AUTÓMATAS Y LENGUAJES FORMALES LENGUAJES REGULARES II LENGUAJES INDEPENDIENTES DEL CONTEXTO

Francisco Hernández Quiroz

Departamento de Matemáticas
Facultad de Ciencias, UNAM
E–mail: fhq@ciencias.unam.mx
Página Web: www.matematicas.unam.mx/fhq

Facultad de Ciencias

Un lenguaje no regular I

El lenguaje $\{a^n b^n\}$ no es regular. Supongamos que lo es y que A es un autómata determinista con k estados y $L(A) = \{a^n b^n\}$.
Sea la cadena $a^m b^m$, con $m > k$. Durante su lectura, el autómata debe haber pasado al menos dos veces por un mismo estado (hay menos estados que copias de a).
Sea q este estado repetido. Entonces:

$$\begin{array}{rcll} \delta^*(s, a^i) & = & q & i < m \\ \delta^*(q, a^j) & = & q & j \neq 0 \\ \delta^*(q, a^r b^m) & = & f \in F & i + j + r = m\,. \end{array}$$

Como A es determinista, tenemos que para toda $p \in \mathbb{N}$

$$\delta^*(q, a^{jp}) = q$$

y en consecuencia

$$\delta^*(s, a^i a^{jp} a^r b^m) = f \in F$$

Un lenguaje no regular II

Pero

$$i + jp + r \neq m,$$

salvo en el caso en que $p = 1$. Por tanto, aunque $a^i a^{jp} a^r b^m \in L(A)$,

$$a^i a^{jp} a^r b^m \notin \{a^n b^n\},$$

lo cual es una contradicción. En conclusión, este lenguaje no es regular.

Teorema del bombeo

El resultado anterior se puede generalizar con el siguiente

Teorema. Sea L un lenguaje regular. Entonces, $\exists k \in \mathbb{N}$ tal que $\forall \alpha, \beta, \gamma \in \Sigma^*$, si

$$\alpha\beta\gamma \in L \quad \text{y} \quad k \leq |\beta|,$$

entonces $\exists \eta, \theta, \varphi \in \Sigma^*$ tales que

$$\beta = \eta\theta\varphi \quad \text{y} \quad \theta \neq \varepsilon \quad \text{y} \quad \alpha\eta\theta^i\varphi\gamma \in L \quad \forall i \in \mathbb{N}.$$

La demostración generaliza el argumento del ejemplo anterior, en el que $\alpha = \varepsilon$, $\beta = a^m$, $\eta = a^i$, $\theta = a^j$, $\varphi = a^r$ y $\gamma = b^m$.

Aplicaciones del lema del bombeo I

Ejemplo 1. El lenguaje $\{a^{2^n}\}$ no es regular. Supongamos que lo es y $A \in \text{DFA}$ reconoce este lenguaje. Sea k el número de estados de A y sea m tal que $2^m > k$. El teorema del bombeo nos dice que $\exists \eta, \theta, \varphi$ tales que

$$a^{2^m} = \alpha\eta\theta^i\varphi\gamma \in L(A) \qquad \forall i \in \mathbb{N}.$$

Como $\theta \neq \varepsilon$, la afirmación anterior quiere decir que

$$|\alpha| + |\eta| + i|\theta| + |\varphi| + |\gamma|$$

siempre es una potencia de 2, lo cual es obviamente falso. Por tanto, A no puede existir y $\{a^{2^n}\}$ no es regular.

Aplicaciones del lema del bombeo II

Ejemplo 2. Sean $\alpha \in \Sigma^*$ y $a \in \Sigma$. Definimos

$$\#a(\alpha) = \text{ el número de apariciones de } a \text{ en } \alpha.$$

Entonces

$$\{\alpha \in \{a, b\}^* \mid \#a(\alpha) = \#b(\alpha)\}$$

no es regular. La demostración en este caso es indirecta:

1. El lenguaje $\{a^*b^*\}$ es regular.
2. $\{a^*b^*\} \cap \{\alpha \in \{a, b\}^* \mid \#a(\alpha) = \#b(\alpha)\} = \{a^nb^n\}$.
3. Los lenguajes regulares son cerrados bajo $\cap$.
4. Por tanto, $\{\alpha \in \{a, b\}^* \mid \#a(\alpha) = \#b(\alpha)\}$ no puede ser regular.

