

PROCESADORES DE LENGUAJES

TEMA III.- FUNDAMENTOS TEÓRICOS DEL ANÁLISIS SINTÁCTICO

Prof. Dr. Nicolás Luis Fernández García

Departamento de Informática y Análisis Numérico
Escuela Politécnica Superior
Universidad de Córdoba

Programa

- Tema I.- Introducción
- Tema II.- Análisis Lexicográfico
- Tema III.- Fundamentos Teóricos del Análisis Sintáctico
- Tema IV.- Análisis Sintáctico Descendente
- Tema V.- Análisis Sintáctico Ascendente

Programa

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad
- 4 Operaciones de limpieza
- 5 Recursividad y factorización
- 6 Formas normales
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos

Introducción

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad
- 4 Operaciones de limpieza
- 5 Recursividad y factorización
- 6 Formas normales
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos

Introducción

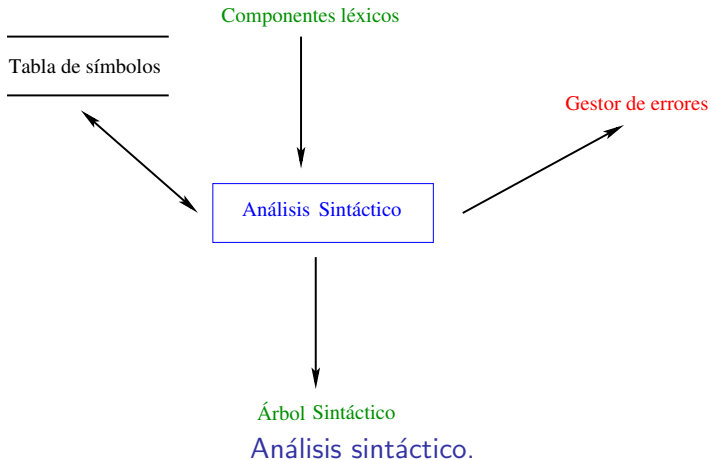
- 1 Introducción
 - El análisis sintáctico en el proceso de traducción

Contenido de la sección

- 1 Introducción
 - El análisis sintáctico en el proceso de traducción

Introducción

El análisis sintáctico en el proceso de traducción



Introducción

El análisis sintáctico en el proceso de traducción

Tareas del análisis sintáctico

- Recibir los componentes léxicos.
- Comprobar que se cumplen las reglas sintácticas de lenguaje de programación:
 - Utiliza una gramática de contexto libre.
 - Genera un árbol sintáctico de forma figurada.

Nota (Observaciones)

- *No tiene contacto directo con el programa fuente.*
- *Tiene acceso a la tabla de símbolos.*
- *Se comunica con el gestor de errores.*

Introducción

El análisis sintáctico en el proceso de traducción

Justificación del uso de las gramáticas de contexto libre

- 1 Permiten especificar sintácticamente las sentencias de los lenguajes de programación.
- 2 Existen herramientas de generadores de analizadores sintácticos a partir de una gramática: YACC, ANTLR, etc.
- 3 Facilitan la generación de código
- 4 Facilitan la detección y el procesamiento de los errores.
- 5 Permiten la ampliación del lenguaje.

Nota

Las gramáticas de contexto libres son potentes, pero también tienen sus limitaciones.

Introducción

El análisis sintáctico en el proceso de traducción

Ejemplo (Lenguaje que no es de contexto libre)

$$L = \{a^n b^n c^n \mid n \geq 1\} = \{abc, aabbcc, aaabbbccc, \dots\}$$

- *Este lenguaje no puede ser generado por una gramática de contexto libre.*
- *Se demuestra con el lema de bombeo de los lenguajes de contexto libre.*

Introducción

El análisis sintáctico en el proceso de traducción

Ejemplo (Coordinación de argumentos de una función)

```
int mcd(int a, int b);  
...  
main ()  
{  
    c=mcd (18, 12);  
}
```

```
int mcd (int a, int b)  
{  
    return ...;  
}
```

Introducción

El análisis sintáctico en el proceso de traducción

Ejemplo (Lenguajes que sí son de contexto libre)

- $L = \{a^n b^n \mid n \geq 1\} = \{ab, aabb, \dots\}$
- $L = \{a^i b^j c^k \mid i, j, k \geq 1\} = \{a, ab, aab, \dots\}$

Gramáticas de contexto libre

- 1 Introducción
- 2 Gramáticas de contexto libre**
- 3 Ambigüedad
- 4 Operaciones de limpieza
- 5 Recursividad y factorización
- 6 Formas normales
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos

Gramáticas de contexto libre

- 2 Gramáticas de contexto libre
 - Introducción
 - Definición
 - Convenios de notación
 - Derivación
 - Árbol sintáctico asociado a una derivación
 - Lenguaje generado por una gramática

Contenido de la sección

- 2 Gramáticas de contexto libre
 - **Introducción**
 - Definición
 - Convenios de notación
 - Derivación
 - Árbol sintáctico asociado a una derivación
 - Lenguaje generado por una gramática

Gramáticas de contexto libre

Introducción

Gramáticas

- Indican las reglas **sintácticas** de los lenguajes.
- Pueden **generar**
 - frases de lenguajes naturales
 - cadenas de lenguajes formales
 - Los lenguajes de programación son un caso particular de lenguajes formales.

Gramáticas de contexto libre

Introducción

Ejemplo (Gramática que genera frases copulativas) 1 / 6

(1) $\langle \textit{oración} \rangle \rightarrow \langle \textit{sujeto} \rangle \langle \textit{verbo} \rangle \langle \textit{atributo} \rangle$

(2) $\langle \textit{sujeto} \rangle \rightarrow \langle \textit{artículo} \rangle \langle \textit{nombre} \rangle$

(3) $\langle \textit{artículo} \rangle \rightarrow \mathbf{el}$

(4) $\langle \textit{artículo} \rangle \rightarrow \mathbf{la}$

(5) $\langle \textit{nombre} \rangle \rightarrow \mathbf{hombre}$

(6) $\langle \textit{nombre} \rangle \rightarrow \mathbf{niña}$

...

Gramáticas de contexto libre

Introducción

Ejemplo (Gramática que genera frases copulativas 2 / 6)

...

(7) <verbo> → **es**

(8) <verbo> → **está**

(9) <verbo> → **parece**

(10) <atributo> → <adjetivo>

(11) <adjetivo> → **alto**

(12) <adjetivo> → **bella**

(13) <adjetivo> → **inteligente**

Gramáticas de contexto libre

Introducción

Ejemplo (Agrupamiento de reglas)

3 / 6

<artículo> → **el** | **la** | **un** | **una**

<nombre> → **hombre** | **niña**

<verbo> → **es** | **está** | **parece**

<adjetivo> → **alto** | **bella** | **inteligente**

Gramáticas de contexto libre

Introducción

Ejemplo (Generación de una frase mediante derivación 4 / 6)

$\langle oración \rangle \xRightarrow{1} \underline{\langle sujeto \rangle} \langle verbo \rangle \langle atributo \rangle$

$\xRightarrow{2} \underline{\langle artículo \rangle} \langle nombre \rangle \langle verbo \rangle \langle atributo \rangle$

$\xRightarrow{4} \underline{la} \langle nombre \rangle \langle verbo \rangle \langle atributo \rangle$

$\xRightarrow{6} \underline{la} \underline{niña} \langle verbo \rangle \langle atributo \rangle$

$\xRightarrow{7} \underline{la} \underline{niña} \underline{es} \langle atributo \rangle$

$\xRightarrow{10} \underline{la} \underline{niña} \underline{es} \underline{\langle adjetivo \rangle}$

$\xRightarrow{13} \underline{la} \underline{niña} \underline{es} \underline{inteligente}$

Gramáticas de contexto libre

Introducción

Ejemplo (Generación de una frase mediante derivación 5 / 6)

- *Notación abreviada*

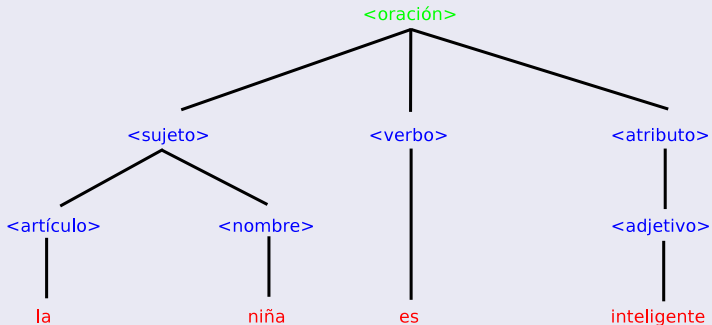
$\langle \text{oración} \rangle \xRightarrow{+} \text{la niña es inteligente}$

Gramáticas de contexto libre

Introducción

Ejemplo (Árbol sintáctico asociado a la derivación

6 / 6)



Gramáticas de contexto libre

Introducción

Nota (Limitación de las gramáticas de contexto libre 1 / 2)

- Error semántico**

$\langle \text{oración} \rangle \xRightarrow{1} \underline{\langle \text{sujeto} \rangle} \langle \text{verbo} \rangle \langle \text{atributo} \rangle$
 $\xRightarrow{2} \underline{\langle \text{artículo} \rangle} \underline{\langle \text{nombre} \rangle} \langle \text{verbo} \rangle \langle \text{atributo} \rangle$
 $\xRightarrow{4} \underline{\text{la}} \langle \text{nombre} \rangle \langle \text{verbo} \rangle \langle \text{atributo} \rangle$
 $\xRightarrow{5} \underline{\text{la}} \underline{\text{hombre}} \langle \text{verbo} \rangle \langle \text{atributo} \rangle$
 $\xRightarrow{9} \underline{\text{la}} \underline{\text{hombre}} \underline{\text{parece}} \langle \text{atributo} \rangle$
 $\xRightarrow{10} \underline{\text{la}} \underline{\text{hombre}} \underline{\text{parece}} \underline{\langle \text{adjetivo} \rangle}$
 $\xRightarrow{12} \underline{\text{la}} \underline{\text{hombre}} \underline{\text{parece}} \underline{\text{bella}}$

Gramáticas de contexto libre

Introducción

Nota (Limitación de las gramáticas de contexto libre 2 / 2)

- *La derivación*

$\langle \text{oración} \rangle \xRightarrow{+}$ **la hombre parece bella**

es *sintácticamente* correcta

pero *no* es *semánticamente* correcta

Gramáticas de contexto libre

Introducción

Ejemplo (Modelo limitado de la gramática

1 / 2)

- *Reglas de producción de la gramática que genera frases copulativas*
 - (1) <oración> \rightarrow <sujeito> **verbo** <atributo>
 - (2) <sujeito> \rightarrow **artículo nombre**
 - (3) <atributo> \rightarrow **adjetivo**
- *Componentes léxicos reconocidos por el analizador léxico*
 - **artículo:** *el, la, los, las, un, una.*
 - **nombre:** *hombre, mujer, niño, niña,*
 - **verbo:** *es, está, parece, son, estan, parecen,*
 - **adjetivo:** *alto, alta, bella, bello, inteligente, ...*

Gramáticas de contexto libre

Introducción

Ejemplo (Modelo limitado de la gramática)

2 / 2

$\langle \text{oración} \rangle \xRightarrow{1} \underline{\langle \text{sujeto} \rangle \text{ verbo } \langle \text{atributo} \rangle}$

$\xRightarrow{2} \underline{\text{artículo nombre}} \text{ verbo } \langle \text{atributo} \rangle$

$\xRightarrow{3} \text{artículo nombre verbo } \underline{\text{adjetivo}}$

- Esta derivación es *sintácticamente correcta* para las siguientes frases copulativas
 - *Semánticamente correctas*
 - la niña es inteligente
 - la mujer es alta
 - *Semánticamente incorrectas*
 - la hombre parece bella
 - los mujer son inteligente

Gramáticas de contexto libre

Introducción

Ejemplo (Expresiones aritméticas: gramática

1 / 3)

 $P = \{$

(1) $\langle \text{asignación} \rangle \rightarrow \mathbf{identificador} = \langle \text{expresión} \rangle$

(2) $\langle \text{expresión} \rangle \rightarrow \langle \text{expresión} \rangle + \langle \text{sumando} \rangle$

(3) $\langle \text{expresión} \rangle \rightarrow \langle \text{sumando} \rangle$

(4) $\langle \text{sumando} \rangle \rightarrow \langle \text{sumando} \rangle * \langle \text{factor} \rangle$

(5) $\langle \text{sumando} \rangle \rightarrow \langle \text{factor} \rangle$

(6) $\langle \text{factor} \rangle \rightarrow \mathbf{número}$

(7) $\langle \text{factor} \rangle \rightarrow \mathbf{identificador}$

(8) $\langle \text{factor} \rangle \rightarrow (\langle \text{expresión} \rangle)$

 $\}$

Gramáticas de contexto libre

Introducción

Ejemplo (Expresiones aritméticas: derivación)

3 / 3)

$$\langle \text{asignación} \rangle \xRightarrow{1} \underline{\text{identificador}} = \underline{\langle \text{expresión} \rangle}$$

$$\xRightarrow{2} \text{identificador} = \underline{\langle \text{expresión} \rangle} + \underline{\langle \text{sumando} \rangle}$$

$$\xRightarrow{3} \text{identificador} = \underline{\langle \text{sumando} \rangle} + \langle \text{sumando} \rangle$$

$$\xRightarrow{4} \text{identificador} = \underline{\langle \text{sumando} \rangle} * \underline{\langle \text{factor} \rangle} + \langle \text{sumando} \rangle$$

$$\xRightarrow{5} \text{identificador} = \underline{\langle \text{factor} \rangle} * \langle \text{factor} \rangle + \langle \text{sumando} \rangle$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{número}} * \underline{\langle \text{factor} \rangle} + \langle \text{sumando} \rangle$$

$$\xRightarrow{7} \text{identificador} = \text{número} * \underline{\text{identificador}} + \langle \text{sumando} \rangle$$

$$\xRightarrow{5} \text{identificador} = \text{número} * \text{identificador} + \underline{\langle \text{factor} \rangle}$$

$$\xRightarrow{6} \text{identificador} = \text{número} * \text{identificador} + \underline{\text{identificador}}$$

Gramáticas de contexto libre

Introducción

Ejemplo (Expresiones aritméticas: derivación abreviada 3 / 3)

$\langle \text{asignación} \rangle \xRightarrow{+} \text{identificador} = \text{número} * \text{identificador} + \text{identificador}$

Contenido de la sección

- 2 Gramáticas de contexto libre
 - Introducción
 - **Definición**
 - Convenios de notación
 - Derivación
 - Árbol sintáctico asociado a una derivación
 - Lenguaje generado por una gramática

Gramáticas de contexto libre

Definición

Definición (Gramática de Contexto Libre)

- $G = (V_N, V_T, P, S)$
 - V_N : alfabeto o vocabulario no terminal
 - V_T : alfabeto o vocabulario terminal
 - Se verifica que
$$V_N \cap V_T = \emptyset$$
 - Vocabulario de la gramática
$$V = V_N \cup V_T$$
 - Conjunto de *reglas de producción*
$$P = \{A \rightarrow \alpha \mid A \in V_N \wedge \alpha \in V^* = (V_N \cup V_T)^*\}$$
 - Símbolo inicial
$$S \in V_N$$

Gramáticas de contexto libre

Definición

Notas (Gramática de Contexto Libre)

- V_N también se puede denotar por Σ_N o N
- V_T también se puede denotar por Σ_T o T
- V también se puede denotar por Σ
- El símbolo inicial S también se denomina **axioma** o **símbolo distinguido**
- Si $A \rightarrow \alpha \in P$ entonces se dice que
 - A : símbolo no terminal de la parte izquierda de la regla.
 - α : parte derecha o alternativa de la regla.

Gramáticas de contexto libre

Definición

Ejemplo (Expresiones aritméticas: gramática

1 / 3)

- $V_N = \{S, E\}$
- $V_T = \{\text{identificador}, =, +, *, (,), \text{número}\}$
- $P = \{$
 - (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + E$
 - (3) $E \rightarrow E * E$
 - (4) $E \rightarrow (E)$
 - (5) $E \rightarrow \text{número}$
 - (6) $E \rightarrow \text{identificador}$ $\}$

Gramáticas de contexto libre

Definición

Ejemplo (Expresiones aritméticas: derivación)

2 / 3

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + E$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + \underline{E * E}$$

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{\text{número}} * E$$

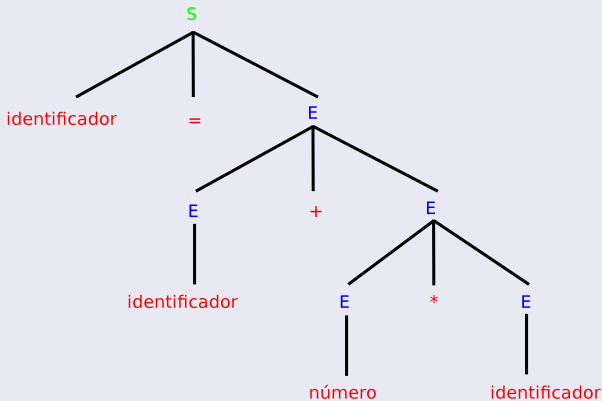
$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Gramáticas de contexto libre

Definición

Ejemplo (Árbol sintáctico asociado a la derivación)

3 / 3



Gramáticas de contexto libre

Definición

Ejemplo (Palíndromo impar: gramática

1 / 3)

$$P = \{$$

- (1) $S \rightarrow \mathbf{a A a}$
- (2) $A \rightarrow \mathbf{a A a}$
- (3) $A \rightarrow \mathbf{b B b}$
- (4) $B \rightarrow \mathbf{b B b}$
- (5) $B \rightarrow \mathbf{c}$

$$\}$$

Gramáticas de contexto libre

Definición

Nota (Palíndromo impar: características)

2 / 3

- Se denomina *palíndromo impar* porque cada palabra se puede **leer igual** de izquierda a derecha que de derecha a izquierda y tiene un elemento central que divide a la palabra.

$$\begin{aligned}L(G) &= \{ a^i b^j \mathbf{c} b^j a^i \mid i, j \geq 1 \} \\ &= \{ \mathbf{a b c b a}, \mathbf{a b b c b b a}, \dots \}\end{aligned}$$

Gramáticas de contexto libre

Definición

Ejemplo (Palíndromo impar: derivación)

3 / 3

$$\begin{aligned} S &\xRightarrow{1} \mathbf{a \underline{A} a} \\ &\xRightarrow{2} \mathbf{a a \underline{A} a a} \\ &\xRightarrow{3} \mathbf{a a \underline{b B b} a a} \\ &\xRightarrow{4} \mathbf{a a b \underline{b B b} b a a} \\ &\xRightarrow{5} \mathbf{a a b b \underline{c} b b a a} \end{aligned}$$

Contenido de la sección

- 2 Gramáticas de contexto libre
 - Introducción
 - Definición
 - **Convenios de notación**
 - Derivación
 - Árbol sintáctico asociado a una derivación
 - Lenguaje generado por una gramática

Gramáticas de contexto libre

Convenios de notación

Convenios de notación

1 / 7

(1) Símbolos terminales: V_T

- **Primeras letras minúsculas** del alfabeto latino:
a, b, c, ...
- **Operadores** aritméticos, lógicos, relacionales:
+, -, *, /, &&, ||, <, >, =, ...
- **Números**:
0, 1, ..., 9, 1.7, -83,01, 7 i, 2 + 3 i,
- **Palabras reservadas**:
if, else, for, ...
- **Signos de puntuación**:
., ;, :, {, }, [,], (,), ...

