GRAMÁTICA DE CONTEXTO LIBRE

Gramática de contexto libre $G = (V_N, V_T, P, S)$ que genera oraciones copulativas:

- $V_N = \{ <\textit{oración}>, <\text{sujeto}>, <\text{verbo}>, <\text{atributo}>, <\text{adjetivo}> \}$
- $V_T = \{$***el, la, hombre, niña, es está, parece, alto, bella, inteligente***$\}$
- $S \in V_N$
- Conjunto de producciones:

$$
\begin{array}{rcl}
P & = & \{ \\
 & (1) & < \text{oración} > \longrightarrow < \text{sujeto} >< \text{verbo} >< \text{atributo} > \\
 & (2) & < \text{sujeto} > \longrightarrow < \text{artículo} >< \text{nombre} > \\
 & (3) & < \text{artículo} > \longrightarrow \boldsymbol{el} \\
 & (4) & < \text{artículo} > \longrightarrow \boldsymbol{la} \\
 & (5) & < \text{nombre} > \longrightarrow \boldsymbol{hombre} \\
 & (6) & < \text{nombre} > \longrightarrow \boldsymbol{niña} \\
 & (7) & < \text{verbo} > \longrightarrow \boldsymbol{es} \\
 & (8) & < \text{verbo} > \longrightarrow \boldsymbol{está} \\
 & (9) & < \text{verbo} > \longrightarrow \boldsymbol{parece} \\
 & (10) & < \text{atributo} > \longrightarrow < \text{adjetivo} > \\
 & (11) & < \text{adjetivo} > \longrightarrow \boldsymbol{alto} \\
 & (12) & < \text{adjetivo} > \longrightarrow \boldsymbol{bella} \\
 & (13) & < \text{adjetivo} > \longrightarrow \boldsymbol{inteligente} \\
 & & \}
\end{array}
$$

Se pueden agrupar las reglas que tienen la misma parte izquierda:

$$
\begin{array}{rcl}
< \text{artículo} > & \longrightarrow & \boldsymbol{el} \mid \boldsymbol{la} \mid \boldsymbol{un} \mid \boldsymbol{una} \\
< \text{nombre} > & \longrightarrow & \boldsymbol{hombre} \mid \boldsymbol{niña} \\
< \text{verbo} > & \longrightarrow & \boldsymbol{es} \mid \boldsymbol{está} \mid \boldsymbol{parece} \\
< \text{adjetivo} > & \longrightarrow & \boldsymbol{alto} \mid \boldsymbol{bella} \mid \boldsymbol{inteligente} \mid \boldsymbol{amable}
\end{array}
$$

DERIVACIONES

Derivación de una oración

$$\begin{aligned}
<\text{oración}> \; & \underset{1}{\Rightarrow} \; <\text{sujeto}><\text{verbo}><\text{atributo}> \\
& \underset{2}{\Rightarrow} \; <\text{artículo}><\text{nombre}><\text{verbo}><\text{atributo}> \\
& \underset{3}{\Rightarrow} \; \textbf{\textit{el}} <\text{nombre}><\text{verbo}><\text{atributo}> \\
& \underset{5}{\Rightarrow} \; \textbf{\textit{el hombre}} <\text{verbo}><\text{atributo}> \\
& \underset{7}{\Rightarrow} \; \textbf{\textit{el hombre es}} <\text{atributo}> \\
& \underset{10}{\Rightarrow} \; \textbf{\textit{el hombre es}} <\text{adjetivo}> \\
& \underset{11}{\Rightarrow} \; \textbf{\textit{el hombre es alto}}
\end{aligned}$$

o abreviadamente

$$<\text{oración}> \; \overset{+}{\Rightarrow} \; \textbf{\textit{el hombre es alto}}$$

Derivación de una oración que es semánticamente incorrecta

$$\begin{aligned}
<\text{oración}> \; & \underset{1}{\Rightarrow} \; <\text{sujeto}><\text{verbo}><\text{atributo}> \\
& \underset{2}{\Rightarrow} \; <\text{artículo}><\text{nombre}><\text{verbo}><\text{atributo}> \\
& \underset{4}{\Rightarrow} \; \textbf{\textit{la}} <\text{nombre}><\text{verbo}><\text{atributo}> \\
& \underset{5}{\Rightarrow} \; \textbf{\textit{la hombre}} <\text{verbo}><\text{atributo}> \\
& \underset{9}{\Rightarrow} \; \textbf{\textit{la hombre parece}} <\text{atributo}> \\
& \underset{10}{\Rightarrow} \; \textbf{\textit{la hombre parece}} <\text{adjetivo}> \\
& \underset{12}{\Rightarrow} \; \textbf{\textit{la hombre parece bella}}
\end{aligned}$$

APLICACIONES DE LAS GRAMÁTICAS DE CONTEXTO LIBRE

Las gramáticas de contexto libre son utilizadas para establecer las reglas sintácticas de los lenguajes de programación.

Gramática que genera asignaciones de expresiones aritméticas.

$$
\begin{array}{rcl}
P & = & \{ \\
& (1) & <\text{asignación}> \longrightarrow \boldsymbol{identificador} = <\textit{expresión}> \\
& (2) & <\text{expresión}> \longrightarrow <\textit{expresión}> \ + \ <\textit{sumando}> \\
& (3) & <\text{expresión}> \longrightarrow <\textit{sumando}> \\
& (4) & <\textit{sumando}> \longrightarrow <\textit{sumando}> \ * \ <factor> \\
& (5) & <\textit{sumando}> \longrightarrow <factor> \\
& (6) & <factor> \longrightarrow \boldsymbol{número} \\
& (7) & <factor> \longrightarrow \boldsymbol{identificador} \\
& (8) & <factor> \longrightarrow (<\boldsymbol{expresión}>) \\
& & \}
\end{array}
$$

Haciendo uso de esta gramática, se puede generar una sentencia de asignación como la siguiente:

$$
\begin{array}{rl}
<\textit{asignación}> & \underset{1}{\Rightarrow} \ \boldsymbol{identificador} \ = \ <\text{expresión}> \\
& \underset{2}{\Rightarrow} \ \boldsymbol{identificador} \ = \ <\text{expresión}> \ + \ <\textit{sumando}> \\
& \underset{3}{\Rightarrow} \ \boldsymbol{identificador} \ = \ <\textit{sumando}> \ + \ <\textit{sumando}> \\
& \underset{4}{\Rightarrow} \ \boldsymbol{identificador} \ = \ <\textit{sumando}> \ * \ <factor> \ + \ <\textit{sumando}> \\
& \underset{5}{\Rightarrow} \ \boldsymbol{identificador} \ = \ <factor> \ * \ <factor> \ + \ <\textit{sumando}> \\
& \underset{6}{\Rightarrow} \ \boldsymbol{identificador} \ = \ \boldsymbol{número} * \ <factor> \ + \ <\textit{sumando}> \\
& \underset{7}{\Rightarrow} \ \boldsymbol{identificador} \ = \ \boldsymbol{número} \ * \ \boldsymbol{identificador} + \ <\textit{sumando}> \\
& \underset{5}{\Rightarrow} \ \boldsymbol{identificador} \ = \ \boldsymbol{número} \ * \ \boldsymbol{identificador} + \ <factor> \\
& \underset{6}{\Rightarrow} \ \boldsymbol{identificador} \ = \ \boldsymbol{número} \ * \ \boldsymbol{identificador} \ + \ \boldsymbol{número}
\end{array}
$$

o abreviadamente

$$<\textit{asignación}> \overset{+}{\Rightarrow} \boldsymbol{identificador} \ = \ \boldsymbol{número} \ * \ \boldsymbol{identificador} \ + \ \boldsymbol{número}$$

DEFINICIÓN FORMAL DE GRAMÁTICA DE CONTEXTO LIBRE

Una ***gramática de contexto libre*** G se define como $G = (V_N, V_T, P, S)$ donde

- V_N es un conjunto finito de símbolos que se denomina ***alfabeto no terminal*** y también puede ser denotado por Σ_N o N.
- V_T es un conjunto finito de símbolos que se denomina ***alfabeto terminal*** y también puede ser denotado por Σ_T o T,
- $S \in V_N$ y es el ***axioma*** o ***símbolo inicial o distinguido*** de la gramática y
- P es el ***conjunto de reglas de reescritura o de producción***.