Relaciones de Myhill–Nerode I

Sea L un lenguaje regular y sea $A \in \text{DFA}$ tal que $L(A) = L$, con $A = (Q, \Sigma, \delta, s, F)$ y sin estados inaccesibles. Definiremos una relación de equivalencia en Σ^*:

$$\alpha \equiv_A \beta \qquad \text{sii} \qquad \delta^*(s, \alpha) = \delta^*(s, \beta).$$

La relación $\equiv_A$ tiene las siguientes propiedades:

1. Es una *congruencia por la derecha*:

$$\alpha \equiv_A \beta \qquad \text{implica que} \qquad \alpha a \equiv_A \beta a.$$

2. Es una *afinación* de L:

$$\alpha \equiv_A \beta \qquad \text{implica que} \qquad \alpha \in L \text{ sii } \beta \in L.$$

Relaciones de Myhill–Nerode II

3. Tiene *índice finito*, es decir, induce un número finito de clases de equivalencia.

Una relación que cumple con 1–3 es una relación de Myhill–Nerode. Sea ahora $R \subseteq \Sigma^*$ un lenguaje arbitrario. La relación de equivalencia $\equiv_R \subseteq \Sigma^* \times \Sigma^*$ se define de esta forma:

$$\alpha \equiv_R \beta \qquad \text{sii} \qquad \forall \gamma \in \Sigma^*. \alpha\gamma \in R \text{ sii } \beta\gamma \in R.$$

Teorema de Myhill–Nerode

Sea $L \subseteq \Sigma^*$. Entonces, las siguientes afirmaciones son equivalentes:

- L es regular.
- Existe una relación de Myhill–Nerode en L.
- La relación $\equiv_L$ tiene índice finito.

Ejemplo

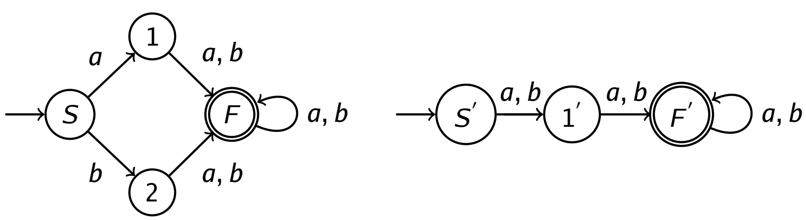
El conjunto $L = \{a^{2^n}\}$ no es regular. Se demostrará utilizando el teorema anterior. Considérense las clases de equivalencia:

$$\alpha \equiv_L \beta \qquad \text{sii} \qquad \forall k.\, \alpha a^k \in L \text{ sii } \beta a^k \in L.$$

Pero $\alpha a^k, \beta a^k \in L$ sii $|\alpha| + k = |\beta| + k = 2^n$, para alguna $n \in \mathbb{N}$. En pocas palabras, por cada valor de k hay una clase de equivalencia distinta, i.e., $\equiv_L$ no tiene índice finito.

Minimalización de estados

En ocasiones, un autómata A puede reemplazarse con un autómata A' que reconoce el mismo lenguaje pero que tiene un conjunto de estados menor:



De hecho, para cada lenguaje regular existe un único DFA (salvo isomorfismo) con un número mínimo de estados que lo reconoce.

