

# Autómatas y Lenguajes Formales 2016-1

Maestría en Ciencia e Ingeniería de la Computación UNAM

Tema 9: Propiedades de cerradura y normalización de gramáticas  
libres de contexto

Dr. Favio Ezequiel Miranda Perea

[favio@ciencias.unam.mx](mailto:favio@ciencias.unam.mx)

Facultad de Ciencias UNAM

3 de febrero de 2020



# Propiedades de cerradura

## Lenguaje Libres del Contexto

La clase de los lenguajes libres del contexto es cerrada bajo las siguientes operaciones:

- Unión: si  $L_1, L_2$  son lenguajes libres del contexto entonces  $L_1 \cup L_2$  es un lenguaje libre del contexto.



# Propiedades de cerradura

## Lenguaje Libres del Contexto

La clase de los lenguajes libres del contexto es cerrada bajo las siguientes operaciones:

- Unión: si  $L_1, L_2$  son lenguajes libres del contexto entonces  $L_1 \cup L_2$  es un lenguaje libre del contexto.
- Concatenación: si  $L_1, L_2$  son lenguajes libres del contexto entonces  $L_1 L_2$  es un lenguaje libre del contexto.



# Propiedades de cerradura

## Lenguaje Libres del Contexto

La clase de los lenguajes libres del contexto es cerrada bajo las siguientes operaciones:

- Unión: si  $L_1, L_2$  son lenguajes libres del contexto entonces  $L_1 \cup L_2$  es un lenguaje libre del contexto.
- Concatenación: si  $L_1, L_2$  son lenguajes libres del contexto entonces  $L_1 L_2$  es un lenguaje libre del contexto.
- Estrella de Kleene: si  $L_1$  es un lenguaje libre del contexto entonces  $L_1^*$  es un lenguaje libre del contexto.



# Cerradura bajo la unión

## Lenguajes Libres del Contexto

Si  $G_1 = \langle V_1, T, S_1, P_1 \rangle$ ,  $G_2 = \langle V_2, T, S_2, P_2 \rangle$  son GLC con  $L_1 = L(G_1)$ ,  $L_2 = L(G_2)$  entonces  $L_1 \cup L_2 = L(G)$  donde  $G$  es la gramática

$$G = \langle V_1 \cup V_2 \cup \{S\}, T, S, P \rangle$$

y  $P$  está dado por  $P_1 \cup P_2$  mas las producciones:

$$S \rightarrow S_1, \quad S \rightarrow S_2$$



# Cerradura bajo la concatenación

## Lenguajes Libres del Contexto

Si  $G_1 = \langle V_1, T, S_1, P_1 \rangle$ ,  $G_2 = \langle V_2, T, S_2, P_2 \rangle$  son GLC con  $L_1 = L(G_1)$ ,  $L_2 = L(G_2)$  entonces  $L_1 L_2 = L(G)$  donde  $G$  es la gramática

$$G = \langle V_1 \cup V_2 \cup \{S\}, T, S, P \rangle$$

y  $P$  está dado por  $P_1 \cup P_2$  mas la producción:

$$S \rightarrow S_1 S_2$$



# Cerradura bajo la estrella de Kleene

## Lenguajes Libres del Contexto

Si  $G_1 = \langle V_1, T, S_1, P_1 \rangle$  es una GLC con  $L_1 = L(G_1)$  entonces  $L_1^* = L(G)$  donde  $G$  es la gramática

$$G = \langle V_1 \cup \{S\}, T, S, P \rangle$$

y  $P$  está dado por  $P_1$  mas las producciones:

$$S \rightarrow S_1 S_1 \quad S \rightarrow \varepsilon$$



# Propiedades de cerradura inválidas

## Lenguajes libres del contexto

En general las siguientes propiedades no son válidas para lenguajes libres del contexto.

- Cerradura bajo la intersección.



# Propiedades de cerradura inválidas

## Lenguajes libres del contexto

En general las siguientes propiedades no son válidas para lenguajes libres del contexto.

- Cerradura bajo la intersección.
- Cerradura bajo el complemento.



# Propiedades de cerradura inválidas

## Lenguajes libres del contexto

En general las siguientes propiedades no son válidas para lenguajes libres del contexto.

- Cerradura bajo la intersección.
- Cerradura bajo el complemento.
- Cerradura bajo la diferencia.



# Propiedades de cerradura inválidas

## Intersección

- La intersección de dos LIC puede ser un lenguaje que no es libre del contexto.