Gramáticas de contexto libre

Convenios de notación

Convenios de notación

2 / 7

(2) Símbolos no terminales: V_N

- **Primeras** letras **mayúsculas** del alfabeto latino:
A, B, C, ... y la letra S.
- Palabras delimitadas por < y >:
<oración>, <expresión>, ...

Gramáticas de contexto libre

Convenios de notación

Convenios de notación

3 / 7

(3) **Símbolos gramaticales:** $V = V_N \cup V_T$

- **Últimas** letras **mayúsculas** del alfabeto latino:
... X, Y, Z

Ejemplos

$$X = \begin{cases} a \in V_T \\ \text{if} \in V_T \\ A \in V_N \\ \langle \text{sumando} \rangle \in V_N \end{cases}$$

Gramáticas de contexto libre

Convenios de notación

Convenios de notación

4 / 7

(4) **Cadenas de símbolos gramaticales:** $V^* = (V_N \cup V_T)^*$

- **Primeras letras minúsculas** del alfabeto griego:

α, β, \dots

Ejemplos

$$\alpha = \begin{cases} \epsilon \\ a \\ B \\ aBB \end{cases}$$

identificador = identificador + < sumando >

Gramáticas de contexto libre

Convenios de notación

Convenios de notación

5 / 7

(5) Cadenas de símbolos terminales: V_T^*

- **Últimas** letras **minúsculas** del alfabeto latino:
... , x, y, z

Ejemplos

$$x = \begin{cases} \epsilon \\ a b b c b b a \\ \text{identificador} = \text{número} * \text{identificador} + \text{identificador} \end{cases}$$

Gramáticas de contexto libre

Convenios de notación

Convenios de notación

6 / 7

(6) **Palabra vacía:** ϵ o λ

Gramáticas de contexto libre

Convenios de notación

Convenios de notación

7 / 7

(7) Agrupamiento de reglas:

- Las reglas

$$(1) A \rightarrow \alpha_1$$

$$(2) A \rightarrow \alpha_2$$

...

$$(n) A \rightarrow \alpha_n$$

- se agrupan de la siguiente forma:

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

Gramáticas de contexto libre

Convenios de notación

Ejemplo (Notación)

$$P = \left\{ \begin{array}{l} S \longrightarrow a A B \\ A \longrightarrow a A b \mid c B d \\ B \longrightarrow c B d \mid c d \end{array} \right\}$$

$$V_N = \{S, A, B\}$$
$$V_T = \{a, b, c, d\}$$

Contenido de la sección

- 2 Gramáticas de contexto libre
 - Introducción
 - Definición
 - Convenios de notación
 - Derivación
 - Árbol sintáctico asociado a una derivación
 - Lenguaje generado por una gramática

Gramáticas de contexto libre

Derivación

Derivación

- Derivación inmediata
- Derivación general
- Derivación por la izquierda
- Derivación por la derecha

Gramáticas de contexto libre

Derivación

Derivación

- Derivación inmediata
- Derivación general
- Derivación por la izquierda
- Derivación por la derecha

Gramáticas de contexto libre

Derivación

Definición (Derivación inmediata)

- Sea $G = (V_N, V_T, P, S)$ una gramática de contexto libre

Si $\alpha = \delta A \gamma \in V^+$

y $A \rightarrow \beta \in P$

entonces se obtiene la siguiente **derivación inmediata**

$$\alpha = \delta A \gamma \xRightarrow{(A \rightarrow \beta)} \delta \beta \gamma = \alpha'$$

Gramáticas de contexto libre

Derivación

Derivación

- Derivación inmediata
- Derivación general
- Derivación por la izquierda
- Derivación por la derecha

Gramáticas de contexto libre

Derivación

Definición (Derivación general)

- Sea $G = (V_N, V_T, P, S)$ una gramática de contexto libre
- Una *derivación general* es una secuencia de cadenas

$$\alpha_0, \alpha_1, \alpha_2, \dots, \alpha_n \in V^*$$

donde $\forall i \in \{0, 1, \dots, n-1\}$

α_i deriva de forma inmediata a α_{i+1}

$$\alpha_0 \Rightarrow \alpha_1$$

$$\alpha_1 \Rightarrow \alpha_2$$

...

$$\alpha_{n-1} \Rightarrow \alpha_n$$

Forma equivalente

$$\alpha_0 \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \dots \Rightarrow \alpha_n$$

Gramáticas de contexto libre

Derivación

Nota (Derivación general)

- *La derivación general*

$$\alpha_0 \Longrightarrow \alpha_1 \Longrightarrow \alpha_2 \Longrightarrow \cdots \Longrightarrow \alpha_n$$

se puede denotar como

- *derivación en n pasos:* $\alpha_0 \xRightarrow{n} \alpha_n$
- *derivación en cero o más pasos:* $\alpha_0 \xRightarrow{*} \alpha_n$
- *derivación en uno o más pasos:* $\alpha_0 \xRightarrow{+} \alpha_n$
- $\forall \alpha \in V^*$ se verifica que

$$\alpha \xRightarrow{0} \alpha$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación: gramática)

1 / 8

- $V_N = \{S, E\}$
- $V_T = \{\text{identificador}, =, +, *, (,), \text{número}\}$
- $P = \{$
 - (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + E$
 - (3) $E \rightarrow E * E$
 - (4) $E \rightarrow (E)$
 - (5) $E \rightarrow \text{número}$
 - (6) $E \rightarrow \text{identificador}$ $\}$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación)

2 / 8

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{3} \text{identificador} = \underline{E * E}$$

$$\xRightarrow{6} \text{identificador} = E * \underline{\text{identificador}}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E} * \text{identificador}$$

$$\xRightarrow{5} \text{identificador} = E + \underline{\text{número}} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación)

3 / 8

$$\underbrace{\epsilon}_{\delta} \underbrace{S}_{A} \underbrace{\epsilon}_{\gamma} \xRightarrow{1} \underbrace{\epsilon}_{\delta} \underbrace{\text{identificador} = E}_{\beta} \underbrace{\epsilon}_{\gamma}$$

$$\xRightarrow{3} \text{identificador} = \underline{E * E}$$

$$\xRightarrow{6} \text{identificador} = E * \underline{\text{identificador}}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E} * \text{identificador}$$

$$\xRightarrow{5} \text{identificador} = E + \underline{\text{número}} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación)

4 / 8

$$S \xRightarrow{1} \underbrace{\text{identificador}}_{\delta} = \underbrace{E}_A \underbrace{\epsilon}_{\gamma}$$

$$\xRightarrow{3} \underbrace{\text{identificador}}_{\delta} = \underbrace{E * E}_{\beta} \underbrace{\epsilon}_{\gamma}$$

$$\xRightarrow{6} \text{identificador} = E * \underline{\text{identificador}}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E} * \text{identificador}$$

$$\xRightarrow{5} \text{identificador} = E + \underline{\text{número}} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación)

5 / 8

$$S \xRightarrow{1} \text{identificador} = E$$

$$\xRightarrow{3} \underbrace{\text{identificador} = E^*}_{\delta} \underbrace{E}_A \underbrace{\epsilon}_{\gamma}$$

$$\xRightarrow{6} \underbrace{\text{identificador} = E^*}_{\delta} \underbrace{\text{identificador}}_{\beta} \underbrace{\epsilon}_{\gamma}$$

$$\xRightarrow{2} \text{identificador} = E + E^* \text{identificador}$$

$$\xRightarrow{5} \text{identificador} = E + \text{número} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \text{identificador}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación)

6 / 8

$$S \xRightarrow{1} \text{identificador} = E$$

$$\xRightarrow{3} \text{identificador} = E * E$$

$$\xRightarrow{6} \underbrace{\text{identificador}}_{\delta} = \underbrace{E}_A * \underbrace{\text{identificador}}_{\gamma}$$

$$\xRightarrow{2} \underbrace{\text{identificador}}_{\delta} = \underbrace{E+E}_{\beta} * \underbrace{\text{identificador}}_{\gamma}$$

$$\xRightarrow{5} \text{identificador} = E + \text{número} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \text{identificador}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación)

7 / 8

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{3} \text{identificador} = E * E$$

$$\xRightarrow{6} \text{identificador} = E * \underline{\text{identificador}}$$

$$\xRightarrow{2} \underbrace{\text{identificador} = E}_{\delta} + \underbrace{E}_A * \underbrace{\text{identificador}}_{\gamma}$$

$$\xRightarrow{5} \underbrace{\text{identificador} = E}_{\delta} + \underbrace{\text{número}}_{\beta} * \underbrace{\text{identificador}}_{\gamma}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación)

8 / 8)

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{3} \text{identificador} = \underline{E * E}$$

$$\xRightarrow{6} \text{identificador} = E * \underline{\text{identificador}}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E} * \text{identificador}$$

$$\xRightarrow{5} \underbrace{\text{identificador}}_{\delta} = \underbrace{E}_A + \underbrace{\text{número} * \text{identificador}}_{\gamma}$$

$$\xRightarrow{6} \underbrace{\text{identificador}}_{\delta} = \underbrace{\text{identificador}}_{\beta} + \underbrace{\text{número} * \text{identificador}}_{\gamma}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Gramática

1 / 2

$$P = \{$$

- (1) $S \rightarrow \mathbf{a A a}$
- (2) $A \rightarrow \mathbf{a A a}$
- (3) $A \rightarrow \mathbf{b B b}$
- (4) $B \rightarrow \mathbf{b B b}$
- (5) $B \rightarrow \mathbf{c}$

$$\}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación)

2 / 2)

$$\begin{aligned} S &\xRightarrow{1} \mathbf{a} \underline{A} \mathbf{a} \\ &\xRightarrow{2} \mathbf{a} \mathbf{a} \underline{A} \mathbf{a} \mathbf{a} \\ &\xRightarrow{3} \mathbf{a} \mathbf{a} \mathbf{b} \underline{B} \mathbf{b} \mathbf{a} \mathbf{a} \\ &\xRightarrow{4} \mathbf{a} \mathbf{a} \mathbf{b} \mathbf{b} \underline{B} \mathbf{b} \mathbf{b} \mathbf{a} \mathbf{a} \\ &\xRightarrow{5} \mathbf{a} \mathbf{a} \mathbf{b} \mathbf{b} \underline{c} \mathbf{b} \mathbf{b} \mathbf{a} \mathbf{a} \in V_T^* \end{aligned}$$

Gramáticas de contexto libre

Derivación

Derivación

- Derivación inmediata
- Derivación general
- Derivación por la izquierda
- Derivación por la derecha

Gramáticas de contexto libre

Derivación

Definición (Derivación inmediata por la izquierda)

- $G = (V_N, V_T, P, S)$

Si $A \rightarrow \beta \in P$ y $\alpha = x A \gamma$

entonces la derivación inmediata por la izquierda se define como:

$$\alpha = x A \gamma \xRightarrow{(A \rightarrow \beta)} x \underline{\beta} \gamma = \alpha'$$

donde

- $x \in V_T^*$
- $A \in V_N$
- $\beta, \gamma \in V^*$

Gramáticas de contexto libre

Derivación

Nota (Derivación inmediata por la izquierda)

*Siempre se procesa el símbolo **no terminal** situado más a la izquierda.*

Gramáticas de contexto libre

Derivación

Definición (Derivación por la izquierda)

- Una derivación es por la izquierda si todas sus derivaciones inmediatas son por la izquierda.

$$\alpha_0 \xRightarrow{I} \alpha_1$$

$$\alpha_1 \xRightarrow{I} \alpha_2$$

...

$$\alpha_{n-1} \xRightarrow{I} \alpha_n$$

que es equivalente a

$$\alpha_0 \xRightarrow{I} \alpha_1 \xRightarrow{I} \alpha_2 \xRightarrow{I} \dots \xRightarrow{I} \alpha_n$$

$$\alpha_0 \xRightarrow{I}^* \alpha_n$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la izquierda: gramática 1 / 8)

- $V_N = \{S, E\}$
- $V_T = \{\text{identificador}, =, +, *, (,), \text{número}\}$
- $P = \{$
 - (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + E$
 - (3) $E \rightarrow E * E$
 - (4) $E \rightarrow (E)$
 - (5) $E \rightarrow \text{número}$
 - (6) $E \rightarrow \text{identificador}$ $\}$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la izquierda)

2 / 8

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + E$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + \underline{E * E}$$

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{\text{número}} * E$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la izquierda

3 / 8)

$$\underbrace{\epsilon}_x \underbrace{S}_A \underbrace{\epsilon}_\gamma \xRightarrow{1} \underbrace{\epsilon}_x \underbrace{\text{identificador} = E}_\beta \underbrace{\epsilon}_\gamma$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + E$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + \underline{E * E}$$

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{\text{número}} * E$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la izquierda)

4 / 8

$$S \xRightarrow{1} \underbrace{\text{identificador}}_x = \underbrace{E}_A \underbrace{\epsilon}_\gamma$$

$$\xRightarrow{2} \underbrace{\text{identificador}}_x = \underbrace{E+E}_\beta \underbrace{\epsilon}_\gamma$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + E$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + \underline{E * E}$$

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{\text{número}} * E$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la izquierda)

5 / 8

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \underline{\text{identificador}} = \underbrace{E}_x + \underbrace{E}_\gamma$$

$$\xRightarrow{6} \underline{\text{identificador}} = \underbrace{\text{identificador}}_x + \underbrace{E}_\gamma$$

$$\xRightarrow{3} \underline{\text{identificador}} = \text{identificador} + \underline{E * E}$$

$$\xRightarrow{5} \underline{\text{identificador}} = \text{identificador} + \underline{\text{número}} * E$$

$$\xRightarrow{6} \underline{\text{identificador}} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la izquierda)

6 / 8)

$$S \xRightarrow{1} \text{identificador} = E$$

$$\xRightarrow{2} \text{identificador} = E + E$$

$$\xRightarrow{6} \underbrace{\text{identificador} = \text{identificador}}_x + \underbrace{E}_A \underbrace{\epsilon}_\gamma$$

$$\xRightarrow{6} \underbrace{\text{identificador} = \text{identificador}}_x + \underbrace{E * E}_\beta \underbrace{\epsilon}_\gamma$$

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \text{número} * E$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \text{identificador}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la izquierda)

7 / 8)

$$S \xRightarrow{1} \underline{\text{identificador}} = \underline{E}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + E$$

$$\xRightarrow{6} \text{identificador} = \underbrace{\text{identificador}}_x + \underbrace{E}_A \underbrace{*E}_\gamma$$

$$\xRightarrow{5} \text{identificador} = \underbrace{\text{identificador}}_x + \underbrace{\text{número}}_\beta \underbrace{*E}_\gamma$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la izquierda)

8 / 8

$$S \xRightarrow{1} \underline{\text{identificador}} = \underline{E}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + E$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + \underline{E * E}$$

$$\xRightarrow{5} \underbrace{\text{identificador} = \text{identificador} + \text{número}}_x * \underbrace{E}_A \underbrace{\epsilon}_\gamma$$

$$\xRightarrow{6} \underbrace{\text{identificador} = \text{identificador} + \text{número}}_x * \underbrace{\text{identificador}}_\beta \underbrace{\epsilon}_\gamma$$

Gramáticas de contexto libre

Derivación

Nota

- *Lo deseable es que la gramática sólo tenga **una derivación por la izquierda** para cada cadena de símbolos terminales.*
- ***En caso contrario**, la gramática sería **ambigua**.*

Gramáticas de contexto libre

Derivación

Derivación

- Derivación inmediata
- Derivación general
- Derivación por la izquierda
- Derivación por la derecha

Gramáticas de contexto libre

Derivación

Definición (Derivación inmediata por la derecha)

- $G = (V_N, V_T, P, S)$

Si $A \rightarrow \beta \in P$ y $\alpha = x A \gamma$

entonces la derivación inmediata por la *derecha* se define como:

$$\alpha = \delta A y \xRightarrow{(A \rightarrow \beta)} \delta \underline{\beta} y = \alpha'$$

donde

- $y \in V_T^*$
- $A \in V_N$
- $\beta, \delta \in V^*$

Gramáticas de contexto libre

Derivación

Nota (Derivación inmediata por la derecha)