Autómata cociente

Sea $A \in$ DFA, $A = (Q, \Sigma, \delta, s, F)$. Definimos una relación de equivalencia entre estados de Q. Sean $q, r \in Q$. Entonces:

$$q \approx r \qquad \text{sii} \qquad \forall \alpha.\, \delta^*(q, \alpha) \in F \text{ sii } \delta^*(r, \alpha) \in F.$$

Dado que $\approx$ es una relación de equivalencia, designaremos con $[q]$ la clase de equivalencia a la que pertenece el estado q. El autómata cociente de A, que denotaremos con $A/\approx = (Q_\approx, \Sigma, \delta_\approx, s_\approx, F_\approx)$, se define así:

$$\begin{aligned} Q_\approx &= \{[q] \mid q \in Q\} \\ \delta_\approx([q], a) &= [\delta(q, a)] \\ s_\approx &= [s] \\ F_\approx &= \{[f] \mid f \in F\}. \end{aligned}$$

Algoritmo de minimización I

El siguiente algoritmo permite construir el autómata cociente a partir del autómata $A = (Q, \Sigma, \delta, s, F)$, con $Q = \{q_1, \ldots, q_k\}$:

1. Se construye una tabla

Estado	q_1	$\ldots$	q_k
q_1	m	$\ldots$	
$\vdots$			
q_k			

   Donde $m = s$ si el estado en el i–ésimo renglón es final y el estado en la j–ésima columna no es final o viceversa. En caso contrario, $m = n$.
2. Se repite lo siguiente hasta que ya no haya cambios:
   Si $(q_i, q_j) = n$ y $(\delta(q_i, a), \delta(q_j, a)) = s$, para algún $a \in \Sigma$, entonces $(q_i, q_j) := s$.
3. Al terminar el paso anterior, $q_i \approx q_j$ sii $(q_i, q_j) = n$.

Algoritmo de minimización II

El algoritmo toma cuando mucho

$$\binom{n}{2} = \frac{n!}{2!(n-2)!}$$

pasos, donde n es el número de estados.

Gramáticas independientes del contexto

- Una gramática $G = \langle \Sigma, \Gamma, S, \rightarrow \rangle$ es *independiente del contexto* sii todas las reglas de producción tienen una de las dos siguientes formas
$$A \rightarrow \alpha \qquad A \rightarrow \varepsilon,$$
  donde $A \in \Gamma$.
- CFG designará el conjunto de gramáticas independientes del contexto.
- El lenguaje generado por G se denotará con $L(G)$.
- Un lenguaje L es *independiente del contexto* sii existe una gramática $G \in$ CFG tal que
$$L = L(G).$$
- CFL es el conjunto de lenguajes independientes del contexto.

Ejemplos

1. *Lenguaje de palindromas*. Sea $G_P = \langle \{a, b\}, \{S\}, S, \rightarrow_P \rangle$, donde
$$S \rightarrow_P aSa \mid bSb \mid \varepsilon.$$
2. Lenguaje de paréntesis. Sea Sea $G_D = \langle \{(,)\}, \{S\}, S, \rightarrow_D \rangle$, con las siguientes reglas
$$S \rightarrow_D (S) \mid SS \mid \varepsilon.$$

Otras definiciones I

Considérese una derivación

$$\alpha_0 \Rightarrow \alpha_1 \Rightarrow \cdots \Rightarrow \alpha_n.$$

Esta derivación presupone reglas de producción

$$A_1 \to \beta_1, \ldots, A_n \to \beta_n$$

tales que

$$\alpha_i = \gamma_i A_i \eta_i \qquad \alpha_{i+1} = \gamma_i \beta_i \eta_i.$$

Esta derivación será *por la izquierda* sii para todo $i \leq n$

$$|\gamma_i| \leq |\gamma_{i+1}| \qquad \gamma_i \in \Sigma^*$$

Otras definiciones II

y será *por la derecha* sii

$$|\eta_i| \leq |\eta_{i+1}| \qquad \eta_i \in \Sigma^*.$$

Una gramática es *no ambigua* sii $\forall \alpha \in L(G)$, α tiene exactamente una derivación por la izquierda (o por la derecha).
Un lenguaje es *no ambiguo* sii existe una gramática no ambigua que lo genere. De lo contrario, será *inherentemente ambiguo*.
Una gramática es *pulcra* sii

(a) $\forall A \in \Gamma. L(A) \neq \emptyset$;

(b) $\forall A \in \Gamma. \exists \alpha, \beta \in \Sigma^*. S \to \alpha A \beta$ o $S = A$.