# Propiedades de cerradura inválidas

## Intersección

- La intersección de dos LIC puede ser un lenguaje que no es libre del contexto.
- $L_1 = \{a^i b^j c^j \mid i, j \geq 1\}$  es libre del contexto:

$$S \rightarrow AB, A \rightarrow aAb \mid ab, B \rightarrow cC \mid c$$



# Propiedades de cerradura inválidas

## Intersección

- La intersección de dos LIC puede ser un lenguaje que no es libre del contexto.
- $L_1 = \{a^i b^j c^j \mid i, j \geq 1\}$  es libre del contexto:

$$S \rightarrow AB, A \rightarrow aAb \mid ab, B \rightarrow cC \mid c$$

- $L_2 = \{a^i b^j c^j \mid i, j \geq 1\}$  es libre del contexto:

$$S \rightarrow AB, A \rightarrow aA \mid a, B \rightarrow bBc \mid bc$$



# Propiedades de cerradura inválidas

## Intersección

- La intersección de dos LIC puede ser un lenguaje que no es libre del contexto.
- $L_1 = \{a^i b^j c^j \mid i, j \geq 1\}$  es libre del contexto:

$$S \rightarrow AB, A \rightarrow aAb \mid ab, B \rightarrow cC \mid c$$

- $L_2 = \{a^i b^j c^j \mid i, j \geq 1\}$  es libre del contexto:

$$S \rightarrow AB, A \rightarrow aA \mid a, B \rightarrow bBc \mid bc$$

- $L_1 \cap L_2 = \{a^i b^i c^i \mid i \geq 1\}$  no es independiente del contexto.



# Propiedades de cerradura inválidas

## Complemento y diferencia

- Si el complemento de un LLC  $L$ ,  $\bar{L}$  fuera también libre del contexto entonces la intersección también lo sería pues

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



# Propiedades de cerradura inválidas

## Complemento y diferencia

- Si el complemento de un LLC  $L$ ,  $\bar{L}$  fuera también libre del contexto entonces la intersección también lo sería pues

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

- Si la diferencia fuera un LLC, entonces también lo sería el complemento pues

$$\bar{L} = \Sigma^* - L$$



# Introducción

## Normalización de gramáticas

- La normalización consiste en transformar todas las producciones de una gramática de manera que tengan cierta forma sintáctica en particular.



# Introducción

## Normalización de gramáticas

- La normalización consiste en transformar todas las producciones de una gramática de manera que tengan cierta forma sintáctica en particular.
- La normalización de gramáticas libres de contexto es útil para homogeneizar la forma de las producciones así como para optimizar los procesos de derivación de cadenas.



# Introducción

## Normalización de gramáticas

- La normalización consiste en transformar todas las producciones de una gramática de manera que tengan cierta forma sintáctica en particular.
- La normalización de gramáticas libres de contexto es útil para homogeneizar la forma de las producciones así como para optimizar los procesos de derivación de cadenas.
- Con las formas normales se facilita la solución al problema de la pertenencia.



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  y una palabra  $u$ , ¿Se cumple  $u \in L(G)$  ?  
Es decir, pertenece  $u$  al lenguaje generado por  $G$ .



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  y una palabra  $u$ , ¿Se cumple  $u \in L(G)$  ?  
Es decir, pertenece  $u$  al lenguaje generado por  $G$ .
- Si la palabra  $u$  es generada por  $G$  la construcción de un árbol de derivación terminará eventualmente.



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  y una palabra  $u$ , ¿Se cumple  $u \in L(G)$  ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .
- Si la palabra  $u$  es generada por  $G$  la construcción de un árbol de derivación terminará eventualmente.
- En caso contrario **no** podemos saber cuando parar en la construcción del árbol.



# VARIABLES INÚTILES

## Definición

- Una variable  $A$  es **accesible** o **alcanzable** si existen  $u, v \in (V \cup T)^*$  tales que  $S \xrightarrow{*} uAv$ . Obsérvese que según esta definición  $S$  es alcanzable.



# VARIABLES INÚTILES

## Definición

- Una variable  $A$  es **accesible** o **alcanzable** si existen  $u, v \in (V \cup T)^*$  tales que  $S \rightarrow^* uAv$ . Obsérvese que según esta definición  $S$  es alcanzable.
- Una variable  $A$  es **productiva** o **terminable** si existe  $w \in T^*$  tal que  $A \rightarrow^* w$ . En particular si  $A \rightarrow \varepsilon$  es una producción entonces  $A$  es productiva.



# VARIABLES INÚTILES

## Definición

- Una variable  $A$  es **accesible** o **alcanzable** si existen  $u, v \in (V \cup T)^*$  tales que  $S \rightarrow^* uAv$ . Obsérvese que según esta definición  $S$  es alcanzable.
- Una variable  $A$  es **productiva** o **terminable** si existe  $w \in T^*$  tal que  $A \rightarrow^* w$ . En particular si  $A \rightarrow \varepsilon$  es una producción entonces  $A$  es productiva.
- Una variable  $A$  es **inútil** si no es alcanzable o no es productiva.



