gamma \in \Gamma^*$ es el contenido de la pila.

La configuracion inicial de un automáta será entonces $\sigma = (q_0, \alpha, Z_0)$ si α es la cadena que usamos de entrada.

Cambio de configuración: Sea $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$ un automáta de pila, $a \in \Sigma$, $\alpha \in \Sigma^*$, $t \in \Gamma$, $\tau, \pi \in \Sigma^*$ y $q, r \in Q$. Definimos las transiciones ($\vdash$) entre dos configuraciones instantaneas de la siguiente manera:

- $(q, a\alpha, t\pi) \vdash (r, \alpha, \tau\pi)$ si $(r, \tau) \in \delta(q, a, t)$.
- $(q, \alpha, t\pi) \vdash (r, \alpha, \tau\pi)$ si $(r, \tau) \in \delta(q, \lambda, t)$.

6.2. Lenguajes reconocidos por un autómata

6.3. Lenguaje aceptado por estado final

Son todas las cadenas $\alpha \in \Sigma^*$ que hacen que el automáta llegue a un estado final $q \in F$.

$$\mathcal{L}(M) = \{\alpha \in \Sigma^* \mid \exists q \in F \text{ tal que } (q_0, \alpha, Z_0) \overset{*}{\vdash} (q, \lambda, Z_0)\}$$

6.3.1. Lenguaje aceptado por pila vacía

Son todas las cadenas $\alpha \in \Sigma^*$ que hacen que el automáta llegue a un estado en el cual la pila quede vacía.

$$\mathcal{L}_\lambda(M) = \{\alpha \in \Sigma^* \mid \exists q \in F \text{ tal que } (q_0, \alpha, Z_0) \overset{*}{\vdash} (q, \lambda, \lambda)\}$$

Teorema 6.1. Sea $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\rangle$ un automáta que acepta el lenguaje $\mathcal{L}(M)$ por estado final, entonces existe $M' = \langle Q', \Sigma, \Gamma', \delta', q_0', X_0, \emptyset\rangle$ tal que $\mathcal{L}_\lambda(M') = \mathcal{L}(M)$.

DEMOSTRACIÓN

Construimos M' de la siguiente manera:

- $Q' = Q \cup \{q_\lambda, q_0'\}$
- $\Gamma' = \Gamma \cup \{X_0\}$
- $\delta' : Q' \times \Sigma \times \Gamma' \to \mathcal{P}(Q' \times \Gamma')$

  1. $\delta'(q_0', \lambda, X_0) = \{(q_0, Z_0X_0)\}$

     M' entra al estado inicial de M con la pila Z_0X_0. como X_0 no es un símbolo de la pila de M, M no puede vaciar la pila.

  2. $\forall q \in Q, a \in \Sigma\cup\{\lambda\}, \delta'(q, a, Z) = \{(r, \tau) \mid (r, \tau) \in \delta(q, a, Z) \text{ con } r \in Q \wedge \tau \in \Gamma^*\}$

     M' sigue las mismas transiciones que M para todos los estados originales.

3. $\forall q \in F, \forall Z \in \Gamma', (q_\lambda, \lambda) \in \delta'(q, \lambda, Z)$

   Cuando M' llega a un estado que es final en M automáticamente se activa el el estado q_λ que desencola el último elemento de la pila (y no pushea nada).

4. $\forall Z \in \Gamma', (q_\lambda, \lambda) \in \delta'(q_\lambda, \lambda, Z)$

   Cuando M' llega al estado q_λ, sin importar cual es el tope de la pila lo desencola hasta que la pila queda vacía (hay un loop sobre q_λ que consume toda la pila).

Veamos que $\mathcal{L}_\lambda(M) = L$, lo vamos a demostrar en dos pasos:

$\mathcal{L}(M) \subseteq \mathcal{L}_\lambda(M)$:

Si $\alpha \in \mathcal{L}(M)$ entonces $(q_0, \alpha, Z_0) \overset{*}{\underset{M}{\vdash}} (q, \lambda, \gamma)$ con $q \in F$ (por def.).