Otras definiciones III

Una producción es *terminal* sii no contiene símbolos no terminales del lado derecho.
Una producción es una regla–ε sii su lado derecho es ε.
Una producción es *unitaria* sii es de la forma de la forma $A \to B$, con $A, B \in \Gamma$.
Una gramática es *propia* sii no contiene regla–ε ni reglas unitarias.

Paréntesis balanceados

La gramática G_D genera el *lenguaje de paréntesis balanceados*, es decir, $\alpha \in L(G_D)$ sii

1. $A(\alpha) = C(\alpha)$;
2. $C(\beta) \leq A(\beta) \qquad \forall \beta \in Pre(\alpha)$;

donde $A(\alpha)$ y $C(\alpha)$ son el número de paréntesis que abren y cierran que aparecen en α, respectivamente.
Demostración. Por inducción en la derivación de $S \Rightarrow \alpha$.

Lenguajes de Dyck

El lenguaje de paréntesis se puede generalizar a los lenguajes de Dyck.
Sea A un alfabeto de símbolos terminales y sea $\bar{A}$ una copia de A, es decir:

(i) $A \cap \bar{A} = \emptyset$

(ii) existe una biyección de A a $\bar{A}$.

Por ejemplo, si $A = \{(\}$, entonces $\bar{A} = \{)\}$.
El lenguaje de Dyck sobre A es el lenguaje generado por las reglas de producción:

$$S \to a_1 S \bar{a}_1 \mid \ldots \mid a_n S \bar{a}_n \mid SS \mid \varepsilon,$$

donde $A = \{a_1, \ldots, a_n\}$ y $\bar{a}_i \in \bar{A}$ es la copia de a_i. Este lenguaje se denotará con D_A^*.
Si $|A| = n$, una notación alternativa es D_n^*.

Formas normales de Chomsky y de Greibach

Sea $G = \langle \Sigma, \Gamma, S, \to \rangle$ un gramática. G tiene *forma normal de Chomsky* (CNF) sii todas las producciones tiene la forma

$$A \to BC \qquad A \to a,$$

donde $A, B, C \in \Gamma$ y $a \in \Sigma$.

En cambio, G tiene *forma normal de Greibach* (GNF) sii todas las producciones tienen la forma

$$A \to bB_1 \cdots B_k$$

con $0 \leq k$, $b \in \Sigma$ y $B_1, \ldots, B_k \in \Gamma$.

Ejemplos

Los conjuntos de producciones siguientes tienen forma normal de Chomsky y de Greibach (respectivamente)

$$S \to AB \mid AC \mid SS \qquad C \to SB \qquad A \to (\qquad B \to)$$
$$S \to (B \mid (SB \mid (BS \mid (SBS \qquad B \to)$$

y generan el lenguaje de paréntesis balanceados.

Las producciones siguientes generan el lenguaje de palindromas

$$S \to AA \mid BB \mid AC \mid BD \qquad C \to SA \qquad D \to SB \qquad A \to a \qquad B \to b$$
$$S \to aA \mid aSA \mid bB \mid bSB \qquad A \to a \qquad B \to b$$

Con excepción en ambos casos de ε.

Todas las CFGtienen equivalente en CNF o GNF

Teorema. Sea $G \in$ CFG. Entonces existen $G_C, G_G \in$ CFG tales que tienen forma normal de Chomsky y de Greibach, respectivamente, y

$$L(G_C) = L(G_G) = L(G) - \{\varepsilon\}.$$

Lema de gramáticas propias I

Sea $G \in$ CFG. Entonces, existe una gramática propia G_P tal que

$$L(G_P) = L(G) - \{\varepsilon\}.$$

Demostración. Sea $\to'$ la relación mínima que

(i) contiene a $\to$;

(ii) si $A \to' \alpha B \beta$ y $B \to \varepsilon$, entonces $A \to' \alpha\beta$

(iii) si $A \to' B$ y $B \to' \gamma$, entonces $A \to' \gamma$.