*Siempre se procesa el símbolo **no terminal** situado más a la derecha.*

Gramáticas de contexto libre

Derivación

Definición (Derivación por la izquierda)

- Una derivación es por la derecha si todas sus derivaciones inmediatas son por la derecha.

$$\alpha_0 \xRightarrow{D} \alpha_1$$

$$\alpha_1 \xRightarrow{D} \alpha_2$$

...

$$\alpha_{n-1} \xRightarrow{D} \alpha_n$$

que es equivalente a

$$\alpha_0 \xRightarrow{D} \alpha_1 \xRightarrow{D} \alpha_2 \xRightarrow{D} \dots \xRightarrow{D} \alpha_n$$

$$\alpha_0 \xRightarrow[D]{*} \alpha_n$$

Gramáticas de contexto libre

Derivación

Ejemplo (Gramática

1 / 2)

- $V_N = \{S, E\}$
- $V_T = \{\text{identificador}, =, +, *, (,), \text{número}\}$
- $P = \{$
 - (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + E$
 - (3) $E \rightarrow E * E$
 - (4) $E \rightarrow (E)$
 - (5) $E \rightarrow \text{número}$
 - (6) $E \rightarrow \text{identificador}$ $\}$

Gramáticas de contexto libre

Derivación

Ejemplo (Derivación por la derecha)

2 / 2

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{3} \text{identificador} = E + \underline{E * E}$$

$$\xRightarrow{6} \text{identificador} = E + E * \underline{\text{identificador}}$$

$$\xRightarrow{5} \text{identificador} = E + \underline{\text{número}} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Gramáticas de contexto libre

Derivación

Nota

- *Lo deseable es que la gramática sólo tenga **una derivación por la derecha** para cada cadena de símbolos terminales.*
- ***En caso contrario**, la gramática sería **ambigua**.*

Contenido de la sección

- 2 Gramáticas de contexto libre
 - Introducción
 - Definición
 - Convenios de notación
 - Derivación
 - **Árbol sintáctico asociado a una derivación**
 - Lenguaje generado por una gramática

Gramáticas de contexto libre

Árbol sintáctico asociado a una derivación

Árbol sintáctico asociado a una derivación

- (1) Los **nodos** del árbol están etiquetados con **símbolos** del vocabulario de la gramática (V) o la **palabra vacía** ϵ
- (2) La **raíz** está etiquetada con el símbolo inicial: $S \in V_N$
- (3) Si un nodo tiene, al menos un **descendiente**, entonces le corresponde un símbolo **no** terminal: $A \in V_N$
- (4) Si un nodo está etiquetado con un símbolo A y se ha aplicado la regla $A \rightarrow X_1 X_2 \dots X_n \in P$ entonces A tiene n **descendientes**:

$$X_1, X_2 \dots X_n \in V^* = (V_N \cup V_T)^*$$

Gramáticas de contexto libre

Árbol sintáctico asociado a una derivación

Ejemplo (Gramática

1 / 2)

$$P = \{$$

- (1) $S \rightarrow a A b$
- (2) $A \rightarrow a A b$
- (3) $A \rightarrow c B d$
- (4) $B \rightarrow c B d$
- (5) $B \rightarrow c d$

$$\}$$

Gramáticas de contexto libre

Árbol sintáctico asociado a una derivación

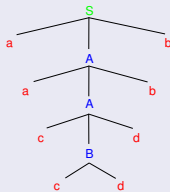
Ejemplo (Árbol de derivación)

2 / 2

$$S \xRightarrow{1} \mathbf{a A b}$$

$$\xRightarrow{2} \mathbf{a a A b b}$$

$$\xRightarrow{3} \mathbf{a a c B d b b}$$

$$\xRightarrow{5} \mathbf{a a c c d d b b}$$


Contenido de la sección

- 2 Gramáticas de contexto libre
 - Introducción
 - Definición
 - Convenios de notación
 - Derivación
 - Árbol sintáctico asociado a una derivación
 - Lenguaje generado por una gramática

Gramáticas de contexto libre

Lenguaje generado por una gramática

Definición (Lenguaje generado por una gramática)

- Si $G = (V_N, V_T, P, S)$ es una gramática de contexto libre entonces el lenguaje que genera se define como

$$L(G) = \{x \mid x \in V_T^* \wedge S \xrightarrow{+}_G x\}$$

Gramáticas de contexto libre

Lenguaje generado por una gramática

Ejemplo (Lenguaje generado por una gramática)

$$P = \{$$
$$(1) S \rightarrow a A b$$
$$(2) A \rightarrow a A b$$
$$(3) A \rightarrow c B d$$
$$(4) B \rightarrow c B d$$
$$(5) B \rightarrow c d$$
$$\}$$
$$L(G) = \{a^i c^j d^j b^i \mid i \geq 1 \wedge j \geq 2\} = \{accddb, \dots\}$$

Gramáticas de contexto libre

Lenguaje generado por una gramática

Notas (Lenguaje generado por una gramática)

- *No existe ningún algoritmo general que permite **comprobar** cuál es el lenguaje generado por una gramática de contexto libre.*
- *No existe ningún algoritmo general que permita **diseñar** una gramática de contexto libre que genere un lenguaje preterdeterminado.*
- *Se debe tener en cuenta la **experiencia** y el **sentido común**.*

Gramáticas de contexto libre

Lenguaje generado por una gramática

Ejercicio (Diseño de gramáticas de contexto libre 1 / 5)

- *Diseña una gramática de contexto libre que permita generar algunas declaraciones variables del lenguaje C*

```
int a, b;
```

```
float x;
```

Gramáticas de contexto libre

Lenguaje generado por una gramática

Ejercicio (Diseño de gramáticas de contexto libre

2 / 5)

- *Declaraciones de punteros del lenguaje C*

```
int *a, **b, c;
```

```
float x, *y, **z;
```

Gramáticas de contexto libre

Lenguaje generado por una gramática

Ejercicio (Diseño de gramáticas de contexto libre

3 / 5)

- *Declaraciones de arrays del lenguaje C*

```
int a[5], b[10][2];  
float x[10], m[3][3];
```

Gramáticas de contexto libre

Lenguaje generado por una gramática

Ejercicio (Diseño de gramáticas de contexto libre

4 / 5)

- *Declaraciones de arrays de punteros del lenguaje C*

```
int *a[5], **b[10][2];  
float *p[10], *m[3][3];
```


Gramáticas de contexto libre

Lenguaje generado por una gramática

Ejercicio (Diseño de gramáticas de contexto libre 5 / 5)

- *Declaraciones de prototipos de funciones del lenguaje C*

```
int f();  
int g(int a);  
int *h(int a, int *b);
```

Ambigüedad

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad**
- 4 Operaciones de limpieza
- 5 Recursividad y factorización
- 6 Formas normales
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos

Ambigüedad

- 3 Ambigüedad
 - Gramática ambigua
 - Lenguaje intrínsecamente ambiguo

Contenido de la sección

- 3 Ambigüedad
 - Gramática ambigua
 - Lenguaje intrínsecamente ambiguo

Ambigüedad

Gramática ambigua

Definición (Ambigüedad)

Una gramática de contexto libre es *ambigua* si cumple alguna de las siguientes condiciones:

- 1.- Existe una cadena que posee *dos derivaciones por la izquierda* diferentes.
- 2.- Existe una cadena que posee *dos derivaciones por la derecha* diferentes.
- 3.- Existe una cadena que posee *dos árboles sintácticos* diferentes.

Ambigüedad

Gramática ambigua

Notas (Ambigüedad)

- *No* existe un algoritmo general para comprobar si una gramática es ambigua o no.
- Se deben hacer comprobaciones particulares.
- Hay gramáticas ambiguas que se pueden *transformar* en otras que no lo son.
- Se debe *evitar* el uso de gramáticas *ambiguas* porque *dificultan o impiden el análisis sintáctico*.

Ambigüedad

Gramática ambigua

Ejemplo (Gramática ambigua)

1 / 6

La gramática de las expresiones aritméticas es *ambigua*.

- $$P = \{$$
- (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + E$
 - (3) $E \rightarrow E * E$
 - (4) $E \rightarrow (E)$
 - (5) $E \rightarrow \text{número}$
 - (6) $E \rightarrow \text{identificador}$
- $$\}$$

Ambigüedad

Gramática ambigua

Ejemplo (Gramática ambigua)

2 / 6)

La asignación

identificador = identificador + número * identificador

*puede ser generada por **dos** derivaciones por la izquierda diferentes*

Ambigüedad

Gramática ambigua

Ejemplo (Gramática ambigua)

3 / 6

Primera derivación por la izquierda

$$S \xRightarrow[1]{} \underline{\text{identificador}} = E$$

$$\xRightarrow[2]{} \text{identificador} = \underline{E + E}$$

$$\xRightarrow[5]{} \text{identificador} = \underline{\text{identificador}} + E$$

$$\xRightarrow[3]{} \text{identificador} = \text{identificador} + \underline{E * E}$$

$$\xRightarrow[6]{} \text{identificador} = \text{identificador} + \underline{\text{número}} * E$$

$$\xRightarrow[5]{} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Ambigüedad

Gramática ambigua

Ejemplo (Gramática ambigua)

4 / 6

Segunda derivación por la izquierda

$$E \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{3} \text{identificador} = \underline{E * E}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E} * E$$

$$\xRightarrow{5} \text{identificador} = \underline{\text{identificador}} + E * E$$

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{\text{número}} * E$$

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Ambigüedad

Gramática ambigua

Nota (Gramática ambigua)

5 / 6

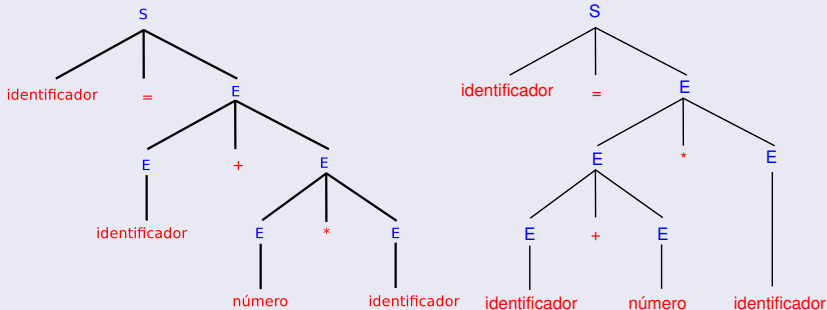
- La *primera* derivación es *correcta* porque tiene en cuenta la *prioridad* de los operadores aritméticos.
- El producto (*) tiene mayor prioridad que la suma (+).

Ambigüedad

Gramática ambigua

Ejemplo (Gramática ambigua: árboles sintácticos)

6 / 6



Ambigüedad

Gramática ambigua

Ejemplo (Gramática no ambigua)

1 / 3

$$P = \{$$

- (1) $S \rightarrow \text{identificador} = E$
- (2) $E \rightarrow E + T$
- (3) $E \rightarrow T$
- (4) $T \rightarrow T * F$
- (5) $T \rightarrow F$
- (6) $F \rightarrow (E)$
- (7) $F \rightarrow \text{identificador}$
- (8) $F \rightarrow \text{número}$

$$\}$$

Ambigüedad

Gramática ambigua

Ejemplo (Gramática no ambigua)

2 / 3

Derivación por la izquierda

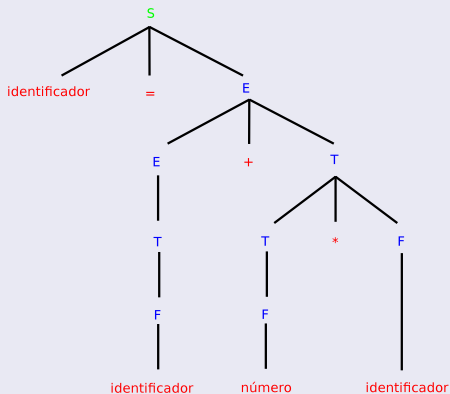
$$\begin{aligned}
 S &\xrightarrow{1} \underline{\text{identificador}} = \underline{E} \xrightarrow{2} \text{identificador} = \underline{E} + T \\
 &\xrightarrow{3} \text{identificador} = \underline{T} + T \xrightarrow{5} \text{identificador} = \underline{F} + T \\
 &\xrightarrow{7} \text{identificador} = \underline{\text{identificador}} + T \\
 &\xrightarrow{4} \text{identificador} = \text{identificador} + \underline{T} * F \\
 &\xrightarrow{5} \text{identificador} = \text{identificador} + \underline{F} * F \\
 &\xrightarrow{8} \text{identificador} = \text{identificador} + \underline{\text{número}} * F \\
 &\xrightarrow{7} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}
 \end{aligned}$$

Ambigüedad

Gramática ambigua

Ejemplo (Gramática no ambigua: árbol sintáctico)

3 / 3



Ambigüedad

Gramática ambigua

Ejemplo (El problema del 'else danzante')

1 / 8

 $P = \{$

...

(1) $S \rightarrow \text{if } C S$ (2) $S \rightarrow \text{if } C S \text{ else } S$ (3) $S \rightarrow I$

...

 $\}$ *donde*

- *S genera sentencias de control*
- *C genera expresiones condicionales*
- *I genera otras sentencias, por ejemplo, de asignación.*

Ambigüedad

Gramática ambigua

Ejemplo (El problema del 'else danzante')

2 / 8)

- *Esta gramática es ambigua porque la sentencia*

if C if C S else S

*puede ser generada por dos derivaciones que tienen asociados
árboles sintácticos diferentes.*

Ambigüedad

Gramática ambigua

Ejemplo (El problema del 'else danzante')

3 / 8)

- *Primera derivación*

$$S \xRightarrow{1} \text{if } C \ S \xRightarrow{2} \text{if } C \ \text{if } C \ S \ \text{else } S$$

- *Segunda derivación*

$$S \xRightarrow{2} \text{if } C \ S \ \text{else } S \xRightarrow{1} \text{if } C \ \text{if } C \ S \ \text{else } S$$

Ambigüedad

Gramática ambigua

Nota (El problema del 'else danzante')

4 / 8

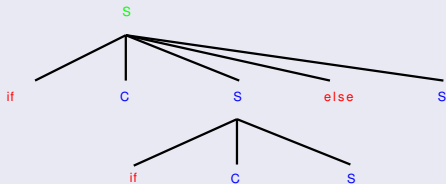
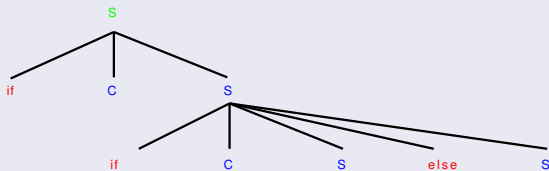
- La *primera* derivación es *correcta* porque asocia el **else** al **if** más cercano.

Ambigüedad

Gramática ambigua

Ejemplo (El problema del 'else danzante')

5 / 8



Ambigüedad

Gramática ambigua

Ejemplo (El problema del 'else danzante')

6 / 8

- *Solución: asociar cada **else** con el **if** más cercano*

...

$$(1) S \rightarrow S_1$$

$$(2) S \rightarrow S_2$$

$$(3) S_1 \rightarrow \text{if } C S_1 \text{ else } S_1$$

$$(4) S_1 \rightarrow I$$

$$(5) S_2 \rightarrow \text{if } C S$$

$$(6) S_2 \rightarrow \text{if } C S_1 \text{ else } S_2$$

...

donde

- S_1 genera la sentencia **if emparejada**
- S_2 genera la sentencia **if no emparejada.**

Ambigüedad

Gramática ambigua

Ejemplo (El problema del 'else danzante')

7 / 8

- Derivación

$$\begin{array}{l}
 S \xRightarrow{2} \underline{S_2} \\
 \xRightarrow{5} \text{if } C \underline{S} \\
 \xRightarrow{1} \text{if } C \underline{S_1} \\
 \xRightarrow{3} \text{if } C \text{ if } C \underline{S_1} \text{ else } S_2
 \end{array}$$

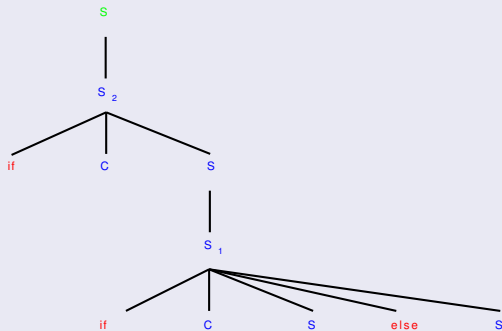
Ambigüedad

Gramática ambigua

Ejemplo (El problema del 'else danzante')

8 / 8

Árbol sintáctico que asocia **else** al **if** más cercano.



Contenido de la sección

- 3 Ambigüedad
 - Gramática ambigua
 - Lenguaje intrínsecamente ambiguo

Ambigüedad

Lenguaje intrínsecamente ambiguo

Definición (Lenguaje intrínsecamente ambiguo)

Un lenguaje es *intrínsecamente ambiguo* si todas las gramáticas que lo generan son ambiguas.

$$L = L(G_1) = L(G_2) = \dots = L(G_N)$$

donde G_1, G_2, \dots, G_N ambiguas

Ambigüedad

Lenguaje intrínsecamente ambiguo

Ejemplo (Lenguaje intrínsecamente ambiguo)

1 / 4

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

L solamente puede ser generado por gramáticas ambiguas.