Por la regla 1, tenemos que $(q_0', \alpha, X_0) \overset{*}{\underset{M'}{\vdash}} (q_0, \alpha, Z_0X_0)$

Por la regla 2, $(q_0, \alpha, Z_0) \overset{*}{\underset{M'}{\vdash}} (q, \lambda, \gamma)$, entonces

$$(q_0', \alpha, X_0) \overset{*}{\underset{M'}{\vdash}} (q_0, \alpha, Z_0X_0) \overset{*}{\underset{M'}{\vdash}} (q, \lambda, \gamma X_0)$$

Por las reglas 3 y 4 es cierto que $(q, \lambda, \gamma X_0) \overset{*}{\underset{M'}{\vdash}} (q_\lambda, \lambda, \lambda)$

Por lo tanto, $\alpha \in \mathcal{L}_\lambda(M)$

$\mathcal{L}_\lambda(M) \subseteq L$:

Si $\alpha \in \mathcal{L}_\lambda(M)$ entonces, por como construimos M' vale que

$$(q_0', \alpha, X_0) \underset{M'}{\vdash} \underbrace{(q_0, \alpha, Z_0X_0) \overset{*}{\underset{M'}{\vdash}} (q, \lambda, \gamma X_0)}_{A} \overset{*}{\underset{M'}{\vdash}} (q_\lambda, \lambda, \lambda)$$

De A se puede ver que $(q_0, \alpha, Z_0) \overset{*}{\underset{M}{\vdash}} (q, \lambda, \gamma)$. Además como los únicos estados que pueden llegar a q_λ son estados finales de M (por 3) vale que $q \in F$. Luego $\alpha \in \mathcal{L}(M)$

Teorema 6.2. Sea $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, \emptyset \rangle$ y $\mathcal{L}_\lambda(M)$ el lenguaje que acepta por pila vacía. Entonces existe $M' = \langle Q', \Sigma, \Gamma', \delta', q_0', X_0, F \rangle$ tal que $\mathcal{L}(M') = \mathcal{L}_\lambda(M)$.

DEMOSTRACIÓN

Construimos M' de la siguiente manera:

- $Q' = Q \cup \{q'_0, q'_f\}$
- $\Gamma' = \Gamma \cup \{X_0\}$
- $F = \{q'_f\}$
- $\delta' : Q' \times (\Sigma \cup \{\lambda\}) \times \Gamma' \to \mathcal{P}(Q' \times \Gamma'^*)$

 1. $\delta'(q'_0, \lambda, X_0) = \{(q_0, Z_0X_0)\}$

 M' activa automaticamente el estado q_0 con pila Z_0X_0 y como X_0 no es un símbolo de Γ, M no puede consumirlo y vaciar la pila.

 2. $\forall q \in Q, \forall a \in \Sigma \cup \{\lambda\}, \delta'(q, a, Z) = \delta(q, a, Z)$

 Simulación de M.

 3. $\forall q \in Q,\ (q'_f, \lambda) \in \delta(q, \lambda, X_0)$

 Todos los estados a los que se puede llega con la pila vacía en M, llegarán con X_0 en M'. En este caso, podrán saltar automaticamente al estado $q'_f \in F$.

La demostración de que ambos lenguajes son equivalentes es similar a la anterior.

6.4. Gramáticas independientes de contexto

Para toda grámatica independiente de contexto G, puede definirse un autómata de pila M que acepta por pila vacía el lenguaje generado por dicha gramática.

DEMOSTRACIÓN

Esto no es una demostración completa.

Sea $G = \langle V_N, V_T, P, S\rangle$ una gramática independiente de contexto, vamos a contruir $M = \langle Q, \Sigma, \Gamma, \delta, q_0, S, \emptyset\rangle$ tal que $\mathcal{L}(M) = \mathcal{L}(G)$:

- $Q = \{q_0\}$
- $\Sigma = V_T$
- $\Gamma = V_N \cup V_T$
- $\delta(q_0, a, t) = \begin{cases} \{(q_0, \alpha) : t \to \alpha \in P\} & \text{si } t \in V_N \wedge a = \lambda \\ \{(q_0, \lambda)\} & \text{si } t \in V_T \wedge a = t \neq \lambda \end{cases}$

Veamos que M acepta $\mathcal{L}(G)$:

- Si en el tope de la pila hay un símbolo no términal $t \in V_N$, entonces el autómata lo remplazará por el lado derecho α de alguna producción que tenga a t en el lado izquierdo. Esto lo hará de manera tal que el símbolo que está más a la izquierda en el lado derecho de dicha producción sea el siguiente tope de la pila.

- Si en el tope de la pila hay un símbolo terminal $t \in V_T$, entonces el autómata verificará que este sea igual al próximo símbolo en la cadena de entrada y lo desapilará.

6.5. Autómatas de pila deterministicos

Es un autómata de pila $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$ que cumple que para todo $q \in Q,\ a \in \sigma$ y $A \in \Gamma$:

1. $|\delta(q, a, A)| \leq 1$

2. $|\delta(q, \lambda, A)| \leq 1$

3. $|\delta(q, \lambda, A) = 1 \implies |\delta(q, a, A)| = 0$

6.5.1. Propiedad del prefijo

Se dice que un lenguaje L posee la **propiedad del prefijo** si y sollo si para todo par de cadenas **no nulas** x e y es cierto que $x \in L \implies xy \notin L$.