# Algoritmo para hallar variables productivas

## Transformaciones en GLC

$$\text{Prod} := \{A \in V \mid A \rightarrow w \in P, w \in T^*\}$$


# Algoritmo para hallar variables productivas

Transformaciones en GLC

$$\text{Prod} := \{A \in V \mid A \rightarrow w \in P, w \in T^*\}$$

**repetir**

$$\text{Prod} := \text{Prod} \cup \{A \in V \mid A \rightarrow w, w \in (T \cup \text{Prod})^*\}$$


# Algoritmo para hallar variables productivas

Transformaciones en GLC

$Prod := \{A \in V \mid A \rightarrow w \in P, w \in T^*\}$

**repetir**

$Prod := Prod \cup \{A \in V \mid A \rightarrow w, w \in (T \cup Prod)^*\}$

**hasta** que no se añaden nuevas variables a  $Prod$ .



# Ejemplo

## Variables productivas

$$\begin{array}{l} S \rightarrow ACD \mid bBd \mid ab \\ A \rightarrow aB \mid aA \mid C \\ B \rightarrow aDS \mid aB \\ C \rightarrow aCS \mid CB \mid CC \\ D \rightarrow bD \mid ba \\ E \rightarrow AB \mid aDb \end{array}$$



# Ejemplo

## Variables productivas

$$\begin{aligned} S &\rightarrow ACD \mid bBd \mid ab \\ A &\rightarrow aB \mid aA \mid C \\ B &\rightarrow aDS \mid aB \\ C &\rightarrow aCS \mid CB \mid CC \\ D &\rightarrow bD \mid ba \\ E &\rightarrow AB \mid aDb \end{aligned}$$

$$Prod = \{S, D, B, E, A\}$$



# Ejemplo

## Eliminación de variables improductivas

C es la única variable improductiva, se elimina junto con todas las reglas donde figure:

$$\begin{aligned} S &\rightarrow bBd \mid ab \\ A &\rightarrow aB \mid aA \\ B &\rightarrow aDS \mid aB \\ D &\rightarrow bD \mid ba \\ E &\rightarrow AB \mid aDb \end{aligned}$$



# Algoritmo para hallar variables accesibles

Transformaciones en GLC

$$Acc := \{S\}$$



# Algoritmo para hallar variables accesibles

Transformaciones en GLC

$$Acc := \{S\}$$

**repetir**

$$Acc := Acc \cup \{A \in V \mid \exists B \rightarrow uAv \in P, B \in Acc, u, v \in (V \cup T)^*\}$$



# Algoritmo para hallar variables accesibles

Transformaciones en GLC

$$Acc := \{S\}$$

**repetir**

$$Acc := Acc \cup \{A \in V \mid \exists B \rightarrow uAv \in P, B \in Acc, u, v \in (V \cup T)^*\}$$

**hasta** que no se añaden nuevas variables a  $Acc$ .



# Ejemplo

## Variables accesibles

$$\begin{array}{l} S \rightarrow aS \mid AaB \mid ACS \\ A \rightarrow aS \mid AaB \mid AC \\ B \rightarrow bB \mid DB \mid BB \\ C \rightarrow aDa \mid ABD \mid ab \\ D \rightarrow aD \mid DD \mid ab \\ E \rightarrow FF \mid aa \\ F \rightarrow aE \mid EF \end{array}$$



# Ejemplo

## Variables accesibles

$$\begin{array}{l} S \rightarrow aS \mid AaB \mid ACS \\ A \rightarrow aS \mid AaB \mid AC \\ B \rightarrow bB \mid DB \mid BB \\ C \rightarrow aDa \mid ABD \mid ab \\ D \rightarrow aD \mid DD \mid ab \\ E \rightarrow FF \mid aa \\ F \rightarrow aE \mid EF \end{array}$$

$$Acc = \{S, A, B, C, D\}$$



# Ejemplo

## Eliminación de variables inaccesibles

$E, F$  son variables inaccesibles, se eliminan junto con todas las reglas donde figuren:

$$\begin{aligned} S &\rightarrow aS \mid AaB \mid ACS \\ A &\rightarrow aS \mid AaB \mid AC \\ B &\rightarrow bB \mid DB \mid BB \\ C &\rightarrow aDa \mid ABD \mid ab \\ D &\rightarrow aD \mid DD \mid ab \end{aligned}$$



# Eliminación de variables inútiles

## Transformaciones en GLC

Para eliminar variables inútiles se aplican los dos algoritmos anteriores  
**en el siguiente orden:**



# Eliminación de variables inútiles

## Transformaciones en GLC

Para eliminar variables inútiles se aplican los dos algoritmos anteriores  
**en el siguiente orden:**

- Eliminar variables no productivas.



# Eliminación de variables inútiles

## Transformaciones en GLC

Para eliminar variables inútiles se aplican los dos algoritmos anteriores  
**en el siguiente orden:**

- Eliminar variables no productivas.
- Eliminar variables no accesibles.



# Eliminación de variables inútiles

## Importancia del orden de los algoritmos

Si se aplican los algoritmos en orden inverso el resultado puede ser una gramática que aún contenga variable inútiles.



# Eliminación de variables inútiles

## Importancia del orden de los algoritmos

Si se aplican los algoritmos en orden inverso el resultado puede ser una gramática que aún contenga variable inútiles.

$$S \rightarrow a \mid AB, \quad A \rightarrow aA \mid \epsilon$$



# Eliminación de variables inútiles

## Importancia del orden de los algoritmos

Si se aplican los algoritmos en orden inverso el resultado puede ser una gramática que aún contenga variable inútiles.

$$S \rightarrow a \mid AB, \quad A \rightarrow aA \mid \epsilon$$

Al eliminar primero las variables no accesibles se obtiene la misma gramática, al ser todas las variables accesibles.