Se verifica que

$$V_N \cup V_T = V$$
$$V_N \cap V_T = \emptyset$$

V es el ***alfabeto o vocabulario*** de la gramática y también suele ser denotado por Σ.

El conjunto de producciones P se define como

$$P = \{(A, \alpha) | A \in V_N, \alpha \in V^* = (V_N \cup V_T)^*\}$$

El par (A, α) suele ser denotado por

$$A \rightarrow \alpha$$

siendo A la parte izquierda de la regla de producción y α la parte derecha.

EJEMPLO: PALÍNDROMO IMPAR

Una gramática "formal" establece las reglas que permiten generar palabras de un lenguaje.

Sea G una gramática de contexto libre compuesta por el siguiente conjunto de reglas de producción:

$$
\begin{array}{rcl}
P & = & \{ \\
 & (1) & S \longrightarrow aAa \\
 & (2) & A \longrightarrow aAa \\
 & (3) & A \longrightarrow bBb \\
 & (4) & B \longrightarrow bBb \\
 & (5) & B \longrightarrow c \\
 & & \}
\end{array}
$$

El lenguaje que genera la gramatica G puede ser expresado como:

$$
\begin{aligned}
L(G) &= \{a^i b^j c b^j a^i | i, j \geq 1\} \\
&= \{abcba, abcba, abbcbba, \ldots, aacaa, aabbcbbaa, \ldots\}
\end{aligned}
$$

Este lenguaje se denomina "palíndromo impar" porque cada una de las palabras del lenguaje se puede leer igual de izquierda a derecha que de derecha a izquierda y tiene un elemento central que divide a la palabra.

La derivación que genera la palabra *aabbcbbaa* es la siguiente:

$$
\begin{array}{rcl}
S & \underset{1}{\Rightarrow} & aAa \\
 & \underset{2}{\Rightarrow} & aaAaa \\
 & \underset{3}{\Rightarrow} & aabBbaa \\
 & \underset{4}{\Rightarrow} & aabbBbbaa \\
 & \underset{5}{\Rightarrow} & aabbcbbaa
\end{array}
$$

CONVENIOS DE NOTACIÓN DE LAS GRAMÁTICAS FORMALES

1. Símbolos terminales, es decir, pertenecientes a V_T:
   - Primeras letras minúsculas del alfabeto latino: $\{a,\ b,\ c,\ \ldots\}$
   - Operadores ariméticos, lógicos, relacionales: $\{+,\ -,\ *,\ /,\ \vee,\ \wedge,\ <,\ >,\ \leq,\ \ldots\}$
   - Cifras numéricas: $\{0,\ 1,\ \ldots,\ 9\}$
   - Símbolos especiales: $\{[,\ ],\ (,\ ),\ ;,\ \ldots\}$
   - Palabras en negrita: {***if, else, entonces, para,*** ...}
2. Símbolos no terminales, es decir, pertenecientes a V_N:
   - Primeras letras mayúsculas del alfabeto latino: $\{A,\ B,\ C,\ \ldots\ \}$
   - Palabras en *cursiva* o encerradas entre “$<$” y “$>$”: {*término*, <término>, ...}
   - La parte izquierda de la primera producción será el símbolo inicial, salvo que se indique lo contrario. Generalmente, el símbolo inicial será el símbolo S.
3. Símbolos gramáticales terminales y no terminales, es decir, pertenecientes a $V = V_N \cup V_T$:
   - Últimas letras mayúsculas del alfabeto latino, salvo la letra S: $\{\ldots,\ X,\ Y,\ Z\ \}$
4. Cadenas de símbolos gramáticales terminales (pertenecientes a V_T^*)
   - Últimas letras minúsculas del alfabeto latino:$\{\ldots\ x,\ y,\ z\ \}$
5. Los símbolos ϵ y λ suelen denotar a la palabra vacía.
6. Cadenas de símbolos gramáticales, es decir, compuestas por símbolos terminales y no terminales (pertenecientes a $V^* = (V_N \cup V_T)^*$.
   - Letras minúsculas del alfabeto griego, excepto ϵ y λ :$\{\alpha, \beta, \gamma \ldots\}$
7. Agrupamiento de reglas: si existen dos o más reglas que tienen la misma parte izquierda, como, por ejemplo:

$$\begin{array}{rcl} A & \longrightarrow & \alpha_1 \\ A & \longrightarrow & \alpha_2 \\ & \cdots & \\ A & \longrightarrow & \alpha_n \end{array}$$

   entonces se pueden agrupar como

$$A \longrightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$$

EJEMPLO: EXPRESIONES ARITMÉTICAS

Gramática de contexto libre $G = (V_N, V_T, P, S)$ que genera expresiones aritméticas:

- $V_N = \{S, E\}$,
- $V_T = \{\boldsymbol{identificador}, =, +, *, (,), \boldsymbol{n\acute{u}mero}\}$,
- $S \in V_N$
- y el conjunto de producciones es:

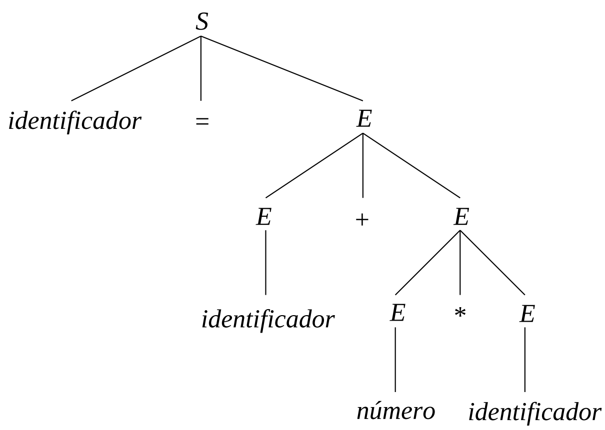
$$
\begin{array}{rcl}
P & = & \{ \\
 & (1) & S \longrightarrow \boldsymbol{identificador} \;=\; E \\
 & (2) & E \longrightarrow E \;+\; E \\
 & (3) & E \longrightarrow E \;*\; E \\
 & (4) & E \longrightarrow (\; E \;) \\
 & (5) & E \longrightarrow \boldsymbol{identificador} \\
 & (6) & E \longrightarrow \boldsymbol{n\acute{u}mero} \\
 & & \}
\end{array}
$$

EJEMPLO: EXPRESIONES ARITMÉTICAS

Derivación de una expresión aritmética

$$
\begin{aligned}
S &\underset{1}{\Rightarrow} \boldsymbol{identificador} = E \\
&\underset{2}{\Rightarrow} \boldsymbol{identificador} = E + E \\
&\underset{3}{\Rightarrow} \boldsymbol{identificador} = E + E * E \\
&\underset{6}{\Rightarrow} \boldsymbol{identificador} = E + \boldsymbol{n\acute{u}mero} * E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = E + \boldsymbol{n\acute{u}mero} * \boldsymbol{identificador} \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + \boldsymbol{n\acute{u}mero} * \boldsymbol{identificador}
\end{aligned}
$$

Árbol sintáctico asociado a la derivación.



DERIVACIONES POR LA IZQUIERDA Y POR LA DERECHA

Derivación por la izquierda:

$$
\begin{aligned}
S &\underset{1}{\Rightarrow} \boldsymbol{identificador} = E \\
&\underset{2}{\Rightarrow} \boldsymbol{identificador} = E + E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + E \\
&\underset{3}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + E * E \\
&\underset{6}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + \boldsymbol{n\acute{u}mero} * E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + \boldsymbol{n\acute{u}mero} * \boldsymbol{identificador}
\end{aligned}
$$

Derivación por la derecha:

$$
\begin{aligned}
S &\underset{1}{\Rightarrow} \boldsymbol{identificador} = E \\
&\underset{2}{\Rightarrow} \boldsymbol{identificador} = E + E \\
&\underset{3}{\Rightarrow} \boldsymbol{identificador} = E + E * E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = E + E * \boldsymbol{identificador} \\
&\underset{6}{\Rightarrow} \boldsymbol{identificador} = E + \boldsymbol{n\acute{u}mero} * \boldsymbol{identificador} \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + \boldsymbol{n\acute{u}mero} * \boldsymbol{identificador}
\end{aligned}
$$

AMBIGÜEDAD DE LAS GRAMÁTICAS

La gramática anterior es ambigua porque puede generar la cadena

identificador = identificador + número * identificador

mediante dos derivaciones por la izquierda diferentes.