Ambigüedad

Lenguaje intrínsecamente ambiguo

Ejemplo (Lenguaje intrínsecamente ambiguo)

2 / 4

- Una gramática que genera el lenguaje L

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

Ambigüedad

Lenguaje intrínsecamente ambiguo

Ejemplo (Lenguaje intrínsecamente ambiguo)

3 / 4

- La gramática G es ambigua

Primera derivación

$$\begin{aligned}
 S &\xRightarrow{1} \underline{A C} \\
 &\xRightarrow{4} \underline{a b} C \\
 &\xRightarrow{6} \underline{a b c}
 \end{aligned}$$

Segunda derivación

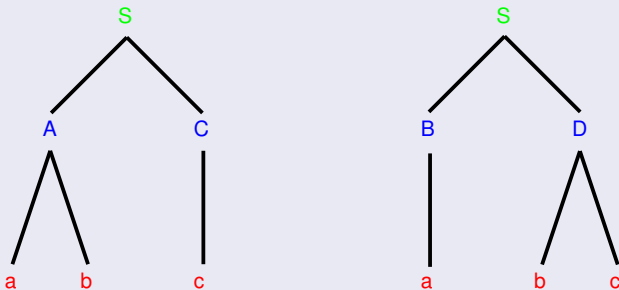
$$\begin{aligned}
 S &\xRightarrow{2} \underline{B D} \\
 &\xRightarrow{8} \underline{a} D \\
 &\xRightarrow{10} \underline{a b c}
 \end{aligned}$$

Ambigüedad

Lenguaje intrínsecamente ambiguo

Ejemplo (Lenguaje intrínsecamente ambiguo)

4 / 4



Árboles sintácticos diferentes

Operaciones de limpieza

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad
- 4 Operaciones de limpieza**
- 5 Recursividad y factorización
- 6 Formas normales
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos

Operaciones de limpieza

- 4 Operaciones de limpieza
 - Símbolos útiles e inútiles
 - Reglas superfluas
 - Gramática propia

Contenido de la sección

- 4 Operaciones de limpieza
 - Símbolos útiles e inútiles
 - Reglas superfluas
 - Gramática propia

Operaciones de limpieza

Símbolos útiles e inútiles

Definición (Símbolo útil)

Sea $X \in V = V_N \cup V_T$

Se dice que X es **útil** si es un símbolo *accesible* y *generador*.

es decir, el símbolo X aparece al menos en una derivación de una cadena perteneciente al lenguaje generado por la gramática

$$\exists S \xRightarrow{*} \alpha X \beta \xRightarrow{*} x_1 x_2 x_3 = x \in L(G)$$

Operaciones de limpieza

Símbolos útiles e inútiles

Símbolos útiles

- Símbolos generadores
- Símbolos accesibles

Operaciones de limpieza

Símbolos útiles e inútiles

Definición (Símbolo generador)

Sea $X \in V = V_N \cup V_T$

se dice que X es *generador* si

$$\exists X \xrightarrow[G]{*} x \in V_T^*$$

Notas

- Si $X = A \in V_N$ entonces A es *generador* si y solamente si $L(G_A) \neq \emptyset$
- Si $X = a \in V_T$ entonces a es *generador* porque $a \xrightarrow{0} a$

Operaciones de limpieza

Símbolos útiles e inútiles

Algoritmo (Selección de símbolos generadores

1 / 2)

- **Entrada**

- $G = (V_N, V_T, P, S)$
Gramática de contexto libre.

- **Salida**

- $G' = (V'_N, V_T, P', S)$
*Gramática de contexto libre **sin** símbolos **no generadores**.*

Operaciones de limpieza

Símbolos útiles e inútiles

Algoritmo (Selección de símbolos generadores

2 / 2)

inicio

$Viejo \leftarrow \emptyset$

$Nuevo \leftarrow \{A \mid A \in V_N \wedge \exists A \rightarrow x \in P \wedge x \in V_T^*\}$

mientras ($Nuevo \neq Viejo$) **hacer**

$Viejo \leftarrow Nuevo$

$Nuevo \leftarrow Viejo \cup \{A \mid A \in V_N \wedge \exists A \rightarrow \alpha \in P$
 $\wedge \alpha \in (Viejo \cup V_T)^*\}$

fin_mientras

$V'_N \leftarrow Nuevo$

$P' \leftarrow \{A \rightarrow \alpha \mid A \rightarrow \alpha \in P \wedge A \in V'_N \wedge \alpha \in (V'_N \cup V_T)^*\}$

fin

Operaciones de limpieza

Símbolos útiles e inútiles

Nota (Selección de símbolos generadores)

Las reglas de la gramática G' se obtienen a partir de las reglas de la gramática G que sólo tienen símbolos generadores.

$$P' \leftarrow \{A \rightarrow \alpha \mid A \rightarrow \alpha \in P \wedge A \in V'_N \wedge \alpha \in (V'_N \cup V_T)^*\}$$

Operaciones de limpieza

Símbolos útiles e inútiles

Nota (Selección de símbolos generadores)

- Una gramática de contexto libre genera un lenguaje *no vacío* si y solamente si su *símbolo inicial* es un *símbolo generador*.

$$L(G) \neq \emptyset \iff S \in \text{Generadores}$$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos generadores)

1 / 7

$$G = (V_N, V_T, P, S)$$

$$V_N = \{S, A, B, C, D, E, F\}$$

$$V_T = \{a, b, c, d, e\}$$

$$P = \{$$

$$(1) S \rightarrow A B$$

$$(2) S \rightarrow A b$$

$$(3) A \rightarrow a C$$

$$(4) B \rightarrow b C a$$

$$(5) B \rightarrow D b E$$

$$(6) C \rightarrow b$$

$$(7) D \rightarrow F b$$

$$(8) E \rightarrow c a e$$

$$(9) F \rightarrow a D d$$

$$\}$$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos generadores

2 / 7)

- *Alternativas compuestas solamente por símbolos terminales*

$$(6) C \rightarrow b$$

$$(8) E \rightarrow c a e$$

Paso	Viejo	Nuevo
0	\emptyset	$\{C, E\}$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos generadores

3 / 7)

- Alternativas compuestas por símbolos *terminales* o de *Viejo*

$$(3) A \rightarrow a C$$

$$(4) B \rightarrow b C a$$

$$(6) C \rightarrow b$$

$$(8) E \rightarrow c a e$$

Paso	Viejo	Nuevo
0	\emptyset	$\{C, E\}$
1	$\{C, E\}$	$\{A, B, C, E\}$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos generadores

4 / 7)

- Alternativas compuestas por símbolos *terminales* o de *Viejo*

- (1) $S \rightarrow A B$
- (2) $S \rightarrow A b$
- (3) $A \rightarrow a C$
- (4) $B \rightarrow b C a$
- (6) $C \rightarrow b$
- (8) $E \rightarrow c a e$

Paso	Viejo	Nuevo
0	\emptyset	$\{C, E\}$
1	$\{C, E\}$	$\{A, B, C, E\}$
2	$\{A, B, C, E\}$	$\{S, A, B, C, E\}$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos generadores

5 / 7)

Paso	Viejo	Nuevo
0	\emptyset	$\{C, E\}$
1	$\{C, E\}$	$\{A, B, C, E\}$
2	$\{A, B, C, E\}$	$\{S, A, B, C, E\}$
3	$\{S, A, B, C, E\}$	$\{S, A, B, C, E\}$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos generadores)

6 / 7

$$V'_N = \{S, A, B, C, E\}$$

$$V_T = \{a, b, c, d, e\}$$

$$P' = \{$$

$$(1') S \rightarrow A B$$

$$(2') S \rightarrow A b$$

$$(3') A \rightarrow a C$$

$$(4') B \rightarrow b C a$$

$$(5') C \rightarrow b$$

$$(6') E \rightarrow c a e$$

$$\}$$

Operaciones de limpieza

Símbolos útiles e inútiles

Nota (Selección de símbolos generadores

7 / 7)

- *Se han suprimido*
 - *Símbolos no terminales: D, F*
 - *Reglas de producción*
 - (5) $B \rightarrow D b E$
 - (7) $D \rightarrow F b$
 - (9) $F \rightarrow a D d$

Operaciones de limpieza

Símbolos útiles e inútiles

Símbolos útiles

- Símbolos generadores
- Símbolos accesibles

Operaciones de limpieza

Símbolos útiles e inútiles

Definición (Símbolo accesible)

Sea $X \in V = V_N \cup V_T$

se dice que X es *accesible* si

$$\exists S \xrightarrow[G]{*} \alpha X \beta$$

donde $\alpha, \beta \in V^*$

Operaciones de limpieza

Símbolos útiles e inútiles

Algoritmo (Selección de símbolos accesibles)

1 / 2

- **Entrada**

- $G' = (V'_N, V_T, P', S)$

- Gramática de contexto libre sin símbolos no generadores.*

- **Salida**

- $G'' = (V''_N, V'_T, P'', S)$

- Gramática de contexto libre sin símbolos no accesibles.*

Operaciones de limpieza

Símbolos útiles e inútiles

Algoritmo (Selección de símbolos accesibles

2 / 2)

inicio $Viejo \leftarrow \{S\}$ $Nuevo \leftarrow \{X \mid X \in (V'_N \cup V_T)\}$ $\wedge \exists S \rightarrow \alpha X \beta \in P' \wedge \alpha, \beta \in (V'_N \cup V_T)^*\}$ **mientras** ($Nuevo \neq Viejo$) **hacer** $Viejo \leftarrow Nuevo$ $Nuevo \leftarrow Viejo \cup \{X \mid \exists A \rightarrow \alpha X \beta \in P' \wedge A \in Viejo$ $\wedge X \in (V'_N \cup V_T) \wedge \alpha, \beta \in (V'_N \cup V_T)^*\}$ **fin_mientras** $V''_N \leftarrow Nuevo \cap V'_N$ $V'_T \leftarrow Nuevo \cap V_T$ $P'' \leftarrow \{A \rightarrow \alpha \mid A \rightarrow \alpha \in P' \wedge A \in V''_N \wedge \alpha \in (V''_N \cup V'_T)^*\}$ **fin**

Operaciones de limpieza

Símbolos útiles e inútiles

Nota (Selección de símbolos accesibles)

*Las reglas de la gramática G'' se obtienen a partir de las reglas de la gramática G' que sólo tienen **símbolos accesibles**.*

$$P'' \leftarrow \{A \rightarrow \alpha \mid A \rightarrow \alpha \in P' \wedge A \in V''_N \wedge \alpha \in (V''_N \cup V'_T)^*\}$$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos accesibles)

1 / 6

Gramática *sin* símbolos *no* generadores.

$$V'_N = \{S, A, B, C, E\}$$

$$V_T = \{a, b, c, d, e\}$$

$$P' = \{$$

- (1') $S \rightarrow A B$
- (2') $S \rightarrow A b$
- (3') $A \rightarrow a C$
- (4') $B \rightarrow b C a$
- (5') $C \rightarrow b$
- (6') $E \rightarrow c a e$

$$\}$$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos accesibles)

2 / 6

- Reglas del símbolo inicial S

$$(1') S \rightarrow A B$$

$$(2') S \rightarrow A b$$

Paso	Viejo	Nuevo
0	{S}	{S, A, B, b}

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos accesibles)

3 / 6

- Reglas de los símbolos no terminales de *Viejo*

$$(1') S \rightarrow A B$$

$$(2') S \rightarrow A b$$

$$(3') A \rightarrow a C$$

$$(4') B \rightarrow b C a$$

Paso	Viejo	Nuevo
0	{S}	{S, A, B, b}
1	{S, A, B, b}	{S, A, B, C, a, b}

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos accesibles)

4 / 6

- Reglas de los símbolos no terminales de *Viejo*

$$(1') S \rightarrow A B$$

$$(2') S \rightarrow A b$$

$$(3') A \rightarrow a C$$

$$(4') B \rightarrow b C a$$

$$(5') C \rightarrow b$$

Paso	Viejo	Nuevo
0	{S}	{S, A, B, b}
1	{S, A, B, b}	{S, A, B, C, a, b}
2	{S, A, B, C, a, b}	{S, A, B, C, a, b}

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Selección de símbolos accesibles)

5 / 6

$$\begin{aligned} V''_N &= V'_N \cap \text{Nuevo} \\ &= \{S, A, B, C\} \end{aligned}$$

$$\begin{aligned} V'_T &= V_T \cap \text{Nuevo} \\ &= \{a, b\} \end{aligned}$$

$$\begin{aligned} P'' &= \{ \\ (1') & S \rightarrow A B \\ (2') & S \rightarrow A b \\ (3') & A \rightarrow a C \\ (4') & B \rightarrow b C a \\ (5') & C \rightarrow b \\ & \} \end{aligned}$$

Operaciones de limpieza

Símbolos útiles e inútiles

Nota (Selección de símbolos accesibles

6 / 6)

- *Se han suprimido*
 - *Símbolo no terminal: E*
 - *Símbolos terminales: c, d, e*
 - *Regla de regla*
(6') $E \rightarrow c a e$

Operaciones de limpieza

Símbolos útiles e inútiles

Nota (Orden de aplicación de los algoritmos)

- *Los algoritmos se deben aplicar en el siguiente orden:*
 - 1º *Selección de símbolos generadores*
 - 2º *Selección de símbolos accesibles*
- *Si se aplican en el **orden inverso** entonces **no** se garantiza que la gramática resultante tenga todos sus símbolos accesibles.*

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Orden incorrecto de los algoritmos de limpieza 1 / 4)

$$G = (V_N, V_T, P, S)$$

$$V_N = \{S, A, B, C, D, E, F\}$$

$$V_T = \{a, b, c, d, e\}$$

$$P = \{$$

$$(1) S \rightarrow A B$$

$$(2) S \rightarrow A b$$

$$(3) A \rightarrow a C$$

$$(4) B \rightarrow b C a$$

$$(5) B \rightarrow D b E$$

$$(6) C \rightarrow b$$

$$(7) D \rightarrow F b$$

$$(8) E \rightarrow c a e$$

$$(9) F \rightarrow a D d$$

$$\}$$

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Orden incorrecto de los algoritmos de limpieza 2 / 4)

*Selección de símbolos **acesibles***

Paso	Viejo	Nuevo
0	{S}	{S, A, B, b}
1	{S, A, B, b}	{S, A, B, C, D, E, a, b}
2	{S, A, B, C, D, E, a, b}	{S, A, B, C, D, E, F, a, b, c}
3	{S, A, B, C, D, E, F, a, b, c, d, e}	{S, A, B, C, D, E, F, a, b, c, d, e}

Nota

No se ha eliminado ningún símbolo.

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Orden incorrecto de los algoritmos de limpieza 3 / 4)

Selección de símbolos generadores

Paso	Viejo	Nuevo
0	\emptyset	{C, E}
1	{C, E}	{A, B, C, E}
2	{A, B, C, E}	{S, A, B, C, E}
3	{S, A, B, C, E}	{S, A, B, C, E}

Nota

*Se han eliminado los símbolos no terminales **D** y **F**.*

Operaciones de limpieza

Símbolos útiles e inútiles

Ejemplo (Orden incorrecto de los algoritmos de limpieza 4 / 4)

$$G = (V_N, V_T, P, S)$$

$$V_N = \{S, A, B, C, E\}$$

$$V_T = \{a, b, c, d, e\}$$

$$P = \{$$

$$(1) S \rightarrow A B$$

$$(2) S \rightarrow A b$$

$$(3) A \rightarrow a C$$

$$(4) B \rightarrow b C a$$

$$(6) C \rightarrow b$$

$$(8) E \rightarrow c a e$$

$$\}$$

Nota

Los símbolos E , c , d y e no son accesibles.

Contenido de la sección

- 4 Operaciones de limpieza
 - Símbolos útiles e inútiles
 - Reglas superfluas
 - Gramática propia

Operaciones de limpieza

Reglas superfluas

Tipos de reglas superfluas

(1) Regla unitaria

$$A \rightarrow B \in P$$

donde $A, B \in V_N$

(2) Regla épsilon

$$A \rightarrow \epsilon \in P$$

donde $A \in V_N$

Operaciones de limpieza

Reglas superfluas

Nota (Reglas superfluas: características)

- Las **reglas unitarias** y las reglas ϵ
 - *Ralentizan la derivación*
 - *Pueden facilitar el diseño de la gramática.*

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ)

1 / 3)

- $G = (V_N, V_T, P, S)$

 $P = \{$ (1) $S \rightarrow A B C$ (2) $A \rightarrow a A$ (3) $A \rightarrow \epsilon$ (4) $B \rightarrow b B$ (5) $B \rightarrow \epsilon$ (6) $C \rightarrow c C$ (7) $C \rightarrow \epsilon$ $\}$

$$L(G) = \{a^i b^j c^k \mid i, j, k \geq 0\}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ)

2 / 3

$$S \xrightarrow{1} \underline{A} B C$$

$$\xrightarrow{3} \underline{\epsilon} B C = B C$$

$$\xrightarrow{5} \underline{\epsilon} C = C$$

$$\xrightarrow{7} \underline{\epsilon}$$

Nota

Sería mejor usar la regla

$$S \rightarrow \epsilon$$

y obtener la derivación

$$S \Rightarrow \epsilon$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ)

3 / 3

Si se reescribe el conjunto de reglas de la gramática

- $$P' = \{$$
- | | |
|------------------------------|---------------------------|
| (1) $S \rightarrow \epsilon$ | (8) $S \rightarrow A B C$ |
| (2) $S \rightarrow A$ | (9) $A \rightarrow a A$ |
| (3) $S \rightarrow B$ | (10) $A \rightarrow a$ |
| (4) $S \rightarrow C$ | (11) $B \rightarrow b B$ |
| (5) $S \rightarrow A B$ | (12) $B \rightarrow b$ |
| (6) $S \rightarrow A C$ | (13) $C \rightarrow c C$ |
| (7) $S \rightarrow B C$ | (14) $C \rightarrow c$ |
- $$\}$$

Operaciones de limpieza

Reglas superfluas

Definición (Gramática sin ϵ)

Una **gramática** es **sin ϵ** si cumple una de las siguientes condiciones:

- 1 No tiene ninguna regla ϵ .
- 2 Solamente hay una regla ϵ dicha está asociada al símbolo inicial S

$$S \rightarrow \epsilon$$

y además S no aparece en la parte derecha de ninguna regla de la gramática.