Si un lenguaje L no posee la propiedad del prefijo, entonces todo autómata de pila M que acepte L por pila vacía es necesariamente **no deterministico**

7. Gramáticas Independientes del Contexto

Una gramática $G = \langle V_N, V_T, P, S\rangle$ es independiente del contexto si y solo si las producciones en P son de la forma $A \to \alpha$ con $A \in V_N$ y $\alpha \in (V_N \cup V_T)^*$.

Si en particular para toda $A \to \alpha \in P$ pasa que $\alpha \in (V_N \cup V_T)^+$ (o sea, sin reglas borradoras), entonces decimos que G es una **gramática propia**.

7.1. Árboles de Derivación

Sea $G = \langle V_N, V_T, P, S\rangle$ una gramática independiente del contexto. Un árbol de derivación de una cadena ω en G es una estructura de árbol tal que:

- Cada vértice posee una etiqueta que pertenece al conjunto $V_N \cup V_T \cup \{\lambda\}$.
- La raíz del árbol posee la etiqueta S.
- Si un vértice es interior, entonces su etiqueta pertenece al conjunto V_N.
- Si un vértice v_n posee la etiqueta A y sus hijos $v_1, \ldots, v_m$ poseen las etiquetas $X_1, \ldots, X_m$, entonces $A \to X_1 \ldots X_m \in P$.
- Si un vértice es hoja, entonces su etiqueta pertenece al conjunto $V_T \cup \{\lambda\}$.
- Si una hoja posee la etiqueta λ, entonces es el único hijo de su.

Camino: Sea $A \in V_N$ y $X \in V_n \cup V_T$ un vértice de un árbol de derivación. El camino de A a X es la secuencia de vértices $A = v_1, v_2, \ldots, v_n = X$ tal que v_i es hijo de v_{i+1} para todo $i \in \{1, \ldots, n-1\}$.

Altura de un árbol: La altura de un árbol de derivación es la longitud del camino más largo de la raíz a una hoja.

Longitud de un camino: La longitud de un camino es la cantidad de arcos que lo componen.

Lema 7.1. Sea $G = \langle V_N, V_T, P, S\rangle$ una gramática independiente del contexto con $P \neq \emptyset$ y sea $T(S)$ el árbol de derivación de S en G para algún $\alpha \in \Sigma^*$ de altura h.

$$\text{Si } a = \text{máx}\{|\beta| : A \to \beta \in P\}, \text{ entonces } |\alpha| \leq a^h$$

DEMOSTRACIÓN

Por inducción en h:

- **Caso base:** $h = 0$. El único árbol de derivación posible de esta altura es el símbolo S. Por lo tanto $a^h = a^0 = 1 = |S|$.
- **Paso inductivo:** Sea $\mathcal{T}(S)$ el árbol de derivación para α de altura $h+1$. Sea $\gamma \in (V_N \cup V_T)^+$ una cadena tal que $\gamma \Rightarrow \alpha$ en G. Entonces el árbol de derivación de γ en G tiene altura h.

Por hipotesis inductiva, vale que $|\gamma| \leq a^h$. Por lo tanto, $|\alpha| \leq a^h$.

Por otra parte, $|\alpha| \leq a|\gamma|$ ya que en el peor de los casos cada símbolo de γ usa la reducción de mayor tamaño. Luego, $|\alpha| \leq aa^h = a^{h+1}$.

7.2. Gramáticas Ambiguas

Una gramática independiente del contexto G es **ambigua** si y solo si $\exists \alpha \in \mathcal{L}(G)$ tal que posea más de un árbol de derivación.

Lenguaje intrisícamente ambiguo: Un lenguaje independiente del contexto L es **íntrisicamente ambiguo**si y solo si todas las grámaticas que lo tienen como lenguaje son ambiguas.