Primera derivación por la izquierda:

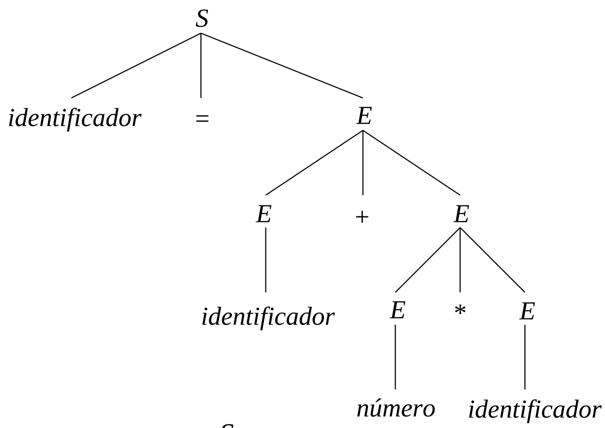
$$
\begin{aligned}
S &\underset{1}{\Rightarrow} \boldsymbol{identificador} = E \\
&\underset{2}{\Rightarrow} \boldsymbol{identificador} = E + E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + E \\
&\underset{3}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + E * E \\
&\underset{6}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + \boldsymbol{número} * E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + \boldsymbol{número} * \boldsymbol{identificador}
\end{aligned}
$$

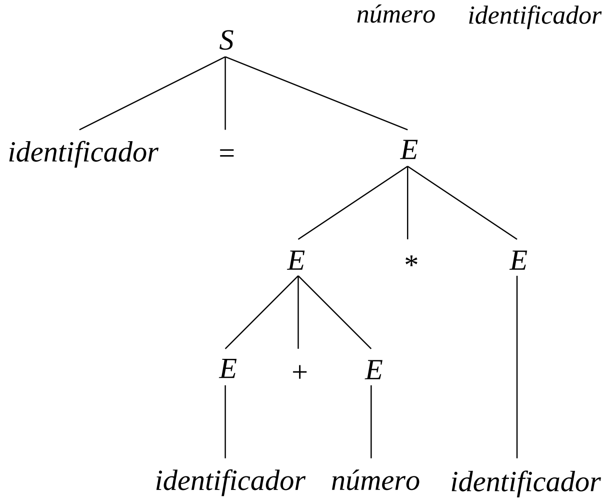
Segunda derivación por la izquierda:

$$
\begin{aligned}
S &\underset{1}{\Rightarrow} \boldsymbol{identificador} = E \\
&\underset{3}{\Rightarrow} \boldsymbol{identificador} = E * E \\
&\underset{2}{\Rightarrow} \boldsymbol{identificador} = E + E * E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + E * E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + \boldsymbol{número} * E \\
&\underset{5}{\Rightarrow} \boldsymbol{identificador} = \boldsymbol{identificador} + \boldsymbol{número} * \boldsymbol{identificador}
\end{aligned}
$$

AMBIGÜEDAD DE LAS GRAMÁTICAS

Árboles de derivación diferentes para una misma cadena.





EJEMPLO: GRAMÁTICA NO AMBIGUA

Conjunto de reglas de producción de una gramática **no ambigua** que puede generar expresiones aritméticas

$$
\begin{array}{rcl}
P & = & \{ \\
 & (1) & S \longrightarrow \boldsymbol{identificador} \ = \ E \\
 & (2) & E \longrightarrow E \ + \ T \\
 & (3) & E \longrightarrow T \\
 & (4) & T \longrightarrow T \ * \ F \\
 & (5) & T \longrightarrow F \\
 & (6) & F \longrightarrow (\ E \ ) \\
 & (7) & F \longrightarrow \boldsymbol{identificador} \\
 & (8) & F \longrightarrow \boldsymbol{n\acute{u}mero} \\
 & & \}
\end{array}
$$

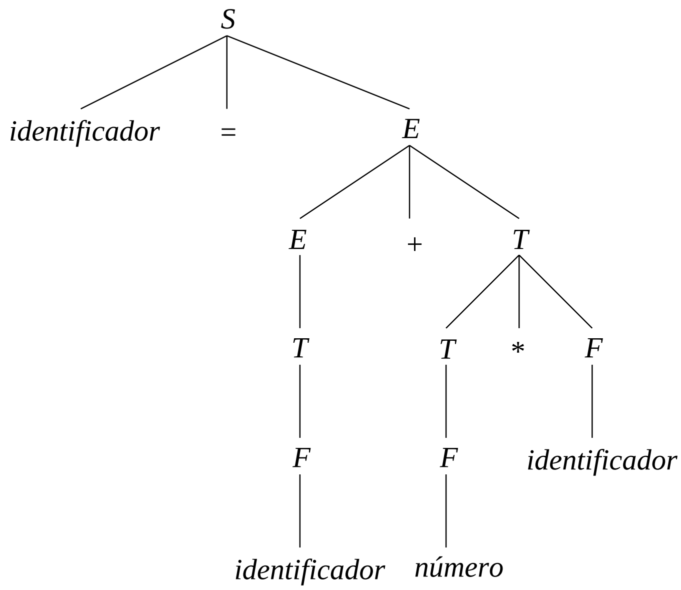
La derivación por la izquierda de la cadena

identificador = identificador + número * identificador

utilizando esta gramática es la siguiente:

$$
\begin{array}{lcl}
S & \underset{1}{\Rightarrow} & \boldsymbol{identificador} \ = \ E \\
 & \underset{2}{\Rightarrow} & \boldsymbol{identificador} \ = \ E \ + \ T \\
 & \underset{3}{\Rightarrow} & \boldsymbol{identificador} \ = \ T \ + \ T \\
 & \underset{5}{\Rightarrow} & \boldsymbol{identificador} \ = \ F \ + \ T \\
 & \underset{7}{\Rightarrow} & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ T \\
 & \underset{4}{\Rightarrow} & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ T \ * \ F \\
 & \underset{5}{\Rightarrow} & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ F \ * \ F \\
 & \underset{8}{\Rightarrow} & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ \boldsymbol{n\acute{u}mero} \ * \ F \\
 & \underset{7}{\Rightarrow} & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ \boldsymbol{n\acute{u}mero} \ * \ \boldsymbol{identificador}
\end{array}
$$

EJEMPLO: GRAMÁTICA NO AMBIGUA



EL PROBLEMA DEL "ELSE DANZANTE"

$$
\begin{array}{llll}
 & & \dots & \\
(1) & S & \longrightarrow & \textbf{\textit{if}}\ C\ S \\
(2) & S & \longrightarrow & \textbf{\textit{if}}\ C\ S\ \textbf{\textit{else}}\ S \\
(3) & S & \longrightarrow & I \\
 & & \dots &
\end{array}
$$

donde S genera sentencias de control, C genera expresiones condicionales e I genera otras sentencias, por ejemplo, de asignación.