Operaciones de limpieza

Reglas superfluas

Nota (Gramática sin ϵ)

Si G es una gramática **sin** ϵ entonces

$$\epsilon \in L(G) \iff S \rightarrow \epsilon \in P$$

Operaciones de limpieza

Reglas superfluas

Nota

- *Toda gramática de contexto libre se pueden transformar en otra gramática sin ϵ .*
- *Para ello es necesario definir previamente el concepto de símbolo anulable.*

Operaciones de limpieza

Reglas superfluas

Definición (Símbolo anulable)

Un símbolo no terminal A es *anulable* si

$$A \xRightarrow{+} \epsilon$$

Operaciones de limpieza

Reglas superfluas

Teorema (Símbolo anulable)

Un símbolo no terminal A es *anulable* si verifica alguna de las siguientes condiciones:

- $\exists A \rightarrow \epsilon \in P$
- $\exists A \rightarrow \alpha \in P$

y α está compuesta solamente por símbolos anulables.

Operaciones de limpieza

Reglas superfluas

Algoritmo (Obtención de los símbolos anulables)

- **Entrada**

- $G = (V_N, V_T, P, S)$

Gramática de contexto libre sin símbolos inútiles.

- **Salida**

- *Conjunto de símbolos **anulables**.*

Operaciones de limpieza

Reglas superfluas

Algoritmo (Obtención de los símbolos anulables)

inicio

$Viejo \leftarrow \emptyset$

$Nuevo \leftarrow \{A \mid A \in V_N \wedge \exists A \rightarrow \epsilon \in P\}$

mientras ($Nuevo \neq Viejo$) **hacer**

$Viejo \leftarrow Nuevo$

$Nuevo \leftarrow Viejo \cup \{A \mid A \in V_N \wedge \exists A \rightarrow \alpha \in P \wedge \alpha \in Viejo^*\}$

fin_mientras

$Anulables \leftarrow Nuevo$

fin

Operaciones de limpieza

Reglas superfluas

Ejemplo (Símbolos anulables)

- $$P = \{$$
- | | |
|---------------------------|-------------------------------|
| (1) $S \rightarrow A D$ | (7) $C \rightarrow \epsilon$ |
| (2) $S \rightarrow B$ | (8) $D \rightarrow A$ |
| (3) $A \rightarrow C D E$ | (9) $D \rightarrow b$ |
| (4) $B \rightarrow C E$ | (10) $E \rightarrow S$ |
| (5) $C \rightarrow S$ | (11) $E \rightarrow a$ |
| (6) $C \rightarrow a$ | (12) $E \rightarrow \epsilon$ |
- $$\}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Símbolos anulables)

Paso	Viejo	Nuevo
0	\emptyset	$\{C, E\}$
1	$\{C, E\}$	$\{B, C, E\}$
2	$\{B, C, E\}$	$\{S, B, C, E\}$
3	$\{S, B, C, E\}$	$\{S, B, C, E\}$

Anulables = $\{S, B, C, E\}$

Operaciones de limpieza

Reglas superfluas

Algoritmo (Obtención de una gramática sin ϵ)

1 / 3

- **Entrada**

- $G = (V_N, V_T, P, S)$ Gramática sin símbolos inútiles
- Conjunto de símbolos *anulables* de G

- **Salida**

- $G' = (V'_N, V_T, P', S')$
Gramática sin ϵ

Operaciones de limpieza

Reglas superfluas

Algoritmo (Obtención de una gramática sin ϵ)

2/ 3

inicio

$$P' \leftarrow \{A \rightarrow \alpha \mid A \rightarrow \alpha \in P \wedge \alpha \neq \epsilon$$

$$\wedge \alpha \text{ no tiene símbolos anulables}\}$$

para $(A \rightarrow \alpha \in P) \wedge (\alpha \text{ contiene símbolos anulables})$ **hacer**

si $\alpha = \alpha_0 B_1 \alpha_1 \dots B_k \alpha_k$

$\wedge \forall i (B_i \in \text{Anulables})$

$\wedge \forall j (\alpha_j \text{ no contiene ningún símbolo anulable})$

entonces $P' \leftarrow P' \cup \{A \rightarrow \alpha_0 X_1 \alpha_1 \dots X_k \alpha_k \mid$

$\forall i (X_i = B_i \vee X_i = \epsilon) \wedge \alpha_0 X_1 \alpha_1 \dots X_k \alpha_k \neq \epsilon\}$

fin_si

fin_para

...

Operaciones de limpieza

Reglas superfluas

Algoritmo (Obtención de una gramática sin ϵ)

3/ 3)

...

si $S \in \text{Anulables}$

entonces

si S aparece en la parte derecha de una regla de P'

entonces

$$V'_N = V_N \cup \{S'\}$$

$$P' \leftarrow P' \cup \{S' \rightarrow \epsilon, S' \rightarrow S\}$$

si_no

$$P' \leftarrow P' \cup \{S \rightarrow \epsilon\}$$

fin_si

fin_si

fin

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ) 1 / 9

Paso 0: reglas sin símbolos anulables y que no son reglas ϵ :

$$\begin{aligned} P' &= \{ \\ (1) \quad &S \longrightarrow AD \\ (6) \quad &C \longrightarrow a \\ (8) \quad &D \longrightarrow A \\ (9) \quad &D \longrightarrow b \\ (11) \quad &E \longrightarrow a \\ &\} \end{aligned}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ) 2 / 9

Paso 1: se procesa la regla $S \rightarrow B$ que contiene el símbolo anulable B

$$S \rightarrow B$$

$$S \rightarrow \epsilon$$

Solamente se añade a P' la primera regla porque la otra es una regla ϵ .

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ) 3 / 9

Paso 2: se procesa la regla $A \rightarrow CDE$ que contiene los símbolos anulables C y E

$$A \rightarrow CDE$$

$$A \rightarrow DE$$

$$A \rightarrow CD$$

$$A \rightarrow D$$

Se añaden a P' porque no son reglas ϵ .

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ)

4 / 9

Paso 3: se procesa la regla $B \rightarrow CE$ que contiene los símbolos anulables C y E

$$B \rightarrow CE$$

$$B \rightarrow E$$

$$B \rightarrow C$$

$$B \rightarrow \epsilon$$

Solamente se añaden a P' las tres primeras reglas porque la otra es una regla ϵ .

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ) 5 / 9

Paso 4: La regla $C \rightarrow S$ contiene el símbolo anulable S lo que provoca la generación de las dos reglas siguientes:

$$C \rightarrow S$$

$$C \rightarrow \epsilon$$

Solamente se añade a P' la primera regla porque la otra es una regla ϵ .

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ) 6 / 9

Paso 5: se procesa la regla $E \rightarrow S$ que contiene el símbolo anulable S

$$E \rightarrow S$$

$$E \rightarrow \epsilon$$

Solamente se añade a P' la primera regla porque la otra es una regla ϵ .

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ)

7 / 9)

Paso final: al ser el símbolo S anulable, se añaden a la gramática las siguientes reglas:

$$S' \longrightarrow \epsilon$$

$$S' \longrightarrow S$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ)

8 / 9

Se han suprimido las reglas ϵ

$$(7) \quad C \longrightarrow \epsilon$$

$$(12) \quad E \longrightarrow \epsilon$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Obtención de una gramática sin ϵ)

9 / 9

Conjunto final de reglas de producción:

$$\begin{aligned} P' = \{ & \\ & S' \rightarrow \epsilon \mid S \\ & S \rightarrow AD \mid B \\ & A \rightarrow CDE \mid DE \mid CD \mid D \\ & B \rightarrow CE \mid E \mid C \\ & C \rightarrow S \mid a \\ & D \rightarrow A \mid b \\ & E \rightarrow S \mid a \\ & \} \end{aligned}$$

Operaciones de limpieza

Reglas superfluas

Definición (Regla unitaria)

$$A \rightarrow B \in P$$

donde $A, B \in V_N$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Gramática con reglas unitarias)

- $$P = \{$$
- (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + T$
 - (3) $E \rightarrow T$
 - (4) $T \rightarrow T * F$
 - (5) $T \rightarrow F$
 - (6) $F \rightarrow \text{número}$
 - (7) $F \rightarrow \text{identificador}$
 - (8) $F \rightarrow (E)$
- $$\}$$

Operaciones de limpieza

Reglas superfluas

Nota (Reglas que no son unitarias)

(6) $F \rightarrow$ **número**

(7) $F \rightarrow$ **identificador**

*porque **número** e **identificador** son *terminales*.*

Operaciones de limpieza

Reglas superfluas

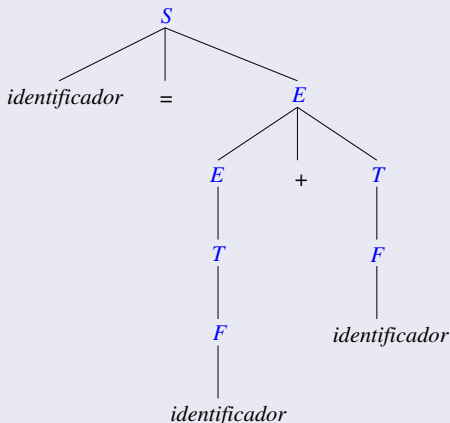
Ejemplo (Ineficiencia de las reglas unitarias)

$$\begin{array}{l}
 S \xRightarrow{1} \underline{\text{identificador}} = E \\
 \xRightarrow{2} \text{identificador} = \underline{E} + T \\
 \xRightarrow{3} \text{identificador} = \underline{T} + T \\
 \xRightarrow{5} \text{identificador} = \underline{F} + T \\
 \xRightarrow{7} \text{identificador} = \underline{\text{identificador}} + T \\
 \xRightarrow{4} \text{identificador} = \text{identificador} + \underline{F} \\
 \xRightarrow{7} \text{identificador} = \text{identificador} + \underline{\text{identificador}}
 \end{array}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Árbol sintáctico asociado a la derivación)



Operaciones de limpieza

Reglas superfluas

Ejemplo (Ineficiencia de las reglas unitarias)

La derivación anterior se podría simplificar eliminando reglas unitarias

$$S \implies \underline{\text{identificador}} = E$$

$$\implies \text{identificador} = \underline{E} + T$$

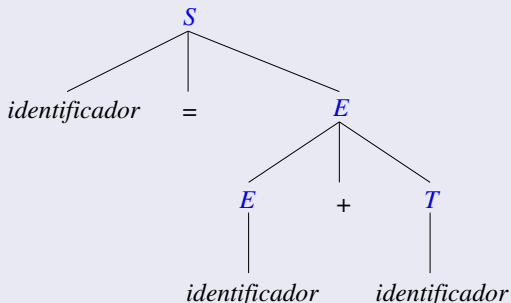
$$\implies \text{identificador} = \underline{\text{identificador}} + T$$

$$\implies \text{identificador} = \text{identificador} + \underline{\text{identificador}}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Árbol sintáctico asociado a la derivación)



Operaciones de limpieza

Reglas superfluas

Definición

Sea G una gramática sin ϵ

Si $A \in V_N$ entonces el conjunto de símbolos no terminales *accesibles* desde A por medio de *reglas unitarias* se define como:

$$N_A = \{B \mid B \in V_N \wedge A \xRightarrow{*} B\}$$

Operaciones de limpieza

Reglas superfluas

Teorema

- Si $A, B \in V_N$ entonces $B \in N_A$ si y sólo si verifica alguna de las siguientes condiciones:
 - $B = A$
 - $\exists A \rightarrow B \in P$
 - $\exists A \xrightarrow{*} C$ y $C \rightarrow B \in P$.

Operaciones de limpieza

Reglas superfluas

Algoritmo (Símbolos no terminales accesibles mediante reglas unitarias)

inicio

$Viejo \leftarrow \emptyset$

$Nuevo \leftarrow \{A\}$

mientras ($Nuevo \neq Viejo$) **hacer**

$Viejo \leftarrow Nuevo$

$Nuevo \leftarrow Viejo \cup \{B \mid A \rightarrow B \in P \wedge A \in Viejo\}$

fin_mientras

$N_A \leftarrow Nuevo$

fin

Operaciones de limpieza

Reglas superfluas

Ejemplo (Símbolos no terminales accesibles mediante reglas unitarias)

$$P = \{$$

- (1) $S \rightarrow \text{identificador} = E$
- (2) $E \rightarrow E + T$
- (3) $E \rightarrow T$
- (4) $T \rightarrow T * F$
- (5) $T \rightarrow F$
- (6) $F \rightarrow \text{número}$
- (7) $F \rightarrow \text{identificador}$
- (8) $F \rightarrow (E)$

$$\}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Símbolos no terminales accesibles mediante reglas unitarias)

Cálculo de N_E

Paso	Viejo	Nuevo
0	\emptyset	$\{E\}$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Símbolos no terminales accesibles mediante reglas unitarias)

Cálculo de N_E

Paso	Viejo	Nuevo
0	\emptyset	$\{E\}$
1	$\{E\}$	$\{E, T\}$

Puesto que (3) $E \rightarrow T$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Símbolos no terminales accesibles mediante reglas unitarias)

Cálculo de N_E

Paso	Viejo	Nuevo
0	\emptyset	$\{E\}$
1	$\{E\}$	$\{E, T\}$
2	$\{E, T\}$	$\{E, T, F\}$

Puesto que (5) $T \rightarrow F$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Símbolos no terminales accesibles mediante reglas unitarias)

Cálculo de N_E

Paso	Viejo	Nuevo
0	\emptyset	$\{E\}$
1	$\{E\}$	$\{E, T\}$
2	$\{E, T\}$	$\{E, T, F\}$
3	$\{E, T, F\}$	$\{E, T, F\}$

$$N_E = \text{Nuevo} = \{E, T, F\}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Símbolos no terminales accesibles mediante reglas unitarias)

- $N_S = \{S\}$
- $N_E = \{E, T, F\}$
- $N_T = \{T, F\}$
- $N_F = \{F\}$

Operaciones de limpieza

Reglas superfluas

Algoritmo (Eliminación de reglas unitarias)

- **Entrada**

- $G = (V_N, V_T, P, S)$ Gramática de contexto libre
- Conjuntos $N_A \forall A \in V_N$

- **Salida**

- $G' = (V'_N, V_T, P', S)$ Gramática *sin* reglas unitarias.

Operaciones de limpieza

Reglas superfluas

Algoritmo (Eliminación de reglas unitarias)

inicio

$P' \leftarrow \emptyset$

para cada $A \in V_N$ **hacer**

para cada $B \in N_A$ **hacer**

si $B \rightarrow \alpha \in P$ *no es una regla unitaria*

entonces $P' \leftarrow P' \cup \{A \rightarrow \alpha\}$

fin_si

fin_para

fin_para

$V'_N \leftarrow \{A \mid A \in V_N \wedge \exists A \rightarrow \alpha \in P'\}$

fin

Operaciones de limpieza

Reglas superfluas

Ejemplo (Eliminación de reglas unitarias)

- $$P = \{$$
- (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + T$
 - (3) $E \rightarrow T$
 - (4) $T \rightarrow T * F$
 - (5) $T \rightarrow F$
 - (6) $F \rightarrow \text{número}$
 - (7) $F \rightarrow \text{identificador}$
 - (8) $F \rightarrow (E)$
- $$\}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Eliminación de reglas unitarias)

*Paso 1: $N_S = \{S\}$, se añaden a P' todas las reglas de S
(ninguna regla es unitaria)*

$S \longrightarrow \text{identificador} = E$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Eliminación de reglas unitarias)

Paso 2: $N_E = \{E, T, F\}$, las alternativas no unitarias de E , T y F se convierten en alternativas de E

$E \rightarrow E + T$

$E \rightarrow T * F$

$E \rightarrow (E)$

$E \rightarrow$ **identificador**

$E \rightarrow$ **número**

Operaciones de limpieza

Reglas superfluas

Ejemplo (Eliminación de reglas unitarias)

Paso 3: $N_T = \{T, F\}$, las alternativas no unitarias de T y F se convierten en alternativas de T .

