

# Formas normales

Clase 17

IIC 2223

Prof. Cristian Riveros

# ¿qué es una forma normal?

## Ejemplo: polinomios

Un polinomio cualquiera:

$$p(x) := (x^3 \cdot ((x - 2) + 3x^2) - (3x^5 - 2x^2)) \cdot 2x + 7$$

Un polinomio cualquiera

cuando **planeamos hacer un algoritmo** sobre polinomios:

$$p(x) := 2x^5 + 4x^3 - 4x + 7$$

Formas normales son útiles en computación  
para **estudiar** un objeto y **diseñar** algoritmos.

# Outline

Forma normal de Chomsky

Formal normal de Greibach

# Outline

Forma normal de Chomsky

Formal normal de Greibach

# Forma normal de Chomsky

## Definición

Una gramática  $\mathcal{G}$  está en **forma normal de Chomsky** (CNF) si todas sus reglas son de la forma:

- $X \rightarrow YZ$
- $X \rightarrow a$

# Forma normal de Chomsky

¿cuáles gramáticas están en CNF?

■  $S \rightarrow a S b \mid \epsilon$



■  $A \rightarrow A B \mid a \mid \epsilon$   
 $B \rightarrow B A \mid b \mid \epsilon$



■  $S \rightarrow AB \mid AC \mid SS$   
 $C \rightarrow SB$   
 $A \rightarrow a$   
 $B \rightarrow b$



# Forma normal de Chomsky

## Definición

Una gramática  $\mathcal{G}$  está en **forma normal de Chomsky** (CNF) si todas sus reglas son de la forma:

- $X \rightarrow YZ$
- $X \rightarrow a$

Si  $\mathcal{G}$  está en CNF:

- ¿puede aceptar la palabra  $\epsilon$ ?
- ¿puede tener reglas **unitarias**?
- ¿puede tener reglas **en vacío**?

# Toda gramática se puede convertir en CNF

Sea  $\mathcal{G} = (V, \Sigma, P, S)$  una CFG tal que  $\epsilon \notin \mathcal{L}(\mathcal{G})$ .

- Primero, suponga que  $\mathcal{G}$  no contiene reglas en vacío o unitarias.
- Por lo tanto, todas las reglas en  $\mathcal{G}$  son de la forma:
  - $X \rightarrow \gamma$  para  $|\gamma| \geq 2$
  - $X \rightarrow a$

¿cómo transformamos  $\mathcal{G}$  en **forma normal de Chomsky**?



# Hacia la forma normal de Chomsky

Sea una gramática  $\mathcal{G}$  donde las reglas son de la forma:

- $X \rightarrow \gamma$  para  $|\gamma| \geq 2$
- $X \rightarrow a$

Paso 1: Convertir todas las reglas a la forma:

- $X \rightarrow Y_1 Y_2 \dots Y_k$  para  $k \geq 2$
- $X \rightarrow a$

Paso 2: Convertir todas las reglas a la forma:

- $X \rightarrow YZ$
- $X \rightarrow a$

¿cómo hacemos el Paso 1? ¿y el Paso 2?

# Hacia la forma normal de Chomsky (Paso 1)

## Paso 1

Convertir todas las reglas a la forma:

- $X \rightarrow Y_1 Y_2 \dots Y_k$  para  $k \geq 2$
- $X \rightarrow a$

Solución:

- Para cada  $a \in \Sigma$ ,  
agregar una nueva variable  $X_a$  y una regla  $X_a \rightarrow a$ .
- Reemplazar todas las ocurrencias antiguas de  $a$  por  $X_a$ .

# Hacia la forma normal de Chomsky (Paso 1)

## Ejemplo del Paso 1

$$S \rightarrow a S b \mid ab$$

---

$$S \rightarrow A S B \mid AB$$
$$A \rightarrow a$$
$$B \rightarrow b$$

# Hacia la forma normal de Chomsky (Paso 1)

## Paso 1

Convertir todas las reglas a la forma  $X \rightarrow Y_1 Y_2 \dots Y_k$  para  $k \geq 2$  o  $X \rightarrow a$ .

Solución:

- Para cada  $a \in \Sigma$ ,  
agregar una nueva variable  $X_a$  y una regla  $X_a \rightarrow a$ .
- Reemplazar todas las ocurrencias antiguas de  $a$  por  $X_a$ .