Esta gramática es ambigua porque la sentencia

$$\textbf{\textit{if}}\ C\ \textbf{\textit{if}}\ C\ S\ \textbf{\textit{else}}\ S$$

puede ser generada mediante dos derivaciones que tienen dos árboles sintácticos diferentes:

1. Primera derivación:

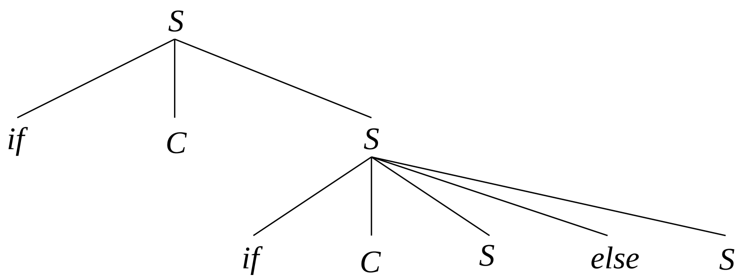
$$
\begin{array}{lll}
S & \underset{1}{\Rightarrow} & \textbf{\textit{if}}\ C\ \underline{S} \\
 & \underset{2}{\Rightarrow} & \textbf{\textit{if}}\ C\ \underline{\textbf{\textit{if}}\ C\ S\ \textbf{\textit{else}}\ S}
\end{array}
$$

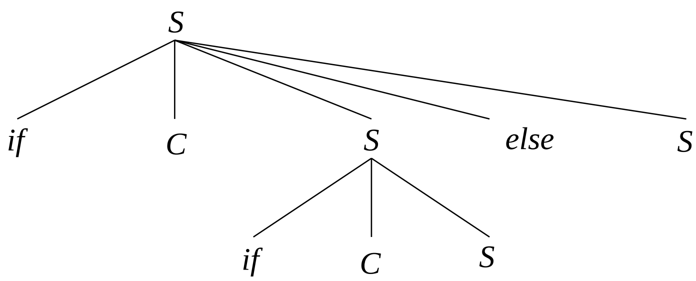
2. Segunda derivación:

$$
\begin{array}{lll}
S & \underset{2}{\Rightarrow} & \textbf{\textit{if}}\ C\ \underline{S}\ \textbf{\textit{else}}\ S \\
 & \underset{1}{\Rightarrow} & \textbf{\textit{if}}\ C\ \underline{\textbf{\textit{if}}\ C\ S}\ \textbf{\textit{else}}\ S
\end{array}
$$

EL PROBLEMA DEL "ELSE DANZANTE"

Árboles correspondientes a la primera y a la segunda derivación.





Puesto que el lenguaje C asosia el "else" al "if más cercano", la derivación correcta es la primera.

EL PROBLEMA DEL "ELSE DANZANTE"

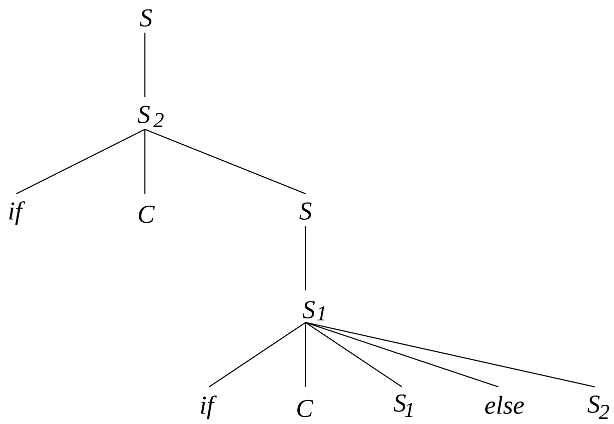
Afortunadamente, se puede reescribir la gramática para que tenga en cuenta este criterio semántico:

$$
\begin{array}{llcl}
 & & \ldots & \\
(1) & S & \longrightarrow & S_1 \\
(2) & S & \longrightarrow & S_2 \\
(3) & S_1 & \longrightarrow & \textbf{\textit{if}}\ C\ \ S_1\ \ \textbf{\textit{else}}\ S_1 \\
(4) & S_1 & \longrightarrow & I \\
(5) & S_2 & \longrightarrow & \textbf{\textit{if}}\ C\ \ S \\
(6) & S_2 & \longrightarrow & \textbf{\textit{if}}\ C\ \ S_1\ \ \textbf{\textit{else}}\ S_2 \\
 & & \ldots &
\end{array}
$$

donde S_1 genera la sentencia "if emparejada" mientras que S_2 genera la sentencia "if no emparejada".

La derivación de la cadena anterior es la siguiente:

$$
\begin{array}{lll}
S & \underset{2}{\Rightarrow} & S_2 \\
 & \underset{5}{\Rightarrow} & \textbf{\textit{if}}\ C\ \underline{S} \\
 & \underset{1}{\Rightarrow} & \textbf{\textit{if}}\ C\ \underline{S_1} \\
 & \underset{3}{\Rightarrow} & \textbf{\textit{if}}\ C\ \underline{\textbf{\textit{if}}\ C\ \ S_1\ \ \textbf{\textit{else}}\ S_2}
\end{array}
$$



Árbol sintáctico asociado a una derivación que asocia el "else al if más cercano".

AMBIGÜEDAD

Lenguajes intrínsecamente ambiguos

El siguiente lenguaje es un lenguaje de contexto libre intrínsecamente ambiguo porque todas las gramáticas de contexto libre que lo generan son ambiguas:

$$L = \{a^i b^i c^j | i, j \geq 1\} \cup \{a^i b^j c^j | i, j \geq 1\}$$

Este lenguaje es de contexto libre porque puede ser generado por una gramática de contexto libre como la siguiente:

$$
\begin{array}{rcl}
P & = & \{ \\
 & (1) & S \longrightarrow AC \\
 & (2) & S \longrightarrow BD \\
 & (3) & A \longrightarrow aAb \\
 & (4) & A \longrightarrow ab \\
 & (5) & C \longrightarrow cC \\
 & (6) & C \longrightarrow c \\
 & (7) & B \longrightarrow aB \\
 & (8) & B \longrightarrow a \\
 & (9) & D \longrightarrow bDc \\
 & (10) & D \longrightarrow bc \\
 & & \}
\end{array}
$$

AMBIGÜEDAD

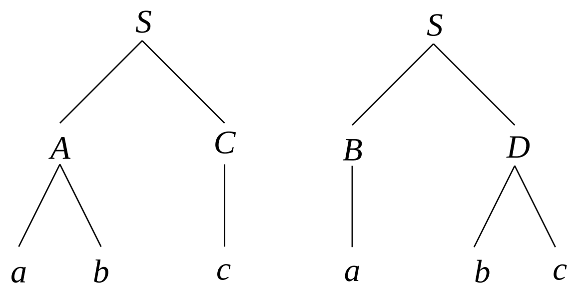
Esta gramática es ambigua porque puede derivar la cadena abc utilizando dos derivaciones por la izquierda diferentes:

1. Primera derivación:

$$\begin{aligned} S &\underset{1}{\Rightarrow} AC \\ &\underset{4}{\Rightarrow} abC \\ &\underset{6}{\Rightarrow} abc \end{aligned}$$

2. Segunda derivación:

$$\begin{aligned} S &\underset{2}{\Rightarrow} BD \\ &\underset{8}{\Rightarrow} aD \\ &\underset{10}{\Rightarrow} abc \end{aligned}$$



(a) (b)

Árboles de derivación diferentes de la cadena abc correspondientes a (a) la primera y a (b) la segunda derivación.