Lema de gramáticas propias II

Y sea $G' \in$ CFG como G, pero con las reglas de producción $\to'$. Entonces

$$L(G) = L(G')$$

Obviamente, $L(G) \subseteq L(G')$. Para $L(G') \subseteq L(G)$, baste observar que toda derivación en G' se puede realizar en G, salvo tal vez en más pasos. Sea ahora G_P como G' pero con la relación $\to''$ que prescinde de las producciones unitarias y $-\varepsilon$ que existan en $\to'$. Entonces

$$L(G_P) = L(G') - \{\varepsilon\} = L(G) - \{\varepsilon\}.$$

Esta última afirmación se demuestra considerando que las derivaciones de longitud mínima en G' prescinden de producciones$-\varepsilon$ y unitarias.

Transformación a forma normal de Chomsky

Una vez que contamos con una gramática propia G_P, podemos construir la CNF:

1. Para cada $a \in \Sigma$, introducimos nuevos no terminales A_a y reglas de producción $A_a \to a$.
2. Sustituimos los terminales por los no terminales del punto anterior en todas las reglas de G_P que no tengan CNF todavía.
3. Dada una regla $A \to B_1 \ldots B_k$ ($k > 2$), introducimos un nuevo no terminal C_1 y las reglas

$$A \to B_1 C_1 \qquad C_1 \to B_2 \ldots B_k.$$

4. Repetimos este procedimiento hasta tener sólo reglas con dos no terminales del lado derecho.
5. Eliminamos las reglas del punto anterior que no tengan CNF.

Transformación a forma normal de Greibach

Sea G_C una gramática en CNF. Para convertirla a GNF:

1. Para cada $X \in \Gamma$, considérese el lenguaje

$$R_{X,x} = \{\beta \in \Gamma^* \mid X \Rightarrow_L x\beta\}.$$

   Este lenguaje es regular. Sea $G_{X,x}$ una gramática lineal por la derecha que lo genere. Sea $S_{X,x}$ el símbolo inicial de esta gramática.
2. Entonces, sustituimos las reglas

$$A \to XY$$

   por reglas

$$A \to x S_{X,x} Y,$$

   y agregamos las producciones y símbolos no terminales de $G_{X,x}$ a nuestra gramática.
3. Eliminamos las reglas$-\varepsilon$ y las reglas unitarias.

Ejemplos I

Lenguaje de paréntesis. Empezamos con forma normal de Chomsky:

$$S \to AB \mid AC \mid SS \qquad C \to SB \qquad A \to (\qquad B \to).$$

Tenemos ahora los siguientes lenguajes $R_{X,x}$

$$\begin{aligned} R_{S,(} &= (B+C)S^* \\ R_{C,(} &= (B+C)S^*B \\ R_{A,(} &= \{\varepsilon\} \\ R_{B,)} &= \{\varepsilon\} \end{aligned}$$

Estas producciones nos generan los lenguajes anteriores:

$$\begin{array}{lll} S_{S,(} \to BX \mid CX & X \to SX \mid \varepsilon & \\ S_{C,(} \to BY \mid CY & Y \to SY \mid BZ & Z \to \varepsilon \\ S_{A,(} \to \varepsilon & & \\ S_{B,)} \to \varepsilon & & \end{array}$$

Ejemplos II

Entonces, tenemos las reglas nuevas

$$\begin{array}{lll} S \to (S_{A,(}B \mid (S_{A,(}C \mid (S_{S,(}S & C \to (S_{S,(}B & A \to (\\ S_{S,(} \to)S_{B,)}X \mid (S_{C,(}X & X \to (S_{S,(}X \mid \varepsilon & B \to) \\ S_{C,(} \to)S_{B,)}Y \mid (S_{C,(}Y & Y \to (S_{S,(}Y \mid)S_{B,)}Z & Z \to \varepsilon \\ S_{A,(} \to \varepsilon & S_{B,)} \to \varepsilon & \end{array}$$