Derivación más a la izquierda: Una **derivación más a la izquierda** $\left(\underset{L}{\Rightarrow}\right)$ de una cadena ω es aquella que se obtiene remplazando el primer símbolo no terminal que contenga por alguna de sus derivaciónes: Si $A \in V_N$, $\alpha \in V_T^*$, $\beta \in (V_N \cup V_T)^*$ y $A \rightarrow \gamma \in P$ entonces $\alpha A \beta \underset{L}{\Rightarrow} \alpha\gamma\beta$

Derivación más a la derecha: Una **derivación más a la derecha** $\left(\underset{R}{\Rightarrow}\right)$ de una cadena ω es aquella que se obtiene remplazando el último símbolo no terminal que contenga por alguna de sus derivaciónes: Si $A \in V_N$, $\alpha \in (V_N \cup V_T)^*$, $\beta \in V_T^*$ y $A \rightarrow \gamma \in P$ entonces $\alpha A \beta \underset{R}{\Rightarrow} \alpha\gamma\beta$

7.3. Lema de Pumping para Lenguajes Independientes del Contexto

Sea L un lenguaje independiente del contexto sobre un alfabeto Σ, existe $n > 0$ tal que para todo $\alpha \in L$, $|\alpha| \geq n$ existe $r, x, y, z, s \in \Sigma^*$ tales que:

1. $\alpha = rxyzs$
2. $|xyz| \leq n$
3. $|xz| > 0$
4. $\forall i \geq 0, rx^iyz^is \in L$

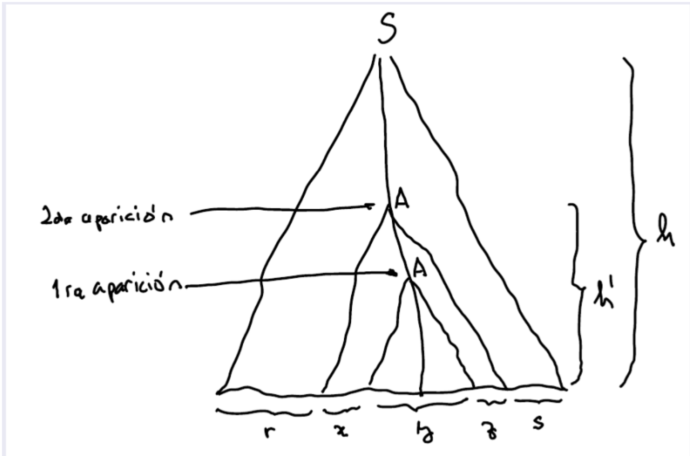
7.3.1. Demostración

Sea $G = \langle V_N, V_T, P, S\rangle$ una gramática independiente del contexto y sea $a = \text{máx}\{|\beta| : A \rightarrow \beta \in P\}$.

1. Tomemos $n = a^{|V_N|+1}$. Sea $\alpha \in \mathcal{L}(G)$ tal que $|\alpha| \geq n$ y sea $T(S)$ un árbol mínimo de derivación de α en G, es decir tiene la mínima altura posible y tal que no existe otro con menos derivaciones.

   Por el lema **??**, resulta que $a^h \geq |\alpha| \geq n = a^{|V_N|+1}$. Por lo tanto, $h \geq |V_N| + 1$. Entonces existe algún símbolo $b \in \alpha$ tal que su camino desde la raíz es de longitud $h \geq |V_N| + 1$.

Como la cantidad de símbolos no terminales es $|V_N|$, entonces en ese camino seguramente existe un no-terminal A repetido. Recorriendo el camino de forma ascendente, buscamos sus primeras dos apariciones:

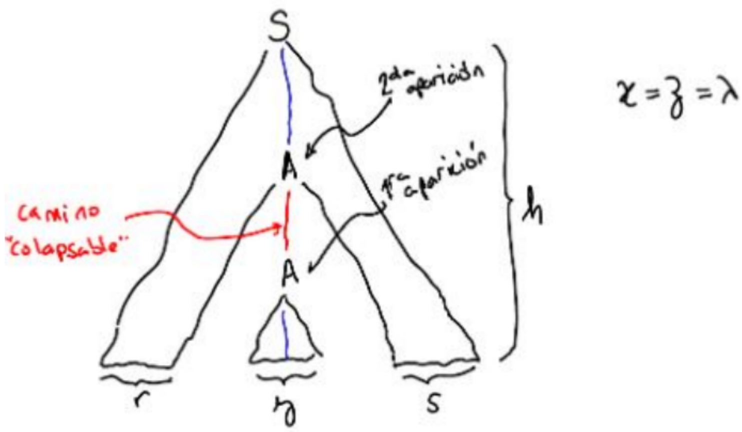


Entonces podemos escribir $\alpha = rxyzs$ con r, x, y, z y s como se muestra en la figura.

2. $|xyz| \leq n = a^{|V_n|+1}$. Como A es el primer no terminal que se repite, podemos asegurar que $h' \leq |V_N| + 1$. La segunda aparición de A tiene que pasar a lo sumo $|V_n|$ pasos mas adelantes, sino se debería repetir otro no terminal antes. Entonces por el lema **??**, resulta que $|xyz| \leq a^{h'} \leq a^{|V_N|+1}$.