# Eliminación de variables inútiles

## Importancia del orden de los algoritmos

Si se aplican los algoritmos en orden inverso el resultado puede ser una gramática que aún contenga variable inútiles.

$$S \rightarrow a \mid AB, \quad A \rightarrow aA \mid \epsilon$$

Al eliminar primero las variables no accesibles se obtiene la misma gramática, al ser todas las variables accesibles. Posteriormente al eliminar variables improductivas resulta

$$S \rightarrow a, \quad A \rightarrow aA \mid \epsilon$$



# Eliminación de variables inútiles

## Importancia del orden de los algoritmos

Si se aplican los algoritmos en orden inverso el resultado puede ser una gramática que aún contenga variable inútiles.

$$S \rightarrow a \mid AB, \quad A \rightarrow aA \mid \varepsilon$$

Al eliminar primero las variables no accesibles se obtiene la misma gramática, al ser todas las variables accesibles. Posteriormente al eliminar variables improductivas resulta

$$S \rightarrow a, \quad A \rightarrow aA \mid \varepsilon$$

y claramente  $A$  es inútil por ser inaccesible.



# Eliminación de $\epsilon$ -producciones

Transformaciones en GLC

- Una producción de la forma  $A \rightarrow \epsilon$  es una  $\epsilon$ -producción



# Eliminación de $\epsilon$ -producciones

## Transformaciones en GLC

- Una producción de la forma  $A \rightarrow \epsilon$  es una  $\epsilon$ -producción
- Una variable  $A$  se llama **anulable** si  $A \rightarrow^* \epsilon$ .



# Algoritmo para hallar variables anulables

Transformaciones en GLC

$$Anul := \{ A \in V \mid A \rightarrow \varepsilon \in P \}$$



# Algoritmo para hallar variables anulables

Transformaciones en GLC

$$Anul := \{A \in V \mid A \rightarrow \varepsilon \in P\}$$

**repetir**

$$Anul := Anul \cup \{A \in V \mid \exists \ A \rightarrow w \in P, \ w \in Anul^*\}$$



# Algoritmo para hallar variables anulables

Transformaciones en GLC

$$Anul := \{A \in V \mid A \rightarrow \varepsilon \in P\}$$

**repetir**

$$Anul := Anul \cup \{A \in V \mid \exists \ A \rightarrow w \in P, \ w \in Anul^*\}$$

**hasta** que no se añaden nuevas variables a  $Anul$ .



# Eliminación de $\epsilon$ -producciones

## Transformaciones en GLC

- Eliminar todas las  $\epsilon$ -producciones.



# Eliminación de $\varepsilon$ -producciones

## Transformaciones en GLC

- Eliminar todas las  $\varepsilon$ -producciones.
- Para cada producción de la forma  $A \rightarrow w_1 \dots w_n$  agregar las producciones  $A \rightarrow v_1 \dots v_n$  donde:



# Eliminación de $\varepsilon$ -producciones

## Transformaciones en GLC

- Eliminar todas las  $\varepsilon$ -producciones.
- Para cada producción de la forma  $A \rightarrow w_1 \dots w_n$  agregar las producciones  $A \rightarrow v_1 \dots v_n$  donde:
  - ▶  $v_i = w_i$  si  $w_i \notin Anul$ .



# Eliminación de $\varepsilon$ -producciones

## Transformaciones en GLC

- Eliminar todas las  $\varepsilon$ -producciones.
- Para cada producción de la forma  $A \rightarrow w_1 \dots w_n$  agregar las producciones  $A \rightarrow v_1 \dots v_n$  donde:
  - ▶  $v_i = w_i$  si  $w_i \notin Anul$ .
  - ▶  $v_i = w_i$  ó  $v_i = \varepsilon$  si  $w_i \in Anul$ .



# Eliminación de $\varepsilon$ -producciones

## Transformaciones en GLC

- Eliminar todas las  $\varepsilon$ -producciones.
- Para cada producción de la forma  $A \rightarrow w_1 \dots w_n$  agregar las producciones  $A \rightarrow v_1 \dots v_n$  donde:
  - ▶  $v_i = w_i$  si  $w_i \notin Anul$ .
  - ▶  $v_i = w_i$  ó  $v_i = \varepsilon$  si  $w_i \in Anul$ .
  - ▶ Verificando que no se anulen todos los  $v_i$  al mismo tiempo.