## Correctitud

Si  $\mathcal{G}'$  es la gramática resultante, entonces se cumple que  $\mathcal{L}(\mathcal{G}') = \mathcal{L}(\mathcal{G})$ .

# Hacia la forma normal de Chomsky (Paso 2)

## Paso 2

Convertir todas las reglas a la forma:

- $X \rightarrow YZ$
- $X \rightarrow a$

## Solución:

Para cada regla  $p : X \rightarrow Y_1 Y_2 \dots Y_k$  con  $k \geq 3$ :

- Agregamos una **nueva** variable  $Z$ .
- Reemplazamos la regla  $p$  por **dos reglas**:

$$X \rightarrow Y_1 Z \quad \text{y} \quad Z \rightarrow Y_2 \dots Y_k$$

**Repetimos** este paso hasta llegar a la forma normal de Chomsky.

# Hacia la forma normal de Chomsky (Paso 2)

## Ejemplo del Paso 2 (continuación)

El resultado del Paso 1 es:

$$\begin{array}{lcl} S & \rightarrow & A S B \mid AB \\ A & \rightarrow & a \\ B & \rightarrow & b \end{array}$$

---

$$\begin{array}{lcl} S & \rightarrow & A Z \mid AB \\ Z & \rightarrow & S B \\ A & \rightarrow & a \\ B & \rightarrow & b \end{array}$$

# Hacia la forma normal de Chomsky (Paso 2)

## Paso 2

Convertir todas las reglas a la forma:  $X \rightarrow YZ$  o  $X \rightarrow a$ .

## Solución:

Para cada regla  $p : X \rightarrow Y_1 Y_2 \dots Y_k$  con  $k \geq 3$ :

- Agregamos una **nueva** variable  $Z$ .
- Reemplazamos la regla  $p$  por **dos reglas**:

$$X \rightarrow Y_1 Z \quad \text{y} \quad Z \rightarrow Y_2 \dots Y_k$$

**Repetimos** este paso hasta llegar a la forma normal de Chomsky.

## Correctitud

Si  $\mathcal{G}''$  es la gramática resultante, entonces se cumple que  $\mathcal{L}(\mathcal{G}'') = \mathcal{L}(\mathcal{G}')$ .

# Toda gramática se puede convertir en CNF

Sea  $\mathcal{G} = (V, \Sigma, P, S)$  una CFG tal que  $\epsilon \notin \mathcal{L}(\mathcal{G})$ .

## Teorema

Existe una gramática  $\mathcal{G}'$  en forma normal de Chomsky tal que:

$$\mathcal{L}(\mathcal{G}') = \mathcal{L}(\mathcal{G})$$

Si  $\mathcal{G}'$  no tiene reglas unitarias ni en vacío,  
entonces  $\mathcal{G}'$  es de **tamaño polinomial** con respecto a  $\mathcal{G}$ .



# Outline

Forma normal de Chomsky

Formal normal de Greibach

# Forma normal de Greibach

## Definición

Una gramática  $\mathcal{G}$  está en **forma normal de Greibach** (GNF) si todas sus reglas son de la forma:

$$\blacksquare X \rightarrow aY_1 \dots Y_k$$

para algún  $k \geq 0$ .

¿qué gramáticas están en GNF?

- $S \rightarrow aSb \mid SS \mid \epsilon$
- $S \rightarrow aSb \mid bSa \mid a \mid b$
- $S \rightarrow aSB \mid bSA \mid a \mid b$   
 $A \rightarrow a \quad B \rightarrow b$

¿para qué nos puede servir la **forma normal de Greibach**?

# Forma normal de Greibach

## Definición

Una gramática  $\mathcal{G}$  está en **forma normal de Greibach** (GNF) si todas sus reglas son de la forma:

$$\blacksquare X \rightarrow aY_1 \dots Y_k$$

para algún  $k \geq 0$ .

Si  $\mathcal{G}$  está en GNF, ¿es posible que  $\epsilon \in \mathcal{L}(\mathcal{G})$ ?

... desde ahora supondremos que  $\epsilon \notin \mathcal{L}(\mathcal{G})$ .

# Toda gramática se puede convertir en GNF

## Definición

Una gramática  $\mathcal{G}$  esta en “casi” GNF si todas sus reglas son de la forma:

- $X \rightarrow a\gamma$

con  $\gamma \in (V \cup \Sigma)^*$ .