GRAMÁTICAS RECURSIVAS POR LA IZQUIERDA

Ejemplos de gramáticas recursivas por la izquierda:

- Parte izquierda de una sentencia de asignación múltiple en el lenguaje C: dado el siguiente conjunto de producciones

$$\begin{array}{rcl} P & = & \{ \\ & (1) & S \longrightarrow LE \\ & (2) & L \longrightarrow L \ \boldsymbol{identificador} \ = \\ & (3) & L \longrightarrow \boldsymbol{identificador} \ = \\ & (4) & E \longrightarrow E \ + \ T \\ & & \ldots \\ & & \} \end{array}$$

se puede generar la siguiente derivación recursiva

$$\begin{array}{rcl} S & \underset{1}{\Rightarrow} & LE \\ & \underset{2}{\Rightarrow} & L \ \boldsymbol{identificador} \ = \ E \\ & \underset{2}{\Rightarrow} & L \ \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ = \ E \\ & \underset{3}{\Rightarrow} & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ = \ E \end{array}$$

- Lista de parámetros de un procedimiento o función:

$$\begin{array}{rcl} P & = & \{ \\ & & S \longrightarrow \boldsymbol{identificador} \ (L) \\ & & L \longrightarrow L, \boldsymbol{identificador} \\ & & L \longrightarrow \boldsymbol{identificador} \\ & & \ldots \\ & & \} \end{array}$$

- Componente de un *array* de varias dimensiones:

$$\begin{array}{rcl} P & = & \{ \\ & & S \longrightarrow \boldsymbol{identificador} \ D \\ & & D \longrightarrow D[\boldsymbol{número}] \\ & & D \longrightarrow [\boldsymbol{número}] \\ & & \ldots \\ & & \} \end{array}$$

GRAMÁTICAS RECURSIVAS POR LA IZQUIERDA

Eliminación de la recursividad inmediata por la izquierda

- Entrada: $G = (V_N, V_T, P, S)$ gramática de contexto libre con reglas recursivas por la izquierda.
- Salida: $G' = (V'_N, V_T, P', S)$ gramática sin reglas de producción recursivas por la izquierda.

inicio
    $P' \leftarrow \emptyset$
    para cada $A \in V_N$ ***hacer***
        si A no tiene producciones recursivas
            entonces se añaden a P' las producciones de A
        si no
            si $A \longrightarrow A\alpha_1|A\alpha_2|\cdots|A\alpha_p|\beta_1|\beta_2|\cdots|\beta_q \in P$
                donde $\alpha_i \neq \epsilon \; \forall i \in \{1, 2, \ldots, p\}$
                y β_j no empieza por $A \; \forall j \in \{1, 2, \ldots, q\}$
                entonces
                    se añaden a P' las producciones
                    $A \longrightarrow \beta_j|\beta_j A' \quad \forall i \in \{1, 2, \ldots, q\}$
                    $A' \longrightarrow \alpha_i|\alpha_i A' \quad \forall j \in \{1, 2, \ldots, p\}$
                    donde A' es un nuevo símbolo no terminal
            fin si
        fin si
    fin para
fin

GRAMÁTICAS RECURSIVAS POR LA IZQUIERDA

Considérese el conjunto de producciones de la siguiente gramática:

$$
\begin{aligned}
P \;=\; \{ & \\
& S \longrightarrow \textbf{\textit{identificador}} \;=\; E \\
& E \longrightarrow E \;+\; T \mid T \;*\; F \mid (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \\
& T \longrightarrow T \;*\; F \mid (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \\
& F \longrightarrow (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \\
& \}
\end{aligned}
$$

Paso 1: La producción de S no es recursiva por la izquierda y, por tanto, se añade a P'.

Paso 2: El símbolo E tiene una regla recursiva por la izquierda y cuatro más que no lo son. Las nuevas reglas que se añaden a P' son:

$$
\begin{aligned}
E \;&\longrightarrow\; T \;*\; F \mid \;(\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \mid \\
&\;\; T \;*\; F\; E' \mid \;(\; E \;)\; E' \mid \textbf{\textit{identificador}}\; E' \mid \textbf{\textit{número}}\; E' \\
E' \;&\longrightarrow\; +\; T \mid \;+\; T\; E'
\end{aligned}
$$

Paso 3: El símbolo T tiene una regla recursiva por la izquierda y tres más que no lo son. Las nuevas reglas que se añaden a P' son:

$$
\begin{aligned}
T \;&\longrightarrow\; (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \mid \\
&\;\; (\; E \;)\; T' \mid \textbf{\textit{identificador}}\; T' \mid \textbf{\textit{número}}\; T' \\
T' \;&\longrightarrow\; *\; F \mid \;*\; F\; T'
\end{aligned}
$$

Paso 4: El símbolo F no posee reglas recursivass. Por tanto, todas ellas se añaden a P'

El conjunto de producciones resultante es:

$$
\begin{aligned}
P' \;=\; \{ & \\
& S \longrightarrow \textbf{\textit{identificador}} \;=\; E \\
& E \longrightarrow T \;*\; F \mid \;(\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \mid \\
& \; T \;*\; F\; E' \mid \;(\; E \;)\; E' \mid \textbf{\textit{identificador}}\; E' \mid \textbf{\textit{número}}\; E' \\
& E' \longrightarrow +\; T \mid \;+\; T\; E' \\
& T \longrightarrow (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \mid \\
& \; (\; E \;)\; T' \mid \textbf{\textit{identificador}}\; T' \mid \textbf{\textit{número}}\; T' \\
& T' \longrightarrow *\; F \mid \;*\; F\; T' \\
& F \longrightarrow (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \\
& \}
\end{aligned}
$$

GRAMÁTICAS RECURSIVAS POR LA IZQUIERDA

Para comprobar que las dos gramáticas son equivalentes, considérense las siguientes derivaciones. La primera derivación se ha generado utilizando la gramática original con producciones recursivas por la izquierda:

$$\begin{array}{lll}
S & \Rightarrow & \boldsymbol{identificador} \ = \ E \\
 & \Rightarrow & \boldsymbol{identificador} \ = \ E \ + \ T \\
 & \Rightarrow & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ T \\
 & \Rightarrow & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ \boldsymbol{identificador}
\end{array}$$

La segunda derivación genera la misma cadena que la anterior pero utiliza las producciones de la nuevas gramática sin recursividad por la izquierda:

$$\begin{array}{lll}
S & \Rightarrow & \boldsymbol{identificador} \ = \ E \\
 & \Rightarrow & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ E' \\
 & \Rightarrow & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ T \\
 & \Rightarrow & \boldsymbol{identificador} \ = \ \boldsymbol{identificador} \ + \ \boldsymbol{identificador}
\end{array}$$

GRAMÁTICAS RECURSIVAS POR LA IZQUIERDA

Eliminación de la recursividad general por la izquierda

- Entrada: $G = (V_N, V_T, P, S)$ gramática de contexto libre propia, es decir, sin ciclos, sin producciones ϵ ni símbolos inútiles.
- Salida: $G' = (V'_N, V_T, P', S)$ gramática sin recursividad por la izquierda.

inicio
  $P' \leftarrow \emptyset$
  Ordénense los símbolos no terminales de la gramática: $\{A_1, A_2, \ldots, A_n\}$
  para i ***de*** 1 ***a*** n ***hacer***
    para j ***de*** 1 ***a*** $i-1$ ***hacer***
      si $A_i \longrightarrow A_j \ \gamma \ \in P$
        entonces
        Añadir a P' las producciones
          $A_i \longrightarrow \delta_1 \ \gamma \ | \ \cdots \ | \ \delta_k \ \gamma$
        donde
          $A_j \longrightarrow \delta_1 \ | \ \cdots \ | \ \delta_k$
        son las producciones actuales de A_j
      fin si
    fin para
    Eliminar la recursividad inmediata por la izquierda
    de las producciones de A_i.
  fin para
fin

GRAMÁTICAS RECURSIVAS POR LA IZQUIERDA

Eliminación de la recursividad general por la izquierda

Considérese la siguiente gramática recursiva por la izquierda:

$$
\begin{array}{rcl}
P & = & \{ \\
 & (1) & S \longrightarrow A\ B \\
 & (2) & S \longrightarrow c \\
 & (3) & A \longrightarrow B\ b \\
 & (4) & A \longrightarrow S\ d \\
 & (5) & A \longrightarrow a \\
 & (6) & B \longrightarrow S\ b \\
 & (7) & B \longrightarrow A\ a \\
 & & \}
\end{array}
$$

Ordenamiento de los símbolos no terminales: $\{S, A, B\}$

Paso exterior 1: Producciones de S

Paso interior 1: S no tiene ninguna producción que comience por un símbolo con un número de orden inferior al suyo.

Eliminación de la recusividad inmediata: S no tiene recursividad inmediata por la izquierda.