# Ejemplo

## Eliminación de $\varepsilon$ -producciones

$$\begin{aligned}S &\rightarrow AB \mid ACA \mid ab \\A &\rightarrow aAa \mid B \mid CD \\B &\rightarrow bB \mid bA \\C &\rightarrow cC \mid \varepsilon \\D &\rightarrow aDc \mid CC \mid ABb\end{aligned}$$



# Ejemplo

## Eliminación de $\varepsilon$ -producciones

$$\begin{aligned}S &\rightarrow AB \mid ACA \mid ab \\A &\rightarrow aAa \mid B \mid CD \\B &\rightarrow bB \mid bA \\C &\rightarrow cC \mid \varepsilon \\D &\rightarrow aDc \mid CC \mid ABb\end{aligned}$$

$$Anul = \{C, D, A, S\}$$

# Ejemplo

## Eliminación de $\varepsilon$ -producciones

Al eliminar la producción  $C \rightarrow \varepsilon$  se obtiene la siguiente gramática:

$$\begin{aligned} S &\rightarrow AB \mid ACA \mid ab \mid B \mid CA \mid AA \mid AC \mid A \mid C \mid \varepsilon \\ A &\rightarrow aAa \mid B \mid CD \mid aa \mid C \mid D \\ B &\rightarrow bB \mid bA \mid b \\ C &\rightarrow cC \mid c \\ D &\rightarrow aDc \mid CC \mid ABb \mid ac \mid C \mid Bb \end{aligned}$$



# Acerca de la palabra vacía

## Transformaciones en GLC

- Si originalmente se tenía  $\varepsilon \in L(G)$  la eliminación de  $\varepsilon$  producciones genera una gramática que no genera a  $\varepsilon$ .



# Acerca de la palabra vacía

## Transformaciones en GLC

- Si originalmente se tenía  $\varepsilon \in L(G)$  la eliminación de  $\varepsilon$ -producciones genera una gramática que no genera a  $\varepsilon$ .
- Es posible saber si se pierde la palabra vacía al eliminar  $\varepsilon$ -producciones verificando si  $S \in Anul.$



# Acerca de la palabra vacía

## Transformaciones en GLC

- Si originalmente se tenía  $\varepsilon \in L(G)$  la eliminación de  $\varepsilon$  producciones genera una gramática que no genera a  $\varepsilon$ .
- Es posible saber si se pierde la palabra vacía al eliminar  $\varepsilon$ -producciones verificando si  $S \in Anul.$
- Si se quiere recuperar a  $\varepsilon$  debe agregarse un nuevo símbolo inicial  $S'$  y las producciones  $S' \rightarrow S$  y  $S' \rightarrow \varepsilon$ .



# Acerca de la palabra vacía

## Transformaciones en GLC

- Si originalmente se tenía  $\varepsilon \in L(G)$  la eliminación de  $\varepsilon$  producciones genera una gramática que no genera a  $\varepsilon$ .
- Es posible saber si se pierde la palabra vacía al eliminar  $\varepsilon$ -producciones verificando si  $S \in Anul.$
- Si se quiere recuperar a  $\varepsilon$  debe agregarse un nuevo símbolo inicial  $S'$  y las producciones  $S' \rightarrow S$  y  $S' \rightarrow \varepsilon$ .
- $S' \rightarrow \varepsilon$  es la única  $\varepsilon$ -producción permitida.



# Eliminación de producciones unitarias

## Transformaciones en GLC

- Una producción de la forma  $A \rightarrow B$  donde  $A$  y  $B$  son ambas variables se llama **producción unitaria**.



# Eliminación de producciones unitarias

## Transformaciones en GLC

- Una producción de la forma  $A \rightarrow B$  donde  $A$  y  $B$  son ambas variables se llama **producción unitaria**.
- El **conjunto unitario** de  $A$  se define como sigue:

$$\text{Unit}(A) = \{B \in V \mid A \xrightarrow{*} B \text{ usando sólo prod. unitarias}\}$$



# Eliminación de producciones unitarias

## Transformaciones en GLC

- Una producción de la forma  $A \rightarrow B$  donde  $A$  y  $B$  son ambas variables se llama **producción unitaria**.

- El **conjunto unitario** de  $A$  se define como sigue:

$$\text{Unit}(A) = \{B \in V \mid A \xrightarrow{*} B \text{ usando sólo prod. unitarias}\}$$

- Por definición se tiene  $A \in \text{Unit}(A)$ .



# Algoritmo para hallar el conjunto $Unit(A)$

Transformaciones en GLC

$$Unit(A) := \{A\}$$



# Algoritmo para hallar el conjunto $Unit(A)$

Transformaciones en GLC

$$Unit(A) := \{A\}$$

**repetir**

$$Unit(A) := Unit(A) \cup \{B \in V \mid \exists C \rightarrow B, C \in Unit(A)\}$$



# Algoritmo para hallar el conjunto $Unit(A)$

Transformaciones en GLC

$Unit(A) := \{A\}$

**repetir**

$Unit(A) := Unit(A) \cup \{B \in V \mid \exists C \rightarrow B, C \in Unit(A)\}$