## De “casi” GNF a GNF

- Para cada  $b \in \Sigma$ ,  
agregar un nueva variable  $X_b$  y una regla  $X_b \rightarrow b$ .
- Para cada regla  $X \rightarrow a\gamma$  y  $b \in \Sigma$ ,  
reemplazar todas las ocurrencias de  $b$  en  $\gamma$  por  $X_b$ .

Desde ahora, hablaremos de “casi” GNF como una **gramática en GNF**.

# ¿cómo convertimos una gramática a GNF?

## Ejemplo 1

$$S \rightarrow Xa \mid b$$

$$X \rightarrow Yc$$

$$Y \rightarrow dX \mid e$$

## Ejemplo 2

$$S \rightarrow Xa \mid b$$

$$X \rightarrow Yc$$

$$Y \rightarrow Xd \mid e$$

Necesitamos entender la **recursión por la izquierda** de las variables.

# Grafo de recursión (por la izquierda) de una gramática

## Definición

Para una gramática  $\mathcal{G} = (V, \Sigma, P, S)$  se define el **grafo de recursión**:

$$R_{\mathcal{G}} = (V, E)$$

tal que  $(X, Y) \in E$  si, y solo si,  $(X \rightarrow Y\alpha) \in P$  para algún  $\alpha \in (V \cup \Sigma)^*$ .


¿cuál es el grafo de recursión de los ejemplos anteriores?

$$S \rightarrow Xa \mid b$$
$$X \rightarrow Yc$$
$$Y \rightarrow dX \mid e$$
$$S \rightarrow Xa \mid b$$
$$X \rightarrow Yc$$
$$Y \rightarrow Xd \mid e$$

Si  $R_{\mathcal{G}}$  es **acíclico**, ¿cómo podemos convertir  $\mathcal{G}$  a GNF?

# Hacia la forma normal de Greibach (caso acíclico)

Sea  $\mathcal{G} = (V, \Sigma, P, S)$  tal que su grafo de recursión  $R_{\mathcal{G}} = (V, E)$  es acíclico.

1. Si  $E = \emptyset$ , entonces  
todas las reglas en  $P$  son de la forma  $X \rightarrow a\gamma$ . 
2. Si  $E \neq \emptyset$ , entonces sea  $X \in V$  tal que  
 $(Y, X) \in E$  para algún  $Y$  y  $(X, Z) \notin E$  para todo  $Z$ .

Construimos la gramática  $\mathcal{G}' = (V, \Sigma, P', S)$  tal que:

$$P' = (P \cup \{Y \rightarrow \alpha\beta \in P \mid Y \rightarrow X\beta \in P \wedge X \rightarrow \alpha \in P\}) \setminus \{Y \rightarrow X\beta \in P\}$$

Lema

$$\mathcal{L}(\mathcal{G}) = \mathcal{L}(\mathcal{G}') \text{ y } R_{\mathcal{G}'} = (V, E \setminus \{(Y, X) \in E \mid Y \in V\}).$$

Aplicando el caso 2. hasta llegar a 1.  
encontraremos una gramática equivalente a  $\mathcal{G}$  en GNF.

# Hacia la forma normal de Greibach (caso acíclico)

**input** : Gramática  $\mathcal{G} = (V, \Sigma, P, S)$  y  $R_{\mathcal{G}}$  grafo de recursión acíclico

**output**: Gramática  $\mathcal{G}'$  en GNF

**Function** GNF-aciclico ( $\mathcal{G}$ )

**let**  $R_{\mathcal{G}} := (V, E)$  grafo de recursión acíclico de  $\mathcal{G}$

**let**  $P' := P$

**while**  $E \neq \emptyset$  **do**

**let**  $X \in V : \exists Y. (Y, X) \in E \wedge \forall Z. (X, Z) \notin E$

$P' := P' \cup \{Y \rightarrow \alpha\beta \in P' \mid Y \rightarrow X\beta \in P' \wedge X \rightarrow \alpha \in P'\}$

$P' := P' \setminus \{Y \rightarrow X\beta \in P'\}$

**let**  $R_{\mathcal{G}} = (V, E)$  grafo de recursión acíclico de  $(V, \Sigma, P', S)$

**return**  $\mathcal{G}' = (V, \Sigma, P', S)$

Ejercicio: demuestre la correctitud del algoritmo.