$T \rightarrow T * F$

$T \rightarrow (E)$

$T \rightarrow$ **identificador**

$T \rightarrow$ **número**

Operaciones de limpieza

Reglas superfluas

Ejemplo (Eliminación de reglas unitarias)

Paso 4: $N_F = \{F\}$, se añaden a P' todas las reglas de F (ninguna regla es unitaria)

$F \rightarrow (E)$

$F \rightarrow$ **identificador**

$F \rightarrow$ **número**

Operaciones de limpieza

Reglas superfluas

Ejemplo (Eliminación de reglas unitarias)

Nuevo conjunto de reglas de producción

$$P' = \left\{ \begin{array}{l} S \rightarrow \text{identificador} = E \\ E \rightarrow E + T \mid T * F \mid (E) \mid \text{identificador} \mid \text{número} \\ T \rightarrow T * F \mid (E) \mid \text{identificador} \mid \text{número} \\ F \rightarrow (E) \mid \text{identificador} \mid \text{número} \\ \} \end{array} \right.$$

Operaciones de limpieza

Reglas superfluas

Nota (Eliminación de reglas unitarias)

*Al eliminar las reglas unitarias, algunos símbolos se pueden convertir en **inútiles**.*

Operaciones de limpieza

Reglas superfluas

Ejemplo (Aparición de símbolos inútiles)

1 / 4)

Gramática sin ϵ

$$P' = \left\{ \begin{array}{l} S' \rightarrow \epsilon \mid S \\ S \rightarrow A D \mid B \\ A \rightarrow C D E \mid D E \mid C D \mid D \\ B \rightarrow C E \mid E \mid C \\ C \rightarrow S \mid a \\ D \rightarrow A \mid b \\ E \rightarrow S \mid a \end{array} \right\}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Aparición de símbolos inútiles)

2 / 4

Conjuntos de símbolos no terminales accesibles mediante reglas unitarias

$$N_{S'} = \{S', S, B, C, E\}$$

$$N_S = \{S, B, C, E\}$$

$$N_A = \{A, D\}$$

$$N_B = \{S, B, C, E\}$$

$$N_C = \{S, B, C, E\}$$

$$N_D = \{A, D\}$$

$$N_E = \{S, B, C, E\}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Aparición de símbolos inútiles)

3 / 4

Gramática generada por el algoritmo que elimina reglas unitarias

$$P' = \{$$
$$S' \longrightarrow \epsilon \mid A D \mid C E \mid a \mid c$$
$$S \longrightarrow A D \mid C E \mid a \mid c$$
$$A \longrightarrow C D E \mid D E \mid C D \mid b$$
$$B \longrightarrow C E \mid A D \mid a \mid c$$
$$C \longrightarrow A D \mid C E \mid a \mid c$$
$$D \longrightarrow C D E \mid D E \mid C D \mid b$$
$$E \longrightarrow A D \mid C E \mid a \mid c$$
$$\}$$

Operaciones de limpieza

Reglas superfluas

Ejemplo (Aparición de símbolos inútiles)

4 / 4

- Los símbolos S y B *no son accesibles*, porque no aparecen en la parte derecha de ninguna regla de producción.
- Por tanto, estos dos símbolos son *inútiles* y se pueden omitir.

Contenido de la sección

- 4 Operaciones de limpieza
 - Símbolos útiles e inútiles
 - Reglas superfluas
 - Gramática propia

Operaciones de limpieza

Gramática propia

Definición (Gramática sin ciclos)

Una **gramática es sin ciclos** si no tiene derivaciones de la forma

$$A \xrightarrow{+} A$$

Nota

- *Los ciclos dificultan el proceso de generación de las palabras y por ello deben evitarse.*
- *La aparición de ciclos se debe a la presencia reglas unitarias o reglas ϵ .*

Operaciones de limpieza

Gramática propia

Ejemplo (Gramática con ciclos)

Reglas de una gramática

$$P = \{$$

...

$$A \rightarrow BC$$
$$B \rightarrow \epsilon$$
$$C \rightarrow A$$

...

$$\}$$

Ciclo

$$A \Rightarrow BC \Rightarrow C \Rightarrow A$$

Operaciones de limpieza

Gramática propia

Nota

Obviamente, una gramática sin ϵ que no tenga reglas unitarias no puede generar ciclos.

Operaciones de limpieza

Gramática propia

Definición (Gramática propia)

*Se dice que una **gramática es propia** si es una gramática sin ciclos, sin reglas ϵ , ni símbolos inútiles.*

Recursividad y factorización

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad
- 4 Operaciones de limpieza
- 5 Recursividad y factorización**
- 6 Formas normales
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos

Recursividad y factorización

- 5 Recursividad y factorización
 - Recursividad
 - Factorización por la izquierda
 - Eliminación de la recursividad inmediata y factorización por la izquierda

Contenido de la sección

- 5 Recursividad y factorización
 - Recursividad
 - Factorización por la izquierda
 - Eliminación de la recursividad inmediata y factorización por la izquierda

Recursividad y factorización

Recursividad

Recursividad

- Gramática con recursividad inmediata
- Gramática con recursividad general
- Eliminación de la recursividad inmediata por la izquierda
- Eliminación de la recursividad general por la izquierda

Recursividad y factorización

Recursividad

Recursividad

- Gramática con recursividad inmediata
- Gramática con recursividad general
- Eliminación de la recursividad inmediata por la izquierda
- Eliminación de la recursividad general por la izquierda

Recursividad y factorización

Recursividad

Definición (Gramática con recursividad inmediata)

Una gramática posee recursividad inmediata si

$$\exists A \rightarrow \alpha A \beta \in P$$

donde

- $A \in V_N$
- $\alpha\beta \in V^+ = (V_N \cup V_T)^+$

Recursividad y factorización

Recursividad

Ejemplo (Gramática con recursividad inmediata)

$$P = \{$$

- (1) $S \rightarrow \mathbf{a} S \mathbf{a}$
- (2) $S \rightarrow \mathbf{a} A \mathbf{a}$
- (3) $A \rightarrow \mathbf{b} A \mathbf{b}$
- (4) $A \rightarrow \mathbf{c}$

$$\}$$

Recursividad y factorización

Recursividad

Definición (Recursividad inmediata por la izquierda)

Una gramática posee recursividad inmediata por la izquierda si

$$\exists A \rightarrow A \beta \in P$$

donde

- $A \in V_N$
- $\beta \in V^+ = (V_N \cup V_T)^+$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad inmediata por la izquierda) 1 / 7)

$$P = \{$$

- (1) $S \rightarrow S a$
- (2) $S \rightarrow A a$
- (3) $A \rightarrow A b$
- (4) $A \rightarrow c$

$$\}$$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad inmediata por la izquierda

2 / 7)

 $P = \{$

(1) $\langle \text{asignación} \rangle \rightarrow \mathbf{identificador} = \langle \text{expresión} \rangle$

(2) $\langle \text{expresión} \rangle \rightarrow \langle \text{expresión} \rangle + \langle \text{sumando} \rangle$

(3) $\langle \text{expresión} \rangle \rightarrow \langle \text{sumando} \rangle$

(4) $\langle \text{sumando} \rangle \rightarrow \langle \text{sumando} \rangle * \langle \text{factor} \rangle$

(5) $\langle \text{sumando} \rangle \rightarrow \langle \text{factor} \rangle$

(6) $\langle \text{factor} \rangle \rightarrow \mathbf{número}$

(7) $\langle \text{factor} \rangle \rightarrow \mathbf{identificador}$

(8) $\langle \text{factor} \rangle \rightarrow (\langle \text{expresión} \rangle)$

 $\}$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad inmediata por la izquierda)

3 / 7

- $P = \{$
 - (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + E$
 - (3) $E \rightarrow E * E$
 - (4) $E \rightarrow (E)$
 - (5) $E \rightarrow \text{número}$
 - (6) $E \rightarrow \text{identificador}$ $\}$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad inmediata por la izquierda

4 / 7)

- *Sentencia de asignación múltiple del lenguaje C*

$$P = \{$$

- (1) $S \rightarrow L E$
- (2) $L \rightarrow L \text{ identificador} =$
- (3) $L \rightarrow \text{identificador} =$
- (4) $E \rightarrow E + T$

...

$$\}$$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad inmediata por la izquierda

5 / 7)

- *Derivación recursiva por la izquierda*

$$S \xRightarrow{1} \underline{L} E$$

$$\xRightarrow{2} \underline{L \text{ identificador}} = E$$

$$\xRightarrow{2} \underline{L \text{ identificador}} = \text{identificador} = E$$

$$\xRightarrow{3} \underline{\text{identificador}} = \text{identificador} = \text{identificador} = E$$

...

Recursividad y factorización

Recursividad

Ejemplo (Recursividad inmediata por la izquierda) 6 / 7

- *Lista de parámetros de un procedimiento o función:*

$$P = \{$$

- (1) $S \rightarrow \text{identificador} (L)$
- (2) $L \rightarrow L , \text{identificador}$
- (3) $L \rightarrow \text{identificador}$

...

$$\}$$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad inmediata por la izquierda 7 / 7)

- *Componentes de un array:*

$$P = \left\{ \begin{array}{l} (1) S \rightarrow \text{identificador } D \\ (2) D \rightarrow D [\text{número}] \\ (3) D \rightarrow [\text{número}] \\ \dots \\ \end{array} \right\}$$

Recursividad y factorización

Recursividad

Definición (Recursividad inmediata por la derecha)

Una gramática posee recursividad inmediata por la derecha si

$$\exists A \rightarrow \alpha A \in P$$

donde

- $A \in V_N$
- $\alpha \in V^+ = (V_N \cup V_T)^+$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad inmediata por la derecha)

$$P = \{$$

- (1) $S \rightarrow a S$
- (2) $S \rightarrow a A$
- (3) $A \rightarrow b A$
- (4) $A \rightarrow c$

$$\}$$

Recursividad y factorización

Recursividad

Recursividad

- Gramática con recursividad inmediata
- Gramática con recursividad general
- Eliminación de la recursividad inmediata por la izquierda
- Eliminación de la recursividad general por la izquierda

Recursividad y factorización

Recursividad

Definición (Gramática con recursividad general)

Una gramática posee recursividad general si

$$\exists A \xRightarrow{+} \alpha A \beta$$

donde

- $A \in V_N$
- $\alpha, \beta \in V^* = (V_N \cup V_T)^*$

Recursividad y factorización

Recursividad

Ejemplo (Gramática con recursividad general

1 / 2)

$$P = \{$$

- (1) $S \rightarrow \mathbf{a} A \mathbf{a}$
- (2) $A \rightarrow \mathbf{a} A \mathbf{a}$
- (3) $A \rightarrow \mathbf{b} S \mathbf{b}$
- (4) $A \rightarrow \mathbf{c}$

$$\}$$

Recursividad y factorización

Recursividad

Ejemplo (Gramática con recursividad general

2 / 2)

$S \xRightarrow{1} \underline{a A a}$

$\xRightarrow{2} \underline{a a A a a}$

$\xRightarrow{3} \underline{a a b S b a a}$

...

Recursividad y factorización

Recursividad

Definición (Recursividad general por la izquierda)

Una gramática posee recursividad general por la izquierda si

$$\exists A \xRightarrow{+} A \beta$$

donde

- $A \in V_N$
- $\beta \in V^+ = (V_N \cup V_T)^+$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad general por la izquierda

1 / 2)

$$P = \{$$

- (1) $S \rightarrow A a$
- (2) $A \rightarrow A a$
- (3) $A \rightarrow S b$
- (4) $A \rightarrow c$

$$\}$$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad general por la izquierda)

2 / 2)

$S \xRightarrow{1} \underline{A} a$

$\xRightarrow{2} \underline{A} a a$

$\xRightarrow{3} \underline{S} b a a$

...

Recursividad y factorización

Recursividad

Definición (Recursividad general por la derecha)

Una gramática posee recursividad general por la derecha si

$$\exists A \xRightarrow{+} \alpha A$$

donde

- $A \in V_N$
- $\alpha \in V^+ = (V_N \cup V_T)^+$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad general por la derecha 1 / 2)

$$P = \{$$

- (1) $S \rightarrow \mathbf{a} A$
- (2) $A \rightarrow \mathbf{a} A$
- (3) $A \rightarrow \mathbf{b} S$
- (4) $A \rightarrow \mathbf{c}$

$$\}$$

Recursividad y factorización

Recursividad

Ejemplo (Recursividad general por la derecha)

2 / 2)

$$S \xRightarrow[1]{} \underline{a A}$$

$$\xRightarrow[2]{} \underline{a a A}$$

$$\xRightarrow[3]{} \underline{a a b S}$$

...

Recursividad y factorización

Recursividad

Nota (Gramática recursiva por la izquierda: **inconveniente**)

- *No se puede realizar el análisis sintáctico descendente con gramáticas recursivas por la izquierda.*
- *Dichas gramáticas deben ser convertidas en gramáticas recursivas por la derecha.*

Recursividad y factorización

Recursividad

Recursividad

- Gramática con recursividad inmediata
- Gramática con recursividad general
- Eliminación de la recursividad inmediata por la izquierda
- Eliminación de la recursividad general por la izquierda

Recursividad y factorización

Recursividad

Algoritmo (Eliminación de la recursividad inmediata por la izquierda)

- **Entrada**

- $G = (V_N, V_T, P, S)$

- Gramática **con** reglas recursivas por la izquierda.*

- **Salida**

- $G' = (V'_N, V_T, P', S)$

- Gramática **sin** reglas recursivas por la izquierda.*

Recursividad y factorización

Recursividad

Algoritmo (Eliminación de la recursividad inmediata por la izquierda)

inicio

$P' \leftarrow \emptyset$

para cada $A \in V_N$ **hacer**

si A no tiene reglas recursivas

entonces se añaden a P' las reglas de A

si no si $(A \rightarrow A\alpha_1|A\alpha_2|\dots|A\alpha_p|\beta_1|\beta_2|\dots|\beta_q \in P)$

donde $\forall i \alpha_i \neq \epsilon$ y $\forall j \beta_j$ no empieza por A

entonces se añaden a P' las reglas

$$A \rightarrow \beta_j|\beta_j A' \quad \forall j \in \{1, 2, \dots, q\}$$

$$A' \rightarrow \alpha_i|\alpha_i A' \quad \forall i \in \{1, 2, \dots, p\}$$

fin_si

fin_si

fin_para

fin

Recursividad y factorización

Recursividad

Nota (Eliminación de la recursividad inmediata por la izquierda)

- Si $A \in V_N$ posee recursividad inmediata entonces A' es un nuevo símbolo no terminal.

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad inmediata por la izquierda 1/7)

$$P = \left\{ \begin{array}{l} S \rightarrow \text{identificador} = E \\ E \rightarrow E + T \mid T * F \mid (E) \mid \text{identificador} \mid \text{número} \\ T \rightarrow T * F \mid (E) \mid \text{identificador} \mid \text{número} \\ F \rightarrow (E) \mid \text{identificador} \mid \text{número} \\ \end{array} \right\}$$

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad inmediata por la izquierda 2/7)

Paso 1: S no tiene recursividad por la izquierda

Se añade a P' la regla de S .

$S \rightarrow \mathbf{identificador} = E$

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad inmediata por la izquierda 3/7)

Paso 2: E tiene una regla recursiva por la izquierda

$$E \longrightarrow E \underbrace{+T}_{\alpha_1} \mid \underbrace{T*F}_{\beta_1} \mid \underbrace{(E)}_{\beta_2} \mid \underbrace{\text{identificador}}_{\beta_3} \mid \underbrace{\text{número}}_{\beta_4}$$

Se añaden a P' las siguientes reglas

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

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad inmediata por la izquierda 4/7)

Paso 3: T tiene una regla recursiva por la izquierda

$$T \longrightarrow T \underbrace{*F}_{\alpha_1} \mid \underbrace{(E)}_{\beta_1} \mid \underbrace{\text{identificador}}_{\beta_2} \mid \underbrace{\text{número}}_{\beta_3}$$

Se añaden a P' las siguientes reglas:

$$\begin{aligned} T &\longrightarrow (E) \mid \text{identificador} \mid \text{número} \mid \\ &\quad (E) T' \mid \text{identificador } T' \mid \text{número } T' \\ T' &\longrightarrow * F \mid * F T' \end{aligned}$$

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad inmediata por la izquierda 5/7)

Paso 4: F no posee reglas recursivas.

Se añaden a P' todas las reglas de F

$F \rightarrow (E) \mid \text{identificador} \mid \text{número}$

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad inmediata por la izquierda 6/7)

$$\begin{aligned}
 P' = \{ & \\
 & S \rightarrow \text{identificador} = E \\
 & E \rightarrow T * F \mid (E) \mid \text{identificador} \mid \text{número} \mid \\
 & \quad T * F E' \mid (E) E' \mid \text{identificador } E' \mid \text{número } E' \\
 & E' \rightarrow + T \mid + T E' \\
 & T \rightarrow (E) \mid \text{identificador} \mid \text{número} \mid \\
 & \quad (E) T' \mid \text{identificador } T' \mid \text{número } T' \\
 & T' \rightarrow * F \mid * F T' \\
 & F \rightarrow (E) \mid \text{identificador} \mid \text{número} \\
 & \}
 \end{aligned}$$

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad inmediata por la izquierda 7/7)

- G y G' son gramáticas equivalentes

$$S \xRightarrow{G} \underline{\text{identificador}} = E$$

$$\xRightarrow{G} \text{identificador} = E + T$$

$$\xRightarrow{G} \text{identificador} = \underline{\text{identificador}} + T$$

$$\xRightarrow{G} \text{identificador} = \text{identificador} + \underline{\text{identificador}}$$

$$S \xRightarrow{G'} \underline{\text{identificador}} = E$$

$$\xRightarrow{G'} \text{identificador} = \underline{\text{identificador}} E'$$

$$\xRightarrow{G'} \text{identificador} = \text{identificador} + T$$

$$\xRightarrow{G'} \text{identificador} = \text{identificador} + \underline{\text{identificador}}$$

Recursividad y factorización

Recursividad

Recursividad

- Gramática con recursividad inmediata
- Gramática con recursividad general
- Eliminación de la recursividad inmediata por la izquierda
- Eliminación de la recursividad general por la izquierda

Recursividad y factorización

Recursividad

Algoritmo (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 reglas ϵ , ni símbolos inútiles.*

- **Salida**

- $G' = (V'_N, V_T, P', S)$

- Gramática **sin** recursividad por la izquierda.*

Recursividad y factorización

Recursividad

Algoritmo (Eliminación de la recursividad general por la izquierda)

inicio

$P' \leftarrow \emptyset$

Ordénense los símbolos no terminales de la gramática: $\{A_1, A_2, \dots, A_n\}$

para i **de** 1 **a** n **hacer**

para j **de** 1 **a** $i - 1$ **hacer**

si $A_i \rightarrow A_j \gamma \in P$

entonces

 Añadir a P' las reglas $A_i \rightarrow \delta_1 \gamma \mid \dots \mid \delta_k \gamma$

 donde $A_j \rightarrow \delta_1 \mid \dots \mid \delta_k$ son las reglas actuales de A_j

fin_si

fin_para

 Eliminar la recursividad inmediata por la izquierda de las reglas de A_i .

fin_para

fin

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad general por la izquierda 1 / 8)

$$P = \{$$

- (1) $S \rightarrow A B$
- (2) $S \rightarrow c$
- (3) $A \rightarrow B b$
- (4) $A \rightarrow S d$
- (5) $A \rightarrow a$
- (6) $B \rightarrow S b$
- (7) $B \rightarrow A a$

$$\}$$

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad general por la izquierda 2 / 8)

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

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad general por la izquierda 3 / 8)

- *Paso exterior 1: reglas de producción de S*
 - *Paso interior 1*

S **no** tiene ninguna regla que comience por un símbolo con un número de orden inferior al suyo.
 - *Eliminación de la recursividad inmediata de S*

S **no** tiene recursividad inmediata por la izquierda.