**hasta** que no se añaden nuevas variables a  $Unit(A)$ .



# Eliminación de producciones unitarias

## Transformaciones en GLC

- Para cada  $B \in Unit(A)$  y cada producción  $A \rightarrow w$  agregar la producción

$$B \rightarrow w$$



# Eliminación de producciones unitarias

## Transformaciones en GLC

- Para cada  $B \in Unit(A)$  y cada producción  $A \rightarrow w$  agregar la producción

$$B \rightarrow w$$

- Eliminar todas las producciones unitarias



# Ejemplo

## Eliminación de producciones unitarias

$$\begin{aligned}S &\rightarrow AS \mid AA \mid BA \mid \varepsilon \\A &\rightarrow aA \mid a \\B &\rightarrow bB \mid bC \mid C \\C &\rightarrow aA \mid bA \mid B \mid ab\end{aligned}$$



# Ejemplo

## Eliminación de producciones unitarias

$$\begin{aligned}S &\rightarrow AS \mid AA \mid BA \mid \varepsilon \\A &\rightarrow aA \mid a \\B &\rightarrow bB \mid bC \mid C \\C &\rightarrow aA \mid bA \mid B \mid ab\end{aligned}$$

Los conjuntos unitarios para cada variable son:

$$Unit(S) = \{S\} \quad Unit(A) = \{A\}$$

$$Unit(B) = \{B, C\} \quad Unit(C) = \{C, B\}$$



# Ejemplo

## Eliminación de producciones unitarias

La gramática obtenida al eliminar producciones unitarias es:

$$\begin{aligned}S &\rightarrow AS \mid AA \mid BA \mid \varepsilon \\A &\rightarrow aA \mid a \\B &\rightarrow bB \mid bC \mid aA \mid bA \mid ab \\C &\rightarrow aA \mid bA \mid ab \mid bB \mid bC\end{aligned}$$



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  sin  $\varepsilon$ -producciones ni producciones unitarias y una palabra  $u$  de longitud  $n$ , ¿Se cumple  $u \in L(G)$ ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  sin  $\varepsilon$ -producciones ni producciones unitarias y una palabra  $u$  de longitud  $n$ , ¿Se cumple  $u \in L(G)$ ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .
- Si la palabra  $u$  es generada por  $G$  la construcción de un árbol de derivación terminará eventualmente.



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  sin  $\varepsilon$ -producciones ni producciones unitarias y una palabra  $u$  de longitud  $n$ , ¿Se cumple  $u \in L(G)$ ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .
- Si la palabra  $u$  es generada por  $G$  la construcción de un árbol de derivación terminará eventualmente.
- En caso contrario basta con construir el árbol hasta el nivel  $2n - 1$  para concluir que  $u \notin L(G)$ .



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  sin  $\varepsilon$ -producciones ni producciones unitarias y una palabra  $u$  de longitud  $n$ , ¿Se cumple  $u \in L(G)$ ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .
- Si la palabra  $u$  es generada por  $G$  la construcción de un árbol de derivación terminará eventualmente.
- En caso contrario basta con construir el árbol hasta el nivel  $2n - 1$  para concluir que  $u \notin L(G)$ .
- En cada paso se obtiene un nuevo terminal (a lo mas  $n$  pasos) o se aumenta la longitud de la palabra en 1 ( a lo mas  $n - 1$  pasos)



# Forma Normal de Chomsky

## Definición

Una GLC  $G$  está en forma normal de Chomsky (FNC) si:

- $G$  no contiene variables inútiles.



# Forma Normal de Chomsky

## Definición

Una GLC  $G$  está en forma normal de Chomsky (FNC) si:

- $G$  no contiene variables inútiles.
- $G$  no contiene producciones unitarias ni  $\varepsilon$ -producciones (salvo  $S \rightarrow \varepsilon$ )



# Forma Normal de Chomsky

## Definición

Una GLC  $G$  está en forma normal de Chomsky (FNC) si:

- $G$  no contiene variables inútiles.
- $G$  no contiene producciones unitarias ni  $\varepsilon$ -producciones (salvo  $S \rightarrow \varepsilon$ )
- Todas las producciones son de la forma:

$$A \rightarrow BC \text{ ó } A \rightarrow a$$

donde  $B, C \in V$ ,  $a \in T$



# Transformación a forma normal de Chomsky

## Transformaciones en GLC

Cualquier GLC es equivalente a una gramática en FNC, lo cual se logra como sigue:

- Eliminar las variables inútiles.



# Transformación a forma normal de Chomsky

## Transformaciones en GLC

Cualquier GLC es equivalente a una gramática en FNC, lo cual se logra como sigue:

- Eliminar las variables inútiles.
- Eliminar las  $\varepsilon$ -producciones (salvo  $S \rightarrow \varepsilon$ )



# Transformación a forma normal de Chomsky

## Transformaciones en GLC

Cualquier GLC es equivalente a una gramática en FNC, lo cual se logra como sigue:

- Eliminar las variables inútiles.
- Eliminar las  $\varepsilon$ -producciones (salvo  $S \rightarrow \varepsilon$ )
- Eliminar producciones unitarias.