# ¿cómo eliminamos los ciclos del grafo de recursión?

## Definición

Una gramática  $\mathcal{G}$  se dice **recursiva por la izquierda** si existe  $X \in V$  tal que:

$$X \Rightarrow^+ X\gamma \quad \text{para algún } \gamma \in (V \cup \Sigma)^*$$

¿cuáles gramáticas son recursivas por la izquierda?

$$S \rightarrow Xa \mid b$$

$$X \rightarrow Yc$$

$$Y \rightarrow dX \mid e$$

$$S \rightarrow Xa \mid b$$

$$X \rightarrow Yc$$

$$Y \rightarrow Xd \mid e$$

# ¿cómo eliminamos los ciclos del grafo de recursión?

## Definición

Una gramática  $\mathcal{G}$  se dice **recursiva por la izquierda** si existe  $X \in V$  tal que:

$$X \Rightarrow^+ X\gamma \quad \text{para algún } \gamma \in (V \cup \Sigma)^*$$

## Lema

$\mathcal{G}$  es **recursiva por la izquierda** si, y solo si,  $R_{\mathcal{G}}$  es **cíclico**.

Si **eliminamos la recursividad** por la izquierda de una gramática, entonces habremos **eliminado los ciclos** de  $R_{\mathcal{G}}$ .

(eliminar la recursividad por la izquierda también será **importante más adelante** para algoritmos de parsing)

# Recursión inmediata por la izquierda

Suponga que existe  $X \in V$  tal que:

$$X \rightarrow X\alpha_1 \mid \dots \mid X\alpha_m \mid \beta_1 \mid \dots \mid \beta_n$$

¿cómo podemos **eliminar** la recursión inmediata por la izquierda?

Considere la misma gramática pero **cambiando** las reglas de  $X$  por:

$$\begin{aligned} X &\rightarrow \beta_1 X' \mid \dots \mid \beta_n X' \\ X' &\rightarrow \alpha_1 X' \mid \dots \mid \alpha_m X' \mid \epsilon \end{aligned}$$

¿es la nueva gramática **recursiva inmediata por la izquierda** en  $X$ ?

# Recursión inmediata por la izquierda

Idea de eliminación de recursión inmediata

$$X \rightarrow X\alpha_1 \mid \dots \mid X\alpha_m \mid \beta_1 \mid \dots \mid \beta_n$$

---

$$\begin{aligned} X &\rightarrow \beta_1 X' \mid \dots \mid \beta_n X' \\ X' &\rightarrow \alpha_1 X' \mid \dots \mid \alpha_m X' \mid \epsilon \end{aligned}$$

Ejemplo de eliminar la **recursión inmediata**

$S \rightarrow Xa \mid b$	$S \rightarrow Xa \mid b$
$X \rightarrow Xc \mid d$	$X \rightarrow dX'$
	$X' \rightarrow cX' \mid \epsilon$

# Recursión inmediata por la izquierda

## Teorema

Sea  $\mathcal{G}$  una gramática tal que que existe  $X \in V$ :

$$X \rightarrow X\alpha_1 \mid \cdots \mid X\alpha_m \mid \beta_1 \mid \cdots \mid \beta_n$$

Sea  $\mathcal{G}'$  la misma gramática  $\mathcal{G}$  pero cambiando las reglas de  $X$  por:

$$\begin{aligned} X &\rightarrow \beta_1 X' \mid \cdots \mid \beta_n X' \\ X' &\rightarrow \alpha_1 X' \mid \cdots \mid \alpha_m X' \mid \epsilon \end{aligned}$$

Entonces  $\mathcal{L}(\mathcal{G}) = \mathcal{L}(\mathcal{G}')$ .

## Demostración

Una derivación por la izquierda de  $X$  en  $\mathcal{G}$ :

$$X \Rightarrow_{\text{lm}} X\alpha_{i_1} \Rightarrow_{\text{lm}} X\alpha_{i_2}\alpha_{i_1} \Rightarrow_{\text{lm}} \cdots \Rightarrow_{\text{lm}} X\alpha_{i_p}\alpha_{i_{p-1}}\cdots\alpha_{i_1} \Rightarrow_{\text{lm}} \beta_j\alpha_{i_p}\alpha_{i_{p-1}}\cdots\alpha_{i_1}$$

Una derivación por la derecha de  $X$  en  $\mathcal{G}'$  **equivalente**:

$$X \Rightarrow_{\text{rm}} \beta_j X' \Rightarrow_{\text{rm}} \beta_j \alpha_{i_p} X' \Rightarrow_{\text{rm}} \cdots \Rightarrow_{\text{rm}} \beta_j \alpha_{i_p} \cdots \alpha_{i_2} \alpha_{i_1} X' \Rightarrow_{\text{rm}} \beta_j \alpha_{i_p} \alpha_{i_{p-1}} \cdots \alpha_{i_1} \square$$

# Recursión por la izquierda no-inmediata

Considere la siguiente gramática **recursiva por la izquierda**:

$$\begin{aligned} S &\rightarrow Xa \mid b \\ X &\rightarrow Yc \\ Y &\rightarrow Xd \mid e \end{aligned}$$

¿cómo eliminamos la recursión por la izquierda **no-inmediata**?