Paso exterior 2: Producciones de A

Paso interior 1: Sustitución de las producciones de A que comienzan por S: la regla número (4) $A \longrightarrow S\ d$ se sustituye por las reglas

$$A \longrightarrow A\ B\ d \mid c\ d$$

quedando las reglas de A de la siguiente forma:

$$A \longrightarrow A\ B\ d \mid B\ b \mid c\ d \mid a$$

Eliminación de la recusividad inmediata: se sustituyen las producciones de A por las siguientes producciones:

$$
\begin{array}{rcl}
A & \longrightarrow & B\ b \mid c\ d \mid a \mid \\
 & & B\ b\ A' \mid c\ d\ A' \mid a\ A' \\
A' & \longrightarrow & B\ d \mid B\ d\ A'
\end{array}
$$

Paso exterior 3: Producciones de B

Paso interior 1: Sustitución de las producciones de B que comienzan por S: la regla número (6) $B \longrightarrow S\ b$ se sustituye por las reglas

$$B \longrightarrow A\ B\ b \mid c\ b$$

quedando las reglas de B de la siguiente forma:

$$B \longrightarrow A\ B\ b \mid c\ b \mid A\ a$$

Paso interior 2: Sustitución de las producciones de B que comienzan por A: las regla $B \longrightarrow A\ B\ b$ se sustituye por las reglas

$$\begin{array}{lcl} B & \longrightarrow & B\ b\ B\ b \mid c\ d\ B\ b \mid a\ B\ b \mid \\ & & B\ b\ A'\ B\ b \mid c\ d\ A'\ B\ b \mid a\ A'\ B\ b \end{array}$$

y la regla $B \longrightarrow A\ a$ se sustituye por las reglas

$$\begin{array}{lcl} B & \longrightarrow & B\ b\ a \mid c\ d\ a \mid a\ a \mid \\ & & B\ b\ A'\ a \mid c\ d\ A'\ a \mid a\ A'\ a \end{array}$$

quedando las reglas de B de la siguiente forma:

$$\begin{array}{lcl} B & \longrightarrow & B\ b\ B\ b \mid B\ b\ A'\ B\ b \mid B\ b\ a \mid B\ b\ A'\ a \mid \\ & & c\ d\ B\ b \mid a\ B\ b \mid c\ d\ A'\ B\ b \mid a\ A'\ B\ b \mid \\ & & b\ a \mid c\ d\ a \mid a\ a \mid c\ d\ A'\ a \mid a\ A'\ a \mid \\ & & c\ b \end{array}$$

Eliminación de la recusividad inmediata: se sustituyen las producciones de B por las siguientes producciones:

$$\begin{array}{lcl} B & \longrightarrow & c\ d\ B\ b \mid a\ B\ b \mid c\ d\ A'\ B\ b \mid a\ A'\ B\ b \mid \\ & & b\ a \mid c\ d\ a \mid a\ a \mid c\ d\ A'\ a \mid a\ A'\ a \mid \\ & & c\ b \mid \\ & & c\ d\ B\ b\ B' \mid a\ B\ b\ B' \mid c\ d\ A'\ B\ b\ B' \mid a\ A'\ B\ b\ B' \mid \\ & & b\ a\ B' \mid c\ d\ a\ B' \mid a\ a\ B' \mid c\ d\ A'\ a\ B' \mid a\ A'\ a\ B' \mid \\ & & c\ b\ B' \\ B' & \longrightarrow & b\ B\ b \mid b\ A'\ B\ b \mid b\ a \mid b\ A'\ a \mid \\ & & b\ B\ b\ B' \mid b\ A'\ B\ b\ B' \mid b\ a\ B' \mid b\ A'\ a\ B' \end{array}$$

NECESIDAD DE LA FACTORIZACIÓN POR LA IZQUIERDA

Considérense las siguientes producciones:

$$\begin{array}{lcl} S & \longrightarrow & \boldsymbol{si}\ E\ \boldsymbol{entonces}\ S\ \boldsymbol{si\ no}\ S\ \boldsymbol{fin\ si}\ | \\ & & \boldsymbol{si}\ E\ \boldsymbol{entonces}\ S\ \boldsymbol{fin\ si} \end{array}$$

Al leer el componente léxico ***si*** no se sabe aún qué producción elegir para expandir la sentencia S. Sin embargo, se puede posponer esta decición si se utilizan las siguientes reglas de producción.

$$\begin{array}{lcl} S & \longrightarrow & \boldsymbol{si}\ E\ \boldsymbol{entonces}\ S\ S' \\ S' & \longrightarrow & \boldsymbol{si\ no}\ S\ \boldsymbol{fin\ si}\ |\ \boldsymbol{fin\ si} \end{array}$$

En general, si $A \longrightarrow \alpha\ \beta_1 \mid \alpha\ \beta_2$ son dos producciones de A y la entrada a analizar comienza por una cadena no vacía derivada a partir de α, no se sabe si expandir A a $\alpha\ \beta_1$ o a $\alpha\ \beta_2$. Para ello, las producciones se transforman en las siguientes producciones:

$$\begin{array}{lcl} A & \longrightarrow & \alpha\ A' \\ A' & \longrightarrow & \beta_1 \mid \beta_2 \end{array}$$

FACTORIZACIÓN POR LA IZQUIERDA

Algoritmo de Factorización por la izquierda

- Entrada: $G = (V_N, V_T, P, S)$ gramática de contexto libre propia.
- Salida: $G' = (V'_N, V_T, P', S)$ gramática factorizada por la izquierda.

inicio
    para cada símbolo no terminal A ***hacer***
        mientras A tenga dos producciones <u>actuales</u>
                        con el mismo prefijo
        hacer
            si α es el prefijo más largo de dos o más
                alternativas de A y $\alpha \neq \epsilon$
                entonces
                    Sustituir todas las producciones
                        $A \longrightarrow \alpha\ \beta_1 \mid \cdots \mid \alpha\ \beta_p \mid \gamma_1 \mid \cdots \mid \gamma_q$
                    donde γ_i no empieza por α $\forall i \in \{1, 2, \ldots, q\}$
                    por las producciones
                        $A \longrightarrow \alpha\ A' \mid \gamma_1 \mid \cdots \mid \gamma_q$
                        $A' \longrightarrow \beta_1 \mid \cdots \mid \beta_p$
            fin si
        fin mientras
    fin para
fin

FACTORIZACIÓN POR LA IZQUIERDA

Considérese el siguiente conjunto de producciones de una gramática de contexto libre:

$$
\begin{aligned}
P \;=\; & \{ \\
& S \longrightarrow A\ B\ c \mid A\ B\ d\ e \mid A\ B\ d\ f \mid A\ B\ S \\
& A \longrightarrow a \\
& B \longrightarrow b \\
& \}
\end{aligned}
$$

Paso 1: $\alpha_1 = ABd$ es el prefijo más largo de dos producciones de S. Por tanto, las reglas de S se sustituyen por las siguientes:

$$
\begin{array}{rcl}
S & \longrightarrow & A\ B\ d\ S' \mid A\ B\ c \mid A\ B\ S \\
S' & \longrightarrow & e \mid f
\end{array}
$$

La cadena $\alpha_2 = AB$ es ahora el prefijo más largo de tres producciones actuales de S. Por tanto, las reglas de S se sustituyen por las siguientes:

$$
\begin{array}{rcl}
S & \longrightarrow & A\ B\ S'' \\
S'' & \longrightarrow & d\ S' \mid c \mid S \\
S' & \longrightarrow & e \mid f
\end{array}
$$

Pasos 2 y 3: Las producciones de A y B no requieren factorización.