# Transformación a forma normal de Chomsky

## Transformaciones en GLC

Cualquier GLC es equivalente a una gramática en FNC, lo cual se logra como sigue:

- Eliminar las variables inútiles.
- Eliminar las  $\varepsilon$ -producciones (salvo  $S \rightarrow \varepsilon$ )
- Eliminar producciones unitarias.
- Las producciones restantes son todas de la forma  $A \rightarrow a$  con  $a \in T$  ó  $A \rightarrow w$  con  $|w| \geq 2$



# Eliminación de producciones $A \rightarrow w, |w| \geq 2$

## Forma Normal de Chomsky

- Hacer lo siguiente para cada producción  $P$  de la forma  $A \rightarrow \alpha_1\alpha_2\dots\alpha_n$  con  $\alpha_i \in V \cup T, n \geq 2$  (si  $n = 2$ , al menos uno de  $\alpha_1, \alpha_2$  debe ser terminal, pues si no la producción ya es válida para FNC)



# Eliminación de producciones $A \rightarrow w, |w| \geq 2$

## Forma Normal de Chomsky

- Hacer lo siguiente para cada producción  $P$  de la forma  $A \rightarrow \alpha_1\alpha_2\dots\alpha_n$  con  $\alpha_i \in V \cup T, n \geq 2$  (si  $n = 2$ , al menos uno de  $\alpha_1, \alpha_2$  debe ser terminal, pues si no la producción ya es válida para FNC)
- Si  $\alpha_i \in T$ , digamos  $\alpha_i = a$ , entonces



# Eliminación de producciones $A \rightarrow w, |w| \geq 2$

## Forma Normal de Chomsky

- Hacer lo siguiente para cada producción  $P$  de la forma  $A \rightarrow \alpha_1\alpha_2\dots\alpha_n$  con  $\alpha_i \in V \cup T, n \geq 2$  (si  $n = 2$ , al menos uno de  $\alpha_1, \alpha_2$  debe ser terminal, pues si no la producción ya es válida para FNC)
  - Si  $\alpha_i \in T$ , digamos  $\alpha_i = a$ , entonces
    - ▶ Agregar la producción  $T_a \rightarrow a$ , donde  $T_a$  es una nueva variable.



# Eliminación de producciones $A \rightarrow w, |w| \geq 2$

## Forma Normal de Chomsky

- Hacer lo siguiente para cada producción  $P$  de la forma  $A \rightarrow \alpha_1\alpha_2\dots\alpha_n$  con  $\alpha_i \in V \cup T, n \geq 2$  (si  $n = 2$ , al menos uno de  $\alpha_1, \alpha_2$  debe ser terminal, pues si no la producción ya es válida para FNC)
  - Si  $\alpha_i \in T$ , digamos  $\alpha_i = a$ , entonces
    - ▶ Agregar la producción  $T_a \rightarrow a$ , donde  $T_a$  es una nueva variable.
    - ▶ Cambiar  $\alpha_i$  por  $T_a$  en la producción  $P$



# Eliminación de producciones $A \rightarrow w, |w| \geq 2$

## Forma Normal de Chomsky

- Hacer lo siguiente para cada producción  $P$  de la forma  $A \rightarrow \alpha_1\alpha_2\dots\alpha_n$  con  $\alpha_i \in V \cup T, n \geq 2$  (si  $n = 2$ , al menos uno de  $\alpha_1, \alpha_2$  debe ser terminal, pues si no la producción ya es válida para FNC)
  - Si  $\alpha_i \in T$ , digamos  $\alpha_i = a$ , entonces
    - ▶ Agregar la producción  $T_a \rightarrow a$ , donde  $T_a$  es una nueva variable.
    - ▶ Cambiar  $\alpha_i$  por  $T_a$  en la producción  $P$
- Para cada producción  $P$  de la forma  $A \rightarrow B_1B_2\dots B_m$  con  $B_i \in V, m \geq 3$



# Eliminación de producciones $A \rightarrow w, |w| \geq 2$

## Forma Normal de Chomsky

- Hacer lo siguiente para cada producción  $P$  de la forma  $A \rightarrow \alpha_1\alpha_2\dots\alpha_n$  con  $\alpha_i \in V \cup T, n \geq 2$  (si  $n = 2$ , al menos uno de  $\alpha_1, \alpha_2$  debe ser terminal, pues si no la producción ya es válida para FNC)
- Si  $\alpha_i \in T$ , digamos  $\alpha_i = a$ , entonces
  - ▶ Agregar la producción  $T_a \rightarrow a$ , donde  $T_a$  es una nueva variable.
  - ▶ Cambiar  $\alpha_i$  por  $T_a$  en la producción  $P$
- Para cada producción  $P$  de la forma  $A \rightarrow B_1B_2\dots B_m$  con  $B_i \in V, m \geq 3$ 
  - ▶ Agregar  $(m - 2)$  nuevas variables  $D_1, \dots, D_{m-2}$  y reemplazar a  $P$  con las siguientes producciones:

$$A \rightarrow B_1D_1, D_1 \rightarrow B_2D_2, \dots, D_{m-2} \rightarrow B_{m-1}B_m$$



# Ejemplo

Simulación de producciones  $A \rightarrow w_1 w_2 \dots w_n, n \geq 2$

La producción  $A \rightarrow abBaC$  se simula con producciones simples y binarias como sigue:

- Agregamos las nuevas variables  $T_a$ ,  $T_b$  y las producciones

$$A \rightarrow T_a T_b B T_a C, \quad T_a \rightarrow a, \quad T_b \rightarrow b$$



# Ejemplo

Simulación de producciones  $A \rightarrow w_1 w_2 \dots w_n, n \geq 2$

La producción  $A \rightarrow abBaC$  se simula con producciones simples y binarias como sigue:

- Agregamos las nuevas variables  $T_a, T_b$  y las producciones

$$A \rightarrow T_a T_b B T_a C, \quad T_a \rightarrow a, \quad T_b \rightarrow b$$

- Para simular la producción  $A \rightarrow T_a T_b B T_a C$  agregamos nuevas variables  $D_1, D_2, D_3$  y las producciones binarias necesarias obteniendo finalmente:

$$A \rightarrow T_a D_1, \quad D_1 \rightarrow T_b D_2, \quad D_2 \rightarrow B D_3, \quad D_3 \rightarrow T_a C$$

$$T_a \rightarrow a, \quad T_b \rightarrow b$$



# Forma normal de Chomsky

## Ejemplo

Transformar la siguiente gramática a FNC:

$$S \rightarrow AB \mid aBC \mid SBS$$

$$A \rightarrow aA \mid C$$

$$B \rightarrow bbB \mid b$$

$$C \rightarrow cC \mid \varepsilon$$

# Forma normal de Chomsky

## Ejemplo

La gramática equivalente en FNC es:

$$S \rightarrow AB \mid T_a D_1 \mid SD_2 \mid T_a B \mid T_b D_3 \mid b$$

$$A \rightarrow T_a A \mid T_c C \mid a \mid c$$

$$B \rightarrow T_b D_3 \mid b$$

$$C \rightarrow T_c C \mid c$$

$$D_1 \rightarrow BC$$

$$D_2 \rightarrow BS$$

$$D_3 \rightarrow T_b B$$

$$T_a \rightarrow a$$

$$T_b \rightarrow b$$

$$T_c \rightarrow c$$



# Forma Normal de Greibach (FNG)

## Definición

Una GLC  $G$  está en forma normal de Greibach (FNG) si:

- La variable inicial  $S$  no es recursiva, es decir, no figura en el lado derecho de las producciones.



# Forma Normal de Greibach (FNG)

## Definición

Una GLC  $G$  está en forma normal de Greibach (FNG) si:

- La variable inicial  $S$  no es recursiva, es decir, no figura en el lado derecho de las producciones.
- $G$  no tiene variables inútiles ni  $\varepsilon$ -producciones.



# Forma Normal de Greibach (FNG)

## Definición

Una GLC  $G$  está en forma normal de Greibach (FNG) si:

- La variable inicial  $S$  no es recursiva, es decir, no figura en el lado derecho de las producciones.
- $G$  no tiene variables inútiles ni  $\varepsilon$ -producciones.
- Todas las producciones son de la forma  $A \rightarrow a\alpha$ , con  $\alpha \in V^*$ .



# Transformación a forma normal de Greibach

## Transformaciones en GLC

- Cualquier GLC es equivalente a una gramática en FNG.



# Transformación a forma normal de Greibach

## Transformaciones en GLC

- Cualquier GLC es equivalente a una gramática en FNG.
- El proceso requiere que la gramática esté en Forma Normal de Chomsky.



# Transformación a forma normal de Greibach

## Transformaciones en GLC

- Cualquier GLC es equivalente a una gramática en FNG.
- El proceso requiere que la gramática esté en Forma Normal de Chomsky.
- Con una gramática en forma normal de Greibach se simplifica el problema de la pertenencia.



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  en forma normal de Greibach y una palabra  $u$  de longitud  $n$ , ¿Se cumple  $u \in L(G)$ ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  en forma normal de Greibach y una palabra  $u$  de longitud  $n$ , ¿Se cumple  $u \in L(G)$ ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .
- Si  $u$  es generada por  $G$  la construcción de un árbol de derivación terminará eventualmente.



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  en forma normal de Greibach y una palabra  $u$  de longitud  $n$ , ¿Se cumple  $u \in L(G)$ ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .
- Si  $u$  es generada por  $G$  la construcción de un árbol de derivación terminará eventualmente.
- En caso contrario basta con construir el árbol hasta el nivel  $n$  para concluir que  $u \notin L(G)$ .



# Problema de la pertenencia

## Normalización de gramáticas

- Dada una gramática  $G$  en forma normal de Greibach y una palabra  $u$  de longitud  $n$ , ¿Se cumple  $u \in L(G)$ ? Es decir, pertenece  $u$  al lenguaje generado por  $G$ .
- Si  $u$  es generada por  $G$  la construcción de un árbol de derivación terminará eventualmente.
- En caso contrario basta con construir el árbol hasta el nivel  $n$  para concluir que  $u \notin L(G)$ .
- En cada paso se obtiene un nuevo terminal (a lo mas  $n$  pasos) y por lo general hay una menor ramificación.