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad general por la izquierda 4 / 8)

- *Paso exterior 2: reglas de producción de A*

- *Paso interior 1:*

Sustitución de las reglas de A que comienzan por S:

- *La regla(4) $A \rightarrow S d$ se sustituye por $A \rightarrow A B d \mid c d$*
 - *Nuevas reglas de A:*

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

- *Eliminación de la recursividad inmediata de A:*

$$\begin{array}{l} A \rightarrow B b \mid c d \mid a \mid \\ \quad \quad \quad B b A' \mid c d A' \mid a A' \\ A' \rightarrow B d \mid B d A' \end{array}$$

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad general por la izquierda 5 / 8)

- *Paso exterior 3: reglas de producción de B*

- *Paso interior 1:*

Sustitución de las reglas de B que comienzan por S

- *La regla (6) $B \rightarrow S b$ se sustituye por*

$$B \rightarrow A B b \mid c b$$

- *Nuevas reglas de B*

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

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad general por la izquierda 6 / 8)

- *Paso exterior 3: reglas de producción de B*

- *Paso interior 2:*

Sustitución de las reglas de B que comienzan por A

- *La regla $B \rightarrow A B b$ se sustituye por las reglas*

$$\begin{array}{l}
 B \rightarrow B b B b \mid c d B b \mid a B b \mid \\
 \qquad \qquad \qquad B b A' B b \mid c d A' B b \mid a A' B b
 \end{array}$$

- *y la regla $B \rightarrow A a$ se sustituye por las reglas*

$$\begin{array}{l}
 B \rightarrow B b a \mid c d a \mid a a \mid \\
 \qquad \qquad \qquad B b A' a \mid c d A' a \mid a A' a
 \end{array}$$

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad general por la izquierda 7 / 8)

- *Paso exterior 3: reglas de producción de B*
 - *Nuevas reglas de B:*

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

Recursividad y factorización

Recursividad

Ejemplo (Eliminación de la recursividad general por la izquierda 8 / 8)

- *Paso exterior 3: reglas de producción de B*
 - *Eliminación de la recursividad inmediata de B*

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

Contenido de la sección

- 5 Recursividad y factorización
 - Recursividad
 - **Factorización por la izquierda**
 - Eliminación de la recursividad inmediata y factorización por la izquierda

Recursividad y factorización

Factorización por la izquierda

Factorización por la izquierda

- Considérense las siguientes reglas de producción:

$$S \longrightarrow \text{si } E \text{ entonces } S \text{ si_no } S \text{ fin_si}$$

$$S \longrightarrow \text{si } E \text{ entonces } S \text{ fin_si}$$

- Si se recibe el componente léxico **si**, no se sabe aún qué regla de S se debe utilizar
- Se puede **posponer** esta decisión si se utilizan las siguientes reglas de producción.

$$S \longrightarrow \text{si } E \text{ entonces } S \text{ } S'$$

$$S' \longrightarrow \text{si_no } S \text{ fin si}$$

$$S' \longrightarrow \text{fin_si}$$

Recursividad y factorización

Factorización por la izquierda

Factorización por la izquierda

- En general, si

$$A \longrightarrow \alpha \beta_1 \mid \alpha \beta_2 \mid \cdots \mid \alpha \beta_N$$

son reglas de A que comienzan por $\alpha \neq \epsilon$

entonces no se sabe qué alternativa de A utilizar

- **Solución:** factorizar por la izquierda

$$A \longrightarrow \alpha A'$$

$$A' \longrightarrow \beta_1 \mid \beta_2 \mid \cdots \mid \beta_N$$

Recursividad y factorización

Factorización por la izquierda

Nota (Factorización por la izquierda)

El análisis sintáctico descendente requiere que la gramática esté factorizada por la izquierda.

Recursividad y factorización

Factorización por la izquierda

Algoritmo (Factorización por la izquierda)

1/2

- **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.*

Recursividad y factorización

Factorización por la izquierda

Algoritmo (Factorización por la izquierda)

2/2

inicio

para cada $A \in V_N$ **hacer**

mientras A tenga dos reglas actuales con el mismo prefijo **hacer**

si $\alpha \neq \epsilon$ es el *prefijo más largo* de dos o más alternativas de A

entonces *sustituir todas las reglas de* A

$$A \longrightarrow \alpha \beta_1 \mid \cdots \mid \alpha \beta_p \mid \gamma_1 \mid \cdots \mid \gamma_q$$

donde γ_i no empieza por $\alpha \forall i \in \{1, 2, \dots, q\}$

por las reglas

$$A \longrightarrow \alpha A' \mid \gamma_1 \mid \cdots \mid \gamma_q$$

$$A' \longrightarrow \beta_1 \mid \cdots \mid \beta_p$$

fin_si

fin_mientras

fin_para

fin

Recursividad y factorización

Factorización por la izquierda

Ejemplo (Factorización por la izquierda)

1/4

$$P = \{$$
$$S \longrightarrow A B c \mid A B d e \mid A B d f \mid A B S$$
$$A \longrightarrow a$$
$$B \longrightarrow b$$
$$\}$$

Recursividad y factorización

Factorización por la izquierda

Ejemplo (Factorización por la izquierda)

2/4

- *Paso 1: factorización de las reglas de S (1/2)*
 - $\alpha_1 = ABd$: prefijo más largo

$$S \longrightarrow A B c \mid A B d e \mid A B d f \mid A B S$$

- *Las reglas de S se sustituyen por:*

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

$$S' \longrightarrow e \mid f$$

Recursividad y factorización

Factorización por la izquierda

Ejemplo (Factorización por la izquierda)

3/4

- *Paso 1: factorización de las reglas de S (2/2)*
 - $\alpha_2 = AB$: nuevo prefijo más largo.

$$S \longrightarrow AB d S' \mid AB c \mid AB S$$

- Las reglas actuales de S se sustituyen por:

$$\begin{aligned} S &\longrightarrow AB S'' \\ S'' &\longrightarrow d S' \mid c \mid S \\ S' &\longrightarrow e \mid f \end{aligned}$$

Recursividad y factorización

Factorización por la izquierda

Ejemplo (Factorización por la izquierda)

4/4

- *Pasos 2 y 3*

Las producciones de A y B no requieren factorización.

Recursividad y factorización

Factorización por la izquierda

Ejercicio (Factorización por la izquierda)

$$\begin{aligned}
 P' = \{ & \\
 & S \rightarrow \text{identificador} = E \\
 & E \rightarrow T * F \mid (E) \mid \text{identificador} \mid \text{número} \mid \\
 & \quad T * F E' \mid (E) E' \mid \text{identificador } E' \mid \text{número } E' \\
 & E' \rightarrow + T \mid + T E' \\
 & T \rightarrow (E) \mid \text{identificador} \mid \text{número} \mid \\
 & \quad (E) T' \mid \text{identificador } T' \mid \text{número } T' \\
 & T' \rightarrow * F \mid * F T' \\
 & F \rightarrow (E) \mid \text{identificador} \mid \text{número} \\
 & \}
 \end{aligned}$$

Contenido de la sección

- 5 Recursividad y factorización
 - Recursividad
 - Factorización por la izquierda
 - Eliminación de la recursividad inmediata y factorización por la izquierda

Recursividad y factorización

Eliminación de la recursividad inmediata y factorización por la izquierda

Algoritmo

- **Entrada**

- $G = (V_N, V_T, P, S)$

- Gramática **con** recursividad inmediata por la izquierda.*

- **Salida**

- $G' = (V'_N, V_T, P', S)$

- Gramática **sin** recursividad inmediata por la izquierda y factorizada por la izquierda.*

Recursividad y factorización

Eliminación de la recursividad inmediata y factorización por la izquierda

Algoritmo

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 $A \rightarrow A\alpha_1|A\alpha_2|\dots|A\alpha_p|\beta_1|\beta_2|\dots|\beta_q \in P$

donde $\forall i \alpha_i \neq \epsilon$ y $\forall j \beta_j$ no empieza por A

entonces se añaden a P' las producciones

$A \rightarrow \beta_j A' \quad \forall j \in \{1, 2, \dots, q\}$

$A' \rightarrow \alpha_i A' | \epsilon \quad \forall i \in \{1, 2, \dots, p\}$

fin_si

fin_si

fin_para

fin

Recursividad y factorización

Eliminación de la recursividad inmediata y factorización por la izquierda

Ejemplo (Eliminación de recursividad y factorización 1 / 6)

Gramática sin reglas unitarias

$$\begin{aligned} P = \{ & \\ & S \rightarrow \text{identificador} = E \\ & E \rightarrow E + T \mid T * F \mid (E) \mid \text{identificador} \mid \text{número} \\ & T \rightarrow T * F \mid (E) \mid \text{identificador} \mid \text{número} \\ & F \rightarrow (E) \mid \text{identificador} \mid \text{número} \\ & \} \end{aligned}$$

Recursividad y factorización

Eliminación de la recursividad inmediata y factorización por la izquierda

Ejemplo (Eliminación de recursividad y factorización 2 / 6)

- *Procesamiento de la regla de S*

$$S \rightarrow \mathbf{identificador} = E$$

Se añade a P' porque dicha regla no es recursiva

Recursividad y factorización

Eliminación de la recursividad inmediata y factorización por la izquierda

Ejemplo (Eliminación de recursividad y factorización 3 / 6)

- Procesamiento de las reglas de E

$$E \longrightarrow E \underbrace{+ T}_{\alpha_1} \mid \underbrace{T * F}_{\beta_1} \mid \underbrace{(E)}_{\beta_2} \mid \underbrace{\text{identificador}}_{\beta_3} \mid \underbrace{\text{número}}_{\beta_4}$$

Se añaden a P' las siguientes reglas

$$E \longrightarrow T * F E' \mid (E) E' \mid \text{identificador } E' \mid \text{número } E'$$

$$E' \longrightarrow + T E' \mid \epsilon$$

Recursividad y factorización

Eliminación de la recursividad inmediata y factorización por la izquierda

Ejemplo (Eliminación de recursividad y factorización) 4 / 6

- Procesamiento de las reglas de T

$$T \longrightarrow T \underbrace{*F}_{\alpha_1} \mid \underbrace{(E)}_{\beta_1} \mid \underbrace{\text{identificador}}_{\beta_2} \mid \underbrace{\text{número}}_{\beta_3}$$

Se añaden a P' las siguientes reglas

$$T \longrightarrow (E) T' \mid \text{identificador } T' \mid \text{número } T'$$

$$T' \longrightarrow * F T' \mid \epsilon$$

Recursividad y factorización

Eliminación de la recursividad inmediata y factorización por la izquierda

Ejemplo (Eliminación de recursividad y factorización 5 / 6)

- *Procesamiento de las reglas de F*

$$F \longrightarrow (E) \mid \text{identificador} \mid \text{número}$$

Se añaden a P' porque no son recursivas por la izquierda ni necesitan ser factorizadas.

Recursividad y factorización

Eliminación de la recursividad inmediata y factorización por la izquierda

Ejemplo (Gramática transformada)

6 / 6

$P' = \{$

$S \rightarrow \text{identificador} = E$

$E \rightarrow T * F E' \mid (E) E' \mid \text{identificador } E' \mid \text{número } E'$

$E' \rightarrow + T E' \mid \epsilon$

$T \rightarrow (E) T' \mid \text{identificador } T' \mid \text{número } T'$

$T' \rightarrow * F T' \mid \epsilon$

$F \rightarrow (E) \mid \text{identificador} \mid \text{número}$

$\}$

Formas normales

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad
- 4 Operaciones de limpieza
- 5 Recursividad y factorización
- 6 Formas normales**
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos

Formas normales

- 6 Formas normales
 - Forma normal de Chomsky
 - Forma normal de Greibach

Contenido de la sección

- 6 Formas normales
 - Forma normal de Chomsky
 - Forma normal de Greibach

Formas normales

Forma normal de Chomsky

Definición (Gramática en la forma normal de Chomsky)

- Una gramática está en la *forma normal de Chomsky* (**F.N.C.**) si sus reglas son de la forma:

$$A \longrightarrow B C$$

$$A \longrightarrow a$$

donde

- $A, B, C \in V_N$
- $a \in V_T$

Formas normales

Forma normal de Chomsky

Ejemplo (Gramática en la forma normal de Chomsky 1 / 3)

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

Formas normales

Forma normal de Chomsky

Ejemplo (Gramática en F.N.C.: derivación)

2 / 3

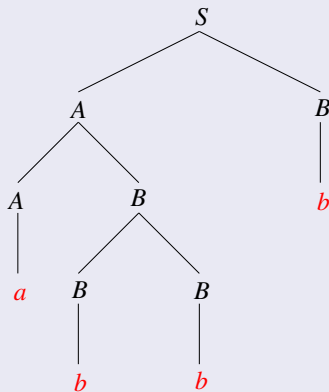
$$\begin{aligned} S &\xRightarrow[1]{} \underline{A} B \\ &\xRightarrow[2]{} \underline{A} B B \\ &\xRightarrow[3]{} \underline{a} B B \\ &\xRightarrow[4]{} a \underline{B} B B \\ &\xRightarrow[5]{} a \underline{b} B B \\ &\xRightarrow[5]{} a b \underline{b} B \\ &\xRightarrow[5]{} a b b \underline{b} \end{aligned}$$

Formas normales

Forma normal de Chomsky

Ejemplo (Gramática en F.N.C.: árbol sintáctico)

3 / 3



Formas normales

Forma normal de Chomsky

Nota (Árboles de la gramáticas en la F. N. C.)

*Los **árboles** sintácticos de las derivaciones de las gramáticas que están en la forma normal de **Chomsky** siempre son **árboles binarios**.*

Formas normales

Forma normal de Chomsky

Nota (Análisis sintáctico)

*El algoritmo de análisis sintáctico **CYK** (Cocke, Younger, Kasami) se aplica a gramáticas que están en la **Forma Normal de Chomsky**.*

Formas normales

Forma normal de Chomsky

Algoritmo (Obtención de la forma normal de Chomsky 1 / 6)

- **Entrada**

- $G = (V_N, V_T, P, S)$
Gramática propia.

- **Salida**

- $G' = (V'_N, V_T, P', S)$
Gramática en la forma normal Chomsky.

Formas normales

Forma normal de Chomsky

Algoritmo (Obtención de la forma normal de Chomsky 2/ 6)

- *Paso 1*

Generación de $G_1 = (V_{N_1}, V_T, P_1, S)$

$$A \longrightarrow B_1 B_2 \cdots B_k \quad \text{donde } k \geq 2$$

$$A \longrightarrow a$$

- *Paso 2*

Generación de $G_2 = (V_{N_2}, V_T, P_2, S)$

$$A \longrightarrow B C$$

$$A \longrightarrow a$$

Formas normales

Forma normal de Chomsky

Nota (Obtención de la forma normal de Chomsky) 3 / 6

Se verifica que $L(G_2) = L(G_1) = L(G) - \{\epsilon\}$.

Formas normales

Forma normal de Chomsky

Algoritmo (Obtención de la forma normal de Chomsky 4 / 6)

Paso 1. Sea $A \rightarrow X_1 X_2 \cdots X_k \in P$

- *Si $k = 1$ entonces*

$A \rightarrow X_1$ se añade a P_1

donde $X_1 \in V_T$, porque la gramática no tiene reglas unitarias.

- *Si $k \geq 2$ entonces*

$A \rightarrow B_1 B_2 \cdots B_k$ se añade a P_1

donde

- *Si $X_i \in V_N$ entonces $B_i = X_i$*
- *Si $X_i = a_i \in V_T$ entonces*
 - *$B_i \in V_{N_1} - V_N$*
 - *$B_i \rightarrow X_i \in P_1$*

Formas normales

Forma normal de Chomsky

Algoritmo (Obtención de la forma normal de Chomsky 5 / 6)

Paso 2. Generación de las reglas de P_2

- Si $A \rightarrow a \in P_1$ entonces $A \rightarrow a \in P_2$.
- Si $A \rightarrow B_1 B_2 \cdots B_k \in P_1$ entonces:
 - Si $k = 2$ entonces $A \rightarrow B_1 B_2 \in P_2$
 - Si $k \geq 3$ entonces se añaden a P_2 las siguientes reglas:

$$\begin{array}{lcl}
 A & \longrightarrow & B_1 C_1 \\
 C_1 & \longrightarrow & B_2 C_2 \\
 & \dots & \\
 C_{k-1} & \longrightarrow & B_{k-2} C_{k-2} \\
 C_{k-2} & \longrightarrow & B_{k-1} B_k
 \end{array}$$

Formas normales

Forma normal de Chomsky

Nota (Obtención de la forma normal de Chomsky 6 / 6)

$\forall i \in \{1, \dots, k - 2\}$ C_i es un nuevo símbolo no terminal.