RECURSIVIDAD INMEDIATA Y FACTORIZACIÓN POR LA IZQUIERDA

Eliminación de la recursividad inmediata por la izquierda y factorización por la izquierda

- Entrada: $G = (V_N, V_T, P, S)$ gramática de contexto libre con reglas recursivas por la izquierda.
- Salida: $G' = (V_N', V_T, P', S)$ gramática sin reglas de producción recursivas por la izquierda y factorizada por la izquierda.

inicio
  $P' \leftarrow \emptyset$
  para cada $A \in V_N$ ***hacer***
    si A no tiene producciones recursivas
      entonces se añaden a P'
          las producciones de A "factorizadas"
    si no
      si $A \longrightarrow A\alpha_1 | A\alpha_2 | \cdots | A\alpha_p | \beta_1 | \beta_2 | \cdots | \beta_q \in P$
        donde $\alpha_i \neq \epsilon \ \forall i \in \{1, 2, \ldots, p\}$
        y β_j no empieza por $A \ \forall j \in \{1, 2, \ldots, q\}$
        entonces
          se añaden a P' las producciones
          $A \longrightarrow \beta_j A' \quad \forall j \in \{1, 2, \ldots, q\}$
          $A' \longrightarrow \alpha_i A' | \epsilon \quad \forall i \in \{1, 2, \ldots, p\}$
          donde A' es un nuevo símbolo no terminal
      fin si
    fin si
  fin para
fin

RECURSIVIDAD INMEDIATA Y FACTORIZACIÓN POR LA IZQUIERDA

Sea la gramática sin reglas unitarias que genera las expresiones aritméticas

$$
\begin{aligned}
P \;=\; \{ & \\
& S \longrightarrow \textbf{\textit{identificador}} \;=\; E \\
& E \longrightarrow E \;+\; T \mid T \;*\; F \mid (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \\
& T \longrightarrow T \;*\; F \mid (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \\
& F \longrightarrow (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \\
& \}
\end{aligned}
$$

La aplicación del algoritmo que factoriza y elimina la recurisividad inmediata por la izquierda genera la la siguiente gramática:

$$
\begin{aligned}
P' \;=\; \{ & \\
& S \longrightarrow \textbf{\textit{identificador}} \;=\; E \\
& E \longrightarrow T \;*\; F \; E' \mid \;(\; E \;)\; E' \mid \textbf{\textit{identificador}}\; E' \mid \textbf{\textit{número}}\; E' \\
& E' \longrightarrow \;+\; T \; E' \mid \epsilon \\
& T \longrightarrow \;(\; E \;)\; T' \mid \textbf{\textit{identificador}}\; T' \mid \textbf{\textit{número}}\; T' \\
& T' \longrightarrow \;*\; F \; T' \mid \epsilon \\
& F \longrightarrow (\; E \;) \mid \textbf{\textit{identificador}} \mid \textbf{\textit{número}} \\
& \}
\end{aligned}
$$

FORMA NORMAL DE CHOMSKY

Se dice que una gramática de contexto libre está en la ***forma normal de Chomsky (F.N.C.)*** si sus reglas son de una de estas dos formas:

$$\begin{array}{lcl} A & \longrightarrow & B\ C \\ A & \longrightarrow & a \end{array}$$

donde $A, B, C \in V_N$ y $a \in V_T$

Sea una gramática en la forma normal de Chomsky con el siguiente conjunto de producciones:

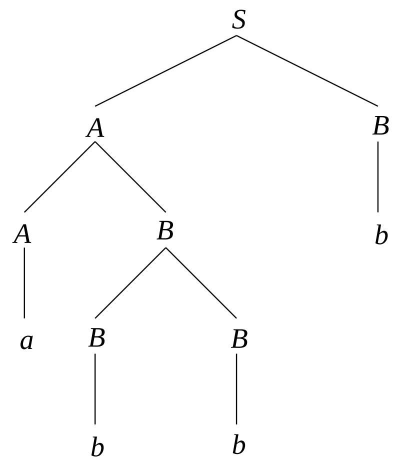
$$\begin{array}{lcl} P & = & \{ \\ & (1) & S \longrightarrow A\ B \\ & (2) & A \longrightarrow A\ B \\ & (3) & A \longrightarrow a \\ & (4) & B \longrightarrow B\ B \\ & (5) & B \longrightarrow b \\ & & \} \end{array}$$

Derivación generada por una gramática en la forma normal de Chomsky:

$$\begin{array}{lcl} S & \underset{1}{\Rightarrow} & A\ B \\ & \underset{2}{\Rightarrow} & A\ B\ B \\ & \underset{3}{\Rightarrow} & a\ B\ B \\ & \underset{4}{\Rightarrow} & a\ B\ B\ B \\ & \underset{5}{\Rightarrow} & a\ b\ B\ B \\ & \underset{5}{\Rightarrow} & a\ b\ b\ B \\ & \underset{6}{\Rightarrow} & a\ b\ b\ b \end{array}$$

FORMA NORMAL DE CHOMSKY

Árbol binario correspondiente a la derivación anterior:



FORMA NORMAL DE CHOMSKY

Algoritmo para obtener la forma normal de Chomsky:

1. Generación de una gramática $G_1 = (V_{N_1}, V_T, P_1, S)$ donde las reglas de P_1 son de la forma
$$\begin{array}{lcll} A & \longrightarrow & B_1\ B_2\ \cdots\ B_k & \text{donde } k \geq 2 \\ A & \longrightarrow & a & \end{array}$$
verificándose que $L(G) = L(G_1)$.

2. A partir de la gramática G_1 obtenida en el paso anterior, se genera otra gramática G_2 que estará en la forma normal de Chomsky y que será equivalente a G, es decir, $L(G) = L(G_2)$.

Paso 1: Sea $A \longrightarrow X_1\ X_2\ \cdots\ X_k \in P$:

1. Si $k = 1$ entonces la regla es simplemente $A \longrightarrow X_1$, donde $X_1 \in V_T$, porque la gramática no tiene reglas unitarias al ser una gramática propia. En este caso, la regla $A \longrightarrow X_1$ se añade a P_1.
2. Si $k \geq 2$ entonces se añade a P_1 la regla $A \longrightarrow B_1\ B_2\ \cdots\ B_k$ donde
   - $B_i = X_i$ si $X_i \in V_N$
   - o B_i es un nuevo símbolo no terminal si $X_i = a_i \in V_T$, en cuyo caso, también se añade a P_1 la regla $B_i \longrightarrow X_i$.

Se verifica que $L(G_1) = L(G) - \{\epsilon\}$.

Paso 2: Para generar las reglas de P_2 se han analizar las reglas de P_1:

1. Si la regla de P_1 es de la forma $A \longrightarrow a$ entonces se añade a P_2.
2. Si $A \longrightarrow B_1\ B_2\ \cdots\ B_k \in P_1$ entonces se pueden presentar dos casos:
   a) Si $k = 2$ entonces la regla está en la forma normal de Chomsky y se añade a P_2.
   b) Si $k \geq 3$ entonces se añaden a P_2 el siguiente conjunto de reglas:
$$\begin{array}{rcl} A & \longrightarrow & B_1\ C_1 \\ C_1 & \longrightarrow & B_2\ C_2 \\ & \cdots & \\ C_{k-1} & \longrightarrow & B_{k-2}\ C_{k-2} \\ C_{k-2} & \longrightarrow & B_{k-1}\ B_k \end{array}$$

G_2 está en la forma normal de Chomsky y $L(G_2) = L(G_1) = L(G) - \{\epsilon\}$.

FORMA NORMAL DE CHOMSKY

Sea el siguiente conjunto de producciones de una gramática de contexto libre propia

$$
\begin{aligned}
P \quad = \quad & \{ \\
& S \longrightarrow a\ \ A\ \ B \\
& A \longrightarrow a\ \ B\ \ b \\
& A \longrightarrow a \\
& B \longrightarrow b\ \ b \\
& \}
\end{aligned}
$$

Paso 1: Se genera el siguiente conjunto de producciones:

$$
\begin{aligned}
P_1 \quad = \quad & \{ \\
& S \longrightarrow B_1\ \ A\ \ B \\
& A \longrightarrow B_1\ \ B\ \ B_2 \\
& A \longrightarrow a \\
& B \longrightarrow B_2\ \ B_2 \\
& B_1 \longrightarrow a \\
& B_2 \longrightarrow b \\
& \}
\end{aligned}
$$

Paso 2:

$$
\begin{aligned}
P_2 \quad = \quad & \{ \\
& S \longrightarrow B_1\ \ C_1 \\
& C_1 \longrightarrow A\ \ B \\
& A \longrightarrow B_1\ \ C_2 \\
& C_2 \longrightarrow B\ \ B_2 \\
& A \longrightarrow a \\
& B \longrightarrow B_2\ \ B_2 \\
& B_1 \longrightarrow a \\
& B_2 \longrightarrow b \\
& \}
\end{aligned}
$$

FORMA NORMAL DE GREIBACH

Se dice que una gramática de contexto libre está en la ***forma normal de Greibach (F.N.G.)*** si sus reglas son de la forma:

$$A \longrightarrow a\ \alpha$$

donde $A \in V_N$, $a \in V_T$ y $\alpha \in V_N^*$

Si $G = (V_N, V_T, P, S)$ es una gramática de contexto libre que está en la forma normal de Chomsky, se va a construir otra gramática que estará en la forma normal de Greibach mediante la aplicación de los siguientes pasos:

1. Aplicación del algoritmo que elimina la recursividad general por la izquierda.
2. Transformación de las reglas de los símbolos no terminales de la gramática original.
3. Transformación de las reglas de los símbolos no terminales obtenidos al eliminar la recursividad inmediata.