Finalmente, eliminamos reglas$-\varepsilon$ y símbolos superfluos:

$$\begin{array}{lll} S \to (B \mid (C \mid (S_{S,(}S & C \to (S_{S,(}B & A \to (\\ S_{S,(} \to)X \mid (S_{C,(}X \mid (S_{C,(} & X \to (S_{S,(}X \mid (S_{S,(} & B \to) \\ S_{C,(} \to)Y \mid (S_{C,(}Y & Y \to (S_{S,(}Y \mid) & \end{array}$$

Teorema del bombeo I

Sea $L \in$ CFL. Entonces, existe $k \in \mathbb{N}$ tal que para toda $\alpha \in L$, si $k \le |\alpha|$, entonces

$$\alpha = \beta\gamma\eta\theta\varphi, \qquad \gamma\theta \neq \varepsilon \qquad \text{y} \qquad |\gamma\eta\theta| \le k$$

y para toda $i \in \mathbb{N}$

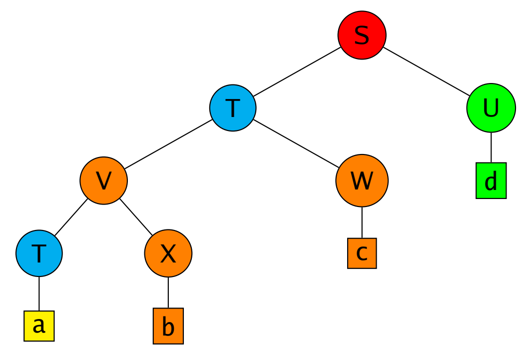
$$\beta\gamma^i\eta\theta^i\varphi \in L.$$

Demostración.

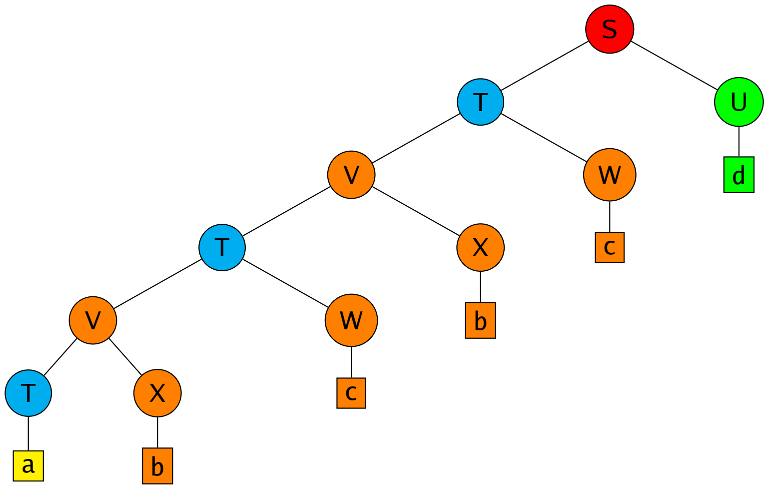
- Sea $G \in$ CFG en CNF y $L(G) = L - \varepsilon$.
- Sea Γ el alfabeto de símbolos no terminales de G, con $|\Gamma| = n$ y sea $k = 2^{n+1}$.
- Entonces, el árbol sintáctico de toda cadena de longitud igual o superior a k debe tener profundidad de $n+1$ al menos.
- Entonces en un camino de longitud máxima de la raíz a las hojas debe aparecer dos veces, al menos, el mismo no terminal.
- Este no terminal puede usarse para definir las cadenas γ y θ.

Ejemplo

Considérense los siguientes árboles de derivación:



Árbol de la cadena $\alpha = abcd$



Árbol de la cadena $\beta\gamma^2\eta\theta^2\varphi = abcbcd$
con $\beta = a$, $\gamma = bc$, $\eta = \theta = \varepsilon$ y $\varphi = d$