3. $|xz| > 0$.

   Supongamos que $|xz| = 0$, esto quiere decir que podriamos remplazar el subárbol con raíz en la segunda aparición de A por el subárbol con raíz en la primera aparición.



Esto es absurdo ya que habíamos dicho que el árbol era mínimo por lo que no debería poder colapsarse ninguno de sus caminos.

4. $\forall i \geq 0, rx^iyz^is \in L$ Finalmente, vamos a demostrar esto por inducción:

   - **Caso base** ($i = 0$)**:** Sabemos que $S \overset{*}{\Rightarrow} rAs$ y $A \overset{*}{\Rightarrow} y$, entonces $S \overset{*}{\Rightarrow} rAs \overset{*}{\Rightarrow} rys$. Además $rys = rx^0yz^0s$.
   - **Paso inductivo** $i-1 \Rightarrow i$**:**

     Hipotesis inductiva: $S \overset{*}{\Rightarrow} rx^{i-1}Az^{i-1}s$.

     Sabemos $A \overset{*}{\Rightarrow} xyz$, entonces:

$$S \overset{*}{\Rightarrow} rx^{i-1}Az^{i-1}s \Rightarrow rx^{i-1}xyzz^{i-1}s \Rightarrow rx^iyz^is$$

     Y por lo tanto $rx^iyz^is \in L$.

7.4. Propiedades de los Lenguajes Independientes del Contexto

- Si L_1 y L_2 son lenguajes independientes del contexto, entonces $L_1 \cup L_2$ también lo es.

DEMOSTRACIÓN

Como L_1 y L_2 son dos lenguajes independientes del contexto, entonces existen dos grámatica independientes del contexto $G_1 = \langle V_N, \Sigma, P_1, S_1 \rangle$ y $G_2 = \langle V_N, \Sigma, P_2, S_2 \rangle$. tales que $L_1 = \mathcal{L}(G_1)$ y $L_2 = \mathcal{L}(G_2)$. Supongamos, además, sin perdida de generlidad que $V_{N_1} \cap V_{N_2} = \emptyset$. Definamos entonces:

$$G = \langle V_{N_1} \cup V_{N_2} \cup \{S\}, \Sigma, P_1 \cup P_2 \cup \{S \to S_1,\ S \to S_2\}, S \rangle$$

Veamos que $\forall \alpha \in \Sigma^*, \alpha \in \mathcal{L}(G) \iff \alpha \in \mathcal{L}(G_1) \cup \mathcal{L}(G_2)$.

$$\begin{aligned} \alpha \in \mathcal{L}(G) &\iff S \overset{*}{\Rightarrow} \alpha \iff S_1 \overset{*}{\Rightarrow} \alpha \vee S_2 \overset{*}{\Rightarrow} \alpha \\ &\iff \alpha \in \mathcal{L}(G_1) \vee \alpha \in \mathcal{L}(G_2) \\ &\iff \alpha \in \mathcal{L}(G_1) \cup \mathcal{L}(G_2) \end{aligned}$$

- Si L_1 y L_2 son lenguajes independientes del contexto, entonces L_1L_2 también lo es.

DEMOSTRACIÓN

- Si L es un lenguaje independiente del contexto, entonces L^+ también lo es. Demostrar?
- Si L es un lenguaje independiente del contexto, entonces L^* también lo es. Demostrar?

- Si L_1 y L_2 son lenguajes independientes del contexto, entonces $L_1 \cap L_2$ no siempre es lenguaje independiente del contexto Demostrar?
- El lenguaje $L = \{ww : w \in \Sigma^*\}$ no es independiente del contexto. Demostrar?
- Existe un lenguaje independiente del contexto que **no-deterministico**. Demostrar?

Símbolo alcanzable: Dada una grámatica $G = \langle V_N, V_T, P, S\rangle$, un símbolo $A \in V_N$ es alcanzable si existe una forma sentencial que lo contiene, $S \stackrel{*}{\Rightarrow} \ldots A \ldots$.

Símbolo activo: Dada una grámatica $G = \langle V_N, V_T, P, S\rangle$ un símbolo $A \in V_N$ es activo si existe $x \in V_T^*$ tal que $A \stackrel{*}{\Rightarrow} x$.

Símbolo útil: Un no terminal A es útil si y solo si es parte de una forma sentencial que genera una cadena de terminales, osea, si $S \stackrel{*}{\Rightarrow} \alpha A \beta \stackrel{*}{\Rightarrow} w$ con $w \in \Sigma^*$ y $\alpha, \beta \in (V_T \cup V_N)^*$.