FORMA NORMAL DE GREIBACH

Paso 1: Aplicación del algoritmo que elimina la recursividad general por la izquierda.

Las reglas de producción resultantes serán de la forma:

$$\begin{array}{lll} A_i & \longrightarrow & A_j\gamma \quad \forall j > i \wedge i, j \in \{i, 2, \ldots, n\} \\ A_i & \longrightarrow & a\gamma \\ A'_i & \longrightarrow & \gamma \end{array}$$

donde $A_i, A_j \in V_N$, $a \in V_T$, los símbolos A'_i ($i \in \{1, 2, \ldots, m\}$) han sido generados al eliminar la recursividad inmediata por la izquierda y $\gamma \in (V_N \cup \{A'_1, A'_2, \ldots, A'_n\})*$. En particular, la reglas del símbolo A_n serán de la forma $A_n \longrightarrow a\ \gamma$ y ya estarán en la forma normal de Greibach.

Paso 2: Transformación de las reglas de los símbolos no terminales de la gramática original mediante la aplicación del siguiente algoritmo:

Transformación de las reglas de los símbolos no terminales de la gramática original

inicio
    para i ***de*** $n-1$ ***a*** 1 ***hacer***
        para j ***de*** $i+1$ ***a*** n ***hacer***
            para cada producción actual de A_i
                    de la forma $A_i \longrightarrow A_j\ \gamma$
            hacer
                si $A_j \longrightarrow a_1\ \alpha_1 \mid a_2\ \alpha_2 \mid \cdots \mid a_p\ \alpha_p$
                    son las producciones actuales de A_j
                    entonces $A_i \longrightarrow a_1\ \alpha_1\ \gamma \mid a_2\ \alpha_2\ \gamma \mid \cdots \mid a_p\ \alpha_p\ \gamma$
                        pasan a ser producciones actuales de A_i
                fin si
            fin para
        fin para
    fin para
fin

Las reglas de producción resultantes serán de la forma:

$$
\begin{array}{lcl}
A_i & \longrightarrow & a\ \gamma \\
A'_i & \longrightarrow & \gamma
\end{array}
$$

En particular, todas la reglas de los símbolos no terminales originales estarán en la forma normal de Greibach.

Paso 3: Aplicación del siguiente algoritmo para transformar las reglas de los símbolos no terminales obtenidos al eliminar la recursividad inmediata:

Transformación de las reglas de los símbolos obtenidos al eliminar la recursividad inmediata

inicio
    para i ***de*** 1 ***a*** m ***hacer***
        para j ***de*** 1 ***a*** n ***hacer***
            para cada producción <u>actual</u> de A'_i
                de la forma $A'_i \longrightarrow A_j \ \gamma$
            hacer
                si $A_j \longrightarrow a_1 \ \alpha_1 \mid a_2 \ \alpha_2 \mid \cdots \mid a_p \ \alpha_p$
                    son las producciones <u>actuales</u> de A_j
                    entonces $A'_i \longrightarrow a_1 \ \alpha_1 \ \gamma \mid a_2 \ \alpha_2 \ \gamma \mid \cdots \mid a_p \ \alpha_p \ \gamma$
                        pasan a ser producciones <u>actuales</u> de A'_i
                fin si
            fin para
        fin para
    fin para
fin

Las reglas de producción resultantes serán de la forma:

$$\begin{array}{lcl} A_i & \longrightarrow & a \ \gamma \\ A'_i & \longrightarrow & a \ \gamma \end{array}$$

Por tanto, todas la reglas estarán en la forma normal de Greibach.

FORMA NORMAL DE GREIBACH

Se va a obtener la gramática en la forma normal de Greibach equivalente a la siguiente gramática, que ya está en la forma normal de Chomsky:

$$
\begin{array}{rcl}
P & = & \{ \\
 & (1) & S \longrightarrow A\ B \\
 & (2) & A \longrightarrow S\ B \\
 & (3) & A \longrightarrow a \\
 & (4) & B \longrightarrow B\ A \\
 & (5) & B \longrightarrow d \\
 & & \}
\end{array}
$$

Paso 1: Aplicación del algoritmo de la recursividad general por la izquierda:

1. La regla de S no necesita transformarse.
2. Se sustituye S por su alternativa en la regla $A \longrightarrow S\ B$ generando la regla:
$$A \longrightarrow A\ B\ B$$
3. Eliminación de la recursividad por la izquierda de las reglas de A. Las reglas actuales de A son
$$A \quad \longrightarrow \quad A\ B\ B \mid a$$
y las reglas que se generan al eliminar la recursividad inmendiata son:
$$
\begin{array}{rcl}
A & \longrightarrow & a \mid a\ A' \\
A' & \longrightarrow & B\ B \mid B\ B\ A'
\end{array}
$$
4. No se tiene que sustituir ningún símbolo en las reglas de B.
5. Eliminación de la recursividad por la izquierda de las reglas de B. Las reglas actuales de B son
$$B \quad \longrightarrow \quad B\ A \mid d$$
y las reglas que se generan al eliminar la recursividad inmendiata son:
$$
\begin{array}{rcl}
B & \longrightarrow & d \mid d\ B' \\
B' & \longrightarrow & A \mid A\ B'
\end{array}
$$

$$
\begin{aligned}
P_1 \;=\; \{ & \\
& S \longrightarrow A\ B \\
& A \longrightarrow a \mid a\ A' \\
& B \longrightarrow d \mid d\ B' \\
& A' \longrightarrow B\ B\ |B\ B\ B' \\
& B' \longrightarrow A \mid A\ B' \\
& \}
\end{aligned}
$$

Paso 2: Transformación de las reglas de los símbolos no terminales originales.

1. Las reglas de B ya están en la forma normal de Greibach.
2. Las reglas de A ya están en la forma normal de Greibach.
3. Se sustituye A por sus alternativas en la regla $S \longrightarrow A\ B$ generando las siguientes reglas de S:

$$S \quad \longrightarrow \quad a\ B \mid a\ A'\ B$$

Paso 3: Transformación de las reglas de los símbolos no terminales generados al eliminar la recursividad inmediata por la izquierda.

1. Trasformación de las reglas de A': $A' \longrightarrow B\ B \mid B\ B\ A'$. Se sustituye B por sus alternativas, generándose las siguientes reglas de A':

$$A' \quad \longrightarrow \quad d\ B \mid d\ B\ A' \mid d\ B'\ B \mid d\ B'\ B\ A'$$

2. Transformación de las reglas de B': $B' \longrightarrow A \mid A\ B'$. Se sustituye A por sus alternativas, generándose las siguientes reglas de B':

$$B' \quad \longrightarrow \quad a \mid a\ A' \mid a\ B' \mid a\ A'\ B'$$

3. Las reglas de B' no requieren ninguna sustitución.

El conjunto de producciones de la gramática en la forma normal de Greibach es:

$$
\begin{aligned}
P_2 \;=\; \{ & \\
& S \longrightarrow a\ B \mid a\ A'\ B \\
& A \longrightarrow a \mid a\ A' \\
& B \longrightarrow d \mid d\ B' \\
& A' \longrightarrow d\ B \mid d\ B\ A' \mid d\ B'\ B \mid d\ B'\ B\ A' \\
& B' \longrightarrow a \mid a\ A' \mid a\ B' \mid a\ A'\ B' \\
& \}
\end{aligned}
$$