Aplicaciones

Como con los lenguajes regulares, este teorema se puede utilizar para demostrar que un lenguaje dado no es independiente del contexto. Ejemplos:

$$\{a^n b^n c^n\},$$

por una aplicación directa del teorema, y

$$\{\alpha\alpha\}$$

pues los CFL y los regulares son cerrados bajo intersección y

$$\{a^n b^m a^n b^m\} = \{\alpha\alpha\} \cap L(a^* b^* a^* b^*)$$

y, obviamente, $\{a^n b^m a^n b^m\}$ no es independiente del contexto

Autómatas de pila

Un autómata de pila no determinista A está formado por

- un conjunto de estados Q;
- un alfabeto de entrada Σ;
- un alfabeto de pila Γ;
- una relación de transición $\delta \subseteq (Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma) \times (Q \times \Gamma^*)$
- un estado inicial s;
- un símbolo inicial de la pila $\perp$;
- un subconjunto de estados finales $F \subseteq Q$.

Llamaremos NPDA a los autómatas de pila no deterministas.

Lenguajes aceptados por NPDA

Una *configuración* de un autómata $A \in$ NPDA es una terna (q, α, β) donde

$$\begin{array}{ccc} q & \in & Q \\ \alpha & \in & \Sigma^* \\ \beta & \in & \Gamma^* \end{array}$$

La configuración inicial es $(s, \alpha, \perp)$. Definiremos ahora una relación entre configuraciones:

$$\begin{array}{lll} \text{si } \delta((q, a, B), (r, \eta)) & \text{entonces} & (q, a\alpha, B\beta) \Rightarrow (r, \alpha, \eta\beta) \\ \text{si } \delta((q, \varepsilon, B), (r, \eta)) & \text{entonces} & (q, \alpha, B\beta) \Rightarrow (r, \alpha, \eta\beta) \end{array}$$

El lenguaje aceptado por un $A \in$ NPDA es

$$L(A) = \{\alpha \in \Sigma^* \mid \exists f \in F.(s, \alpha, \perp) \Rightarrow^* (f, \varepsilon, \beta)\}$$

Aceptación por pila vacía

Una definición alternativa del lenguaje aceptado por un $A \in$ NPDA es

$$L(A) = \{\alpha \in \Sigma^* \mid (s, \alpha, \perp) \Rightarrow^* (q, \varepsilon, \varepsilon)\}$$

Ambas definiciones son equivalentes y puede construirse un $A' \in$ NPDA que acepte por pila vacía el mismo lenguaje que un $A \in$ NPDA que acepte por estado final (y viceversa).

Teorema de equivalencia entre CFG y NPDA I

(a) Sea $G \in$ CFG. Entonces, existe $A \in$ NPDA tal que

$$L(G) = L(A).$$

(b) Sea $A \in$ NPDA. Entonces, existe $G \in$ CFG tal que

$$L(G) = L(A).$$

Demostración
(a) Sea $G = \langle \Sigma, \Gamma, S \rightarrow \rangle$ en FNG. Sea

$$A = (\{q\}, \Sigma, \Gamma, \delta, q, S, q),$$

donde

$$\delta((q, a, A), (q, B_1 \cdots B_k)) \qquad \text{sii} \qquad A \rightarrow aB_1 \cdots B_k.$$

Teorema de equivalencia entre CFG y NPDA II

Si puede demostrar por inducción en la longitud de las derivaciones (por la izquierda) que

$$(q, \alpha\beta, A) \Rightarrow^n_A (q, \beta, \gamma) \qquad \text{sii} \qquad A \Rightarrow^n_G \alpha\gamma.$$

Entonces

$$\begin{aligned} \alpha \in L(G) \quad & \text{sii} \quad S \Rightarrow^*_G \alpha \\ & \text{sii} \quad (q, \alpha, S) \Rightarrow^*_A (q, \varepsilon, \varepsilon) \\ & \text{sii} \quad \alpha \in L(A). \end{aligned}$$

(b) Se demuestra primero que para todo $A \in$ NPDA existe un $A' \in$ NPDA con un solo estado tal que

$$L(A) = L(A')$$

y una vez construido A', los argumentos del caso (a) se pueden aplicar en sentido inverso.