Formas normales

Forma normal de Chomsky

Ejemplo (Obtención de la forma normal de Chomsky 1 / 3)

Gramática de contexto libre propia.

$$P = \left\{ \begin{array}{l} S \rightarrow a A B \\ A \rightarrow a B b \\ A \rightarrow a \\ B \rightarrow b b \end{array} \right\}$$

Formas normales

Forma normal de Chomsky

Ejemplo (Obtención de la forma normal de Chomsky 2 / 3)

Paso 1

$$P_1 = \left\{ \begin{array}{l} S \longrightarrow B_1 A B \\ A \longrightarrow B_1 B B_2 \\ A \longrightarrow a \\ B \longrightarrow B_2 B_2 \\ B_1 \longrightarrow a \\ B_2 \longrightarrow b \\ \end{array} \right\}$$

Formas normales

Forma normal de Chomsky

Ejemplo (Obtención de la forma normal de Chomsky 3 / 3)

Paso 2

$$P_2 = \left\{ \begin{array}{ll} S \rightarrow B_1 C_1 & A \rightarrow a \\ C_1 \rightarrow A B & B \rightarrow B_2 B_2 \\ A \rightarrow B_1 C_2 & B_1 \rightarrow a \\ C_2 \rightarrow B B_2 & B_2 \rightarrow b \\ & \} \end{array} \right.$$

Formas normales

Forma normal de Chomsky

Ejercicio (Obtención de la forma normal de Chomsky 1 / 2)

$$P = \{$$

- (1) $S \rightarrow T L ;$
- (2) $T \rightarrow \mathbf{int}$
- (3) $T \rightarrow \mathbf{float}$
- (4) $L \rightarrow \mathbf{identificador} L'$
- (5) $L' \rightarrow \mathbf{, identificador} L'$
- (6) $L' \rightarrow \epsilon$

$$\}$$

Nota

Previamente, hay que eliminar la regla ϵ

Formas normales

Forma normal de Chomsky

Ejercicio (Obtención de la forma normal de Chomsky 2 / 2)

$$P = \{$$

- (1) $S \rightarrow T \text{ identificador } (P) ;$
- (2) $T \rightarrow \text{int}$
- (3) $P \rightarrow \text{identificador } P'$
- (4) $P' \rightarrow , \text{ identificador } P'$
- (5) $P' \rightarrow \epsilon$

$$\}$$

Nota

Previamente, hay que eliminar la regla ϵ

Contenido de la sección

- 6 Formas normales
 - Forma normal de Chomsky
 - Forma normal de Greibach

Formas normales

Forma normal de Greibach

Definición (Forma normal de Greibach)

- Una gramática está en la *forma normal de Greibach (F.N.G.)* si sus reglas son de la forma:

$$A \longrightarrow a \alpha$$

donde

- $A \in V_N$
- $a \in V_T$
- $\alpha \in V_N^*$

Formas normales

Forma normal de Greibach

Ejemplo (Forma normal de Greibach: gramática

1 / 2)

$$P = \{$$

- (1) $S \rightarrow \mathbf{int} L P$
- (2) $S \rightarrow \mathbf{float} L P$
- (3) $L \rightarrow \mathbf{identificador} L'$
- (4) $L \rightarrow \mathbf{identificador}$
- (5) $L' \rightarrow , I L'$
- (6) $L' \rightarrow , I$
- (7) $I \rightarrow \mathbf{identificador}$
- (8) $P \rightarrow ;$

$$\}$$

Formas normales

Forma normal de Greibach

Ejemplo (Forma normal de Greibach: derivación

2 / 2)

$S \xRightarrow{1} \text{int } \underline{L} P$

$\xRightarrow{3} \text{int } \underline{\text{identificador}} L' P$

$\xRightarrow{6} \text{int } \text{identificador } , \underline{I} P$

$\xRightarrow{7} \text{int } \text{identificador } , \underline{\text{identificador}} P$

$\xRightarrow{8} \text{int } \text{identificador } , \text{identificador } \underline{;} ;$

Formas normales

Forma normal de Greibach

Nota (Análisis sintáctico descendente)

- Una gramática en *FNG* admite el *análisis sintáctico descendente* si $\forall A \in V_N$ sus alternativas comienzan por un símbolo terminal *diferente*.

$$A \longrightarrow a_1 \alpha_1$$

$$A \longrightarrow a_2 \alpha_2$$

...

$$A \longrightarrow a_N \alpha_N$$

Si $i \neq j$ entonces $a_i \neq a_j$

- Véase el tema nº 4.- *Análisis sintáctico descendente*.

Formas normales

Forma normal de Greibach

Ejemplo (Análisis sintáctico descendente)

Gramática en FNG que admite el análisis sintáctico descendente

$$P = \{$$

- (1) $S \rightarrow a A B C$
- (2) $A \rightarrow a A$
- (3) $A \rightarrow b$
- (4) $B \rightarrow b B$
- (5) $B \rightarrow c$
- (6) $C \rightarrow c C$
- (7) $C \rightarrow d$

$$\}$$

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 1 / 9)

- **Entrada**

- $G = (V_N, V_T, P, S)$

- Gramática en la forma normal Chomsky.*

- **Salida**

- $G' = (V'_N, V_T, P', S)$

- Gramática en la forma normal Greibach.*

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 2 / 9)

- *Paso 1*

*Aplicación del algoritmo que elimina la **recursividad general** por la izquierda.*

- *Paso 2*

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

- *Paso 3:*

Transformación de las reglas de los símbolos **no terminales obtenidos** al eliminar la **recursividad inmediata**.

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 3 / 9)

- *Paso 1 (1/3)*

*Aplicación del algoritmo que elimina la **recursividad general por la izquierda**.*

Las reglas de producción resultantes serán de la forma:

$$A_i \longrightarrow A_j \gamma \quad \forall j > i \wedge i, j \in \{1, 2, \dots, n\}$$

$$A_i \longrightarrow a \gamma$$

$$A'_i \longrightarrow \gamma$$

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 4 / 9)

- *Paso 1 (2/3)*

donde

- $A_i, A_j \in V_N$
- $a \in V_T$
- $\forall i A'_i$: *generado al eliminar la recursividad inmediata por la izquierda*
- $\gamma \in (V_N \cup \{A'_1, A'_2, \dots, A'_n\})^*$.

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 5 / 9)

- *Paso 1 (3/3)*
 - *En particular, la reglas del símbolo A_n ya estarán en la forma normal de Greibach.*

$$A_n \rightarrow a \gamma$$

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 6 / 9)

- *Paso 2. 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 $A_i \rightarrow A_j \gamma$ hacer

si $A_j \rightarrow a_1 \alpha_1 \mid a_2 \alpha_2 \mid \dots \mid a_p \alpha_p$ son las reglas actuales de A_j

entonces $A_i \rightarrow a_1 \alpha_1 \gamma \mid a_2 \alpha_2 \gamma \mid \dots \mid a_p \alpha_p \gamma$

pasan a ser producciones actuales de A_i

fin_si

fin_para

fin_para

fin_para

fin

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 7 / 9)

- *Paso 2 (continuación)*
 - *Las reglas de producción resultantes serán de la forma:*

$$A_i \longrightarrow a \gamma$$

$$A'_i \longrightarrow \gamma$$

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

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 8 / 9)

- *Paso 3. 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 regla actual de A'_i de la forma $A'_i \rightarrow A_j \gamma$

hacer si $A_j \rightarrow a_1 \alpha_1 \mid a_2 \alpha_2 \mid \dots \mid a_p \alpha_p$ son las reglas actuales de A_j

entonces $A'_i \rightarrow a_1 \alpha_1 \gamma \mid a_2 \alpha_2 \gamma \mid \dots \mid a_p \alpha_p \gamma$

pasan a ser reglas actuales de A'_i

fin si

fin para

fin para

fin para

fin

Formas normales

Forma normal de Greibach

Algoritmo (Obtención de la forma normal de Greibach 9 / 9)

- *Paso 3. (Continuación)*

Las reglas de producción resultantes serán de la forma:

$$A_i \longrightarrow a \gamma$$

$$A'_i \longrightarrow a \gamma$$

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

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 1 / 10)

- Gramática que está en la forma normal de *Chomsky*.

$$P = \{$$
$$(1) S \rightarrow A B$$
$$(2) A \rightarrow S B$$
$$(3) A \rightarrow a$$
$$(4) B \rightarrow B A$$
$$(5) B \rightarrow d$$
$$\}$$

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 2 / 10)

- *Paso 1. Eliminación de la recursividad general por la izquierda*

(1) *Procesamiento de la regla de S*

- S es el **primer** símbolo y, por tanto, su alternativa no comienza por un símbolo con un número de orden menor.
- S **no** tiene recursividad inmediata por la izquierda.
- La regla de S se añade a P_1

$$S \rightarrow A B$$

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 3 / 10)

- *Paso 1. Eliminación de la recursividad general por la izquierda*

(2) *Procesamiento de las reglas de A*

- *S aparece en la regla de A*

$$A \rightarrow S B$$

Se sustituye S por su alternativa y se genera nueva regla de A

$$A \rightarrow A B B$$

- *Eliminación de la recursividad **inmediata** por la izquierda de A.*

Las reglas actuales de A

$$A \rightarrow A B B \mid a$$

se sustituyen por

$$A \rightarrow a \mid a A'$$

$$A' \rightarrow B B \mid B B A'$$

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 4 / 10)

- *Paso 1. Eliminación de la recursividad general por la izquierda*
- (3) *Procesamiento de las reglas de B*
 - *No hay sustituir ningún símbolo en las reglas de B.*
 - *Eliminación de la recursividad **inmediata** por la izquierda de B.*

Las reglas actuales de B

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

se sustituyen por

$$B \longrightarrow d \mid d B'$$

$$B' \longrightarrow A \mid A B'$$

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 5 / 10)

- Paso 1. Eliminación de la recursividad general por la izquierda*
Gramática generada en el paso 1

$$P_1 = \left\{ \begin{array}{l} S \rightarrow A B \\ A \rightarrow a \mid a A' \\ B \rightarrow d \mid d B' \\ A' \rightarrow B B \mid B B B' \\ B' \rightarrow A \mid A B' \end{array} \right\}$$

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 6 / 10)

- *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.*

$$B \rightarrow d \mid d B'$$

- (2) *Las reglas de A ya están en la forma normal de Greibach.*

$$A \rightarrow a \mid a A'$$

- (3) *Transformación de las reglas de S*

Se sustituye A por sus alternativas en la regla

$$S \rightarrow A B$$

Se generan las siguientes reglas

$$S \rightarrow \underline{a} B \mid \underline{a} A' B$$

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 7 / 10)

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

Gramática generada en el paso 2

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

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 8 / 10)

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

- *Transformación de las reglas de A'*

$$A' \longrightarrow B B \mid B B A'$$

se sustituye B por sus alternativas y se generan las reglas:

$$A' \longrightarrow \underline{d} B \mid \underline{d} B A' \mid \underline{d} B' B \mid \underline{d} B' B A'$$

- *Transformación de las reglas de B'*

$$B' \longrightarrow A \mid A B'$$

se sustituye A por sus alternativas y se generan las reglas:

$$B' \longrightarrow \underline{a} \mid \underline{a} A' \mid \underline{a} B' \mid \underline{a} A' B'$$

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 9 / 10)

- Paso 3. Gramática generada

$$P_3 = \{$$

$$S \rightarrow a B \mid a A' B$$

$$A \rightarrow a \mid a A'$$

$$B \rightarrow d \mid d B'$$

$$A' \rightarrow d B \mid d B A' \mid d B' B \mid d B' B A'$$

$$B' \rightarrow a \mid a A' \mid a B' \mid a A' B'$$

$$\}$$

Formas normales

Forma normal de Greibach

Ejemplo (Obtención de la forma normal de Greibach 10 / 10)

$$\begin{aligned}
 P' = \{ & \\
 & S \rightarrow a B \mid a A' B \\
 & A \rightarrow a \mid a A' \\
 & B \rightarrow d \mid d B' \\
 & A' \rightarrow d B \mid d B A' \mid d B' B \mid d B' B A' \\
 & B' \rightarrow a \mid a A' \mid a B' \mid a A' B' \\
 & \}
 \end{aligned}$$

Formas normales

Forma normal de Greibach

Ejercicio (Obtención de la forma normal de Greibach)

$$P = \{$$

- (1) $S \rightarrow T \text{ identificador } (P) ;$
- (2) $T \rightarrow \text{int}$
- (3) $P \rightarrow \text{identificador } P'$
- (4) $P' \rightarrow , \text{ identificador } P'$
- (5) $P' \rightarrow \epsilon$

$$\}$$

Nota

Previamente, hay que convertirla a la forma normal de Chomsky.

Tipos de análisis sintáctico

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad
- 4 Operaciones de limpieza
- 5 Recursividad y factorización
- 6 Formas normales
- 7 Tipos de análisis sintáctico**
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos

Tipos de análisis sintáctico

- 7 Tipos de análisis sintáctico
 - Métodos universales
 - Métodos descendentes
 - Métodos ascendentes

Contenido de la sección

- 7 Tipos de análisis sintáctico
 - Métodos universales
 - Métodos descendentes
 - Métodos ascendentes

Tipos de análisis sintáctico

Métodos universales

Tipos de análisis sintáctico

- Métodos universales
 - Algoritmo de **CYK**: Cocke, Younger, Kasami.
 - Algoritmo de **Earley**.
 - Algoritmo de **GHR**: Graham, Harrison, Ruzzo.

Tipos de análisis sintáctico

Métodos universales

Tipos de análisis sintáctico

- Métodos universales
 - **Ventaja**
 - Se pueden aplicar a **todas** las gramáticas.
 - **Inconvenientes**
 - Complejidad computacional muy alta
 - En el peor de los casos, la complejidad es cúbica: $O(n^3)$
 - El algoritmo **CYK** requiere que la gramática se transforme previamente a la **FNC**.

Contenido de la sección

- 7 Tipos de análisis sintáctico
 - Métodos universales
 - Métodos descendentes
 - Métodos ascendentes

Tipos de análisis sintáctico

Métodos descendentes

Métodos descendentes

- Generan un **derivación** por la **izquierda** de la cadena de entrada
- El **árbol sintáctico** se genera de **arriba hacia abajo**:
 - desde la raíz hasta las hojas

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Gramática de las expresiones aritméticas)

- $$P = \{$$
- (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow T E'$
 - (3) $E' \rightarrow + T E'$
 - (4) $E' \rightarrow \epsilon$
 - (5) $T \rightarrow F T'$
 - (6) $T' \rightarrow * F T'$
 - (7) $T' \rightarrow \epsilon$
 - (8) $F \rightarrow (E)$
 - (9) $F \rightarrow \text{identificador}$
 - (10) $F \rightarrow \text{número}$
- $$\}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

1 / 2

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = \underline{T} E'$$

$$\xRightarrow{5} \text{identificador} = \underline{F T'} E'$$

$$\xRightarrow{9} \text{identificador} = \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} \underline{\in} E'$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} \underline{+ T E'}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

1 / 2

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = \underline{T} E'$$

$$\xRightarrow{5} \text{identificador} = \underline{F T'} E'$$

$$\xRightarrow{9} \text{identificador} = \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} \underline{\in} E'$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} \underline{+ T E'}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

1 / 2

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = T E'$$

$$\xRightarrow{5} \text{identificador} = F T' E'$$

$$\xRightarrow{9} \text{identificador} = \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} \in E'$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + T E'$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

1 / 2

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = \underline{T} E'$$

$$\xRightarrow{5} \text{identificador} = \underline{F T'} E'$$

$$\xRightarrow{9} \text{identificador} = \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} \underline{\in} E'$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} \underline{+ T E'}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

1 / 2

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = T E'$$

$$\xRightarrow{5} \text{identificador} = F T' E'$$

$$\xRightarrow{9} \text{identificador} = \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} \in E'$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + T E'$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

1 / 2

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = T E'$$

$$\xRightarrow{5} \text{identificador} = F T' E'$$

$$\xRightarrow{9} \text{identificador} = \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} \in E'$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + T E'$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

1 / 2

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = T E'$$

$$\xRightarrow{5} \text{identificador} = F T' E'$$

$$\xRightarrow{9} \text{identificador} = \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} \in E'$$

$$\xRightarrow{3} \text{identificador} = \text{identificador} + T E'$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

2 / 2

$$\begin{aligned}
 &\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{F T' E'} \\
 &\xRightarrow{10} \text{identificador} = \text{identificador} + \underline{\text{número}} T' E' \\
 &\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{F T' E'} \\
 &\xRightarrow{9} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}} T' E' \\
 &\xRightarrow{7} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\epsilon} E' \\
 &\xRightarrow{4} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\epsilon} \\
 &= \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}
 \end{aligned}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

2 / 2

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{F T' E'}$$

$$\xRightarrow{10} \text{identificador} = \text{identificador} + \underline{\text{número}} T' E'$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{F T' E'}$$

$$\xRightarrow{9} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in} E'$$

$$\xRightarrow{4} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in}$$

$$= \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

2 / 2

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{F T' E'}$$

$$\xRightarrow{10} \text{identificador} = \text{identificador} + \underline{\text{número}} T' E'$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{F T' E'}$$

$$\xRightarrow{9} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\epsilon} E'$$

$$\xRightarrow{4} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\epsilon}$$

$$= \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

2 / 2

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{F T' E'}$$

$$\xRightarrow{10} \text{identificador} = \text{identificador} + \underline{\text{número}} T' E'$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{F T' E'}$$

$$\xRightarrow{9} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in E'}$$

$$\xRightarrow{4} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in}$$

$$= \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

2 / 2

$$\xRightarrow{5} \text{identificador} = \text{identificador} + \underline{F T' E'}$$

$$\xRightarrow{10} \text{identificador} = \text{identificador} + \underline{\text{número}} T' E'$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \underline{F T' E'}$$

$$\xRightarrow{9} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}} T' E'$$

$$\xRightarrow{7} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in} E'$$

$$\xRightarrow{4} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in}$$

$$= \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

2 / 2

$$\xRightarrow[5]{} \text{identificador} = \text{identificador} + \