## Estrategia

Dado  $V = \{X_1, \dots, X_n\}$ , removemos la recursión inductivamente en  $n$  tal que, en cada paso  $i$  de la inducción, se cumplirá que para todo  $i, j \leq n$ :

si  $X_i \rightarrow X_j \alpha$ , entonces  $i < j$ .

# Recursión por la izquierda no-inmediata

**input** : Gramática  $\mathcal{G} = (V, \Sigma, P, S)$  y  $V = \{X_1, \dots, X_n\}$

**output**: Gramática  $\mathcal{G}'$  sin recursión por la izquierda

**Function** EliminarRecursión ( $\mathcal{G}$ )

$P' := P$

**for**  $i = 1$  **to**  $n$  **do**

**for**  $j = 1$  **to**  $i - 1$  **do**

**foreach**  $X_i \rightarrow X_j \gamma \in P'$  **do**

**foreach**  $X_j \rightarrow \alpha \in P'$  **do**

$P' := P' \cup \{X_i \rightarrow \alpha \gamma\}$

$P' := P' - \{X_i \rightarrow X_j \gamma\}$

    Remove recursión inmediata para  $X_i$  en  $P'$  (si existe)

$V' := \{X_1, \dots, X_n\} \cup \{X'_1, \dots, X'_n\}$

**return** ( $V', \Sigma, P', S$ )

Ejercicio: demuestre la correctitud del algoritmo.

# Recursión por la izquierda no-inmediata

## Ejemplo

$$S \rightarrow Xa \mid b$$

$$X \rightarrow Yc$$

$$Y \rightarrow Xd \mid e$$

Ordenamos  $S, X, Y$  como  $X_1, X_2, X_3$ .

$$X_1 \rightarrow X_2a \mid b$$

$$X_2 \rightarrow X_3c$$

$$X_3 \rightarrow X_2d \mid e$$

■ Para  $i = 1$ : si  $X_1 \rightarrow X_j\alpha$ , entonces  $1 < j$ .



■ Para  $i = 2$ : si  $X_2 \rightarrow X_j\alpha$ , entonces  $2 < j$ .



Solo para  $i = 3$  debemos **eliminar los ciclos**.



# Recursión por la izquierda no-inmediata

Ejemplo: para caso  $i = 3$

$$X_1 \rightarrow X_2 a \mid b$$

$$X_2 \rightarrow X_3 c$$

$$X_3 \rightarrow X_2 d \mid e$$

---

$$X_1 \rightarrow X_2 a \mid b$$

$$X_2 \rightarrow X_3 c$$

$$X_3 \rightarrow X_3 c d \mid e$$

---

$$X_1 \rightarrow X_2 a \mid b$$

$$X_2 \rightarrow X_3 c$$

$$X_3 \rightarrow e X'_3$$

$$X'_3 \rightarrow c d X'_3 \mid \epsilon$$



# Toda gramática se puede convertir en GNF

Sea  $\mathcal{G} = (V, \Sigma, P, S)$  una CFG tal que  $\epsilon \notin \mathcal{L}(\mathcal{G})$ .

## Teorema

Existe una gramática  $\mathcal{G}'$  en forma normal de Greibach tal que:

$$\mathcal{L}(\mathcal{G}') = \mathcal{L}(\mathcal{G})$$

## Algoritmo

Dado una gramática  $\mathcal{G}$ .

1.  $\mathcal{G}' := \text{EliminarRecursión}(\mathcal{G})$
2.  $\mathcal{G}'' := \text{GNF-aciclico}(\mathcal{G}')$

**Resultado:** Gramática  $\mathcal{G}''$  en forma normal de Greibach ("casi").

OJO: El tamaño de  $\mathcal{G}''$  puede ser **exponencial** en el tamaño de  $\mathcal{G}$ .