Determinismo

- A diferencia de REG, la clase CFL no determinista es estrictamente mayor que la determinista (DCFL).
- Un autómata determinista de pila $D = (Q, \Sigma, \Gamma, \delta, \perp, \dashv, s, F)$ incluye el símbolo especial $\dashv$ que sirve para indicar el final de la cadena de entrada.
- Las transiciones están indicadas por la función

$$\delta : Q \times \Sigma \cup \{\varepsilon, \dashv\} \rightarrow Q \times \Gamma^*.$$

- δ no puede eliminar de la pila a $\perp$, es decir

$$\delta(q, a, \perp) = (r, \alpha\perp).$$

- La aceptación es por estados finales. La aceptación por pila vacía define un conjunto de lenguajes diferente.
- Un ejemplo de un lenguaje en CFL pero no en DCFL es

$$\{a, b\}^* - \{\alpha\alpha \mid \alpha \in \{a, b\}^*\}$$

Algoritmo de Cocke, Kasami y Younger

- Es un algoritmo para *decidir* de manera eficiente si una cadena pertenece a un CFL.
- Sea L un CFL y sea $G = \langle \Sigma, \Gamma, S, \to \rangle$ una CFG en forma normal de Chomsky tal que
$$L(G) = L.$$
- Sea $\alpha \in L$ con longitud n. Sean $0 \leq i < j \leq n$ y sea
$$\alpha_{i,j}$$
la subcadena de α que va de la posición $i + 1$ a la j.
- Sea $T_{i,j} \subseteq \Gamma$ el conjunto de no terminales que generaron $\alpha_{i,j}$.
- El algoritmo calcula el valor de todos los $T_{i,j}$ de manera inductiva. En particular, $S \in T_{0,n}$ si $S \Rightarrow_G^* \alpha$.
- El algoritmo es $O(pn^3)$, p el número de producciones en G.

for $i := 0$ **to** $n - 1$ **do**
  $T_{i,i+1} := \emptyset$;
  for all $A \to a \in G$ **do**
    if $a = \alpha_{i,i+1}$ **then**
      $T_{i,i+1} := T_{i,i+1} \cup \{A\}$
for $m := 2$ **to** n **do**
  for $i := 0$ **to** $n - m$ **do**
    $T_{i,i+m} := \emptyset$;
    for $j := i + 1$ **to** $i + m - 1$ **do**
      for all $A \to BC \in G$ **do**
        if $B \in T_{i,j} \wedge C \in T_{j,i+m}$ **then**
          $T_{i,i+m} := T_{i,i+m} \cup \{A\}$

Un ejemplo I

Reconocimiento de *abba* en el lenguaje de palindromas (p. 23).
Inicialmente agregamos al menos un no terminal a los $T_{i,i+1}$

$$T_{0,1} = \{A\} \qquad T_{1,2} = \{B\} \qquad T_{2,3} = \{B\} \qquad T_{3,4} = \{A\}.$$

Después consideramos los segmentos de longitud 2.
$S \in T_{1,3}$ porque $B \in T_{1,2}$ y $B \in T_{2,3}$ y por la regla $S \to BB$

$$T_{0,2} = \emptyset \qquad T_{1,3} = \{S\} \qquad T_{2,4} = \emptyset$$

Hay más de una forma de construir los segmentos de longitud 3. En particular,

$$\alpha_{1,4} = \alpha_{1,2} \cdot \alpha_{2,4} = \alpha_{1,3} \cdot \alpha_{3,4}.$$

La última es el que nos da C con la regla $C \to SA$

$$T_{0,3} = \emptyset \qquad T_{1,4} = \{C\}.$$

Un ejemplo II

Finalmente, $S \to AC$ y $T_{0,1}$ y $T_{1,4}$ nos dan

$$T_{0,4} = \{S\}.$$

Resultado: aceptación.

Reconocimiento de *bba*.

$$T_{0,1} = \{B\} \qquad T_{1,2} = \{B\} \qquad T_{2,3} = \{A\}$$
$$T_{0,2} = \{S\} \qquad T_{1,3} = \emptyset$$
$$T_{0,3} = \{C\}.$$

Resultado: rechazo, pues $S \notin T_{0,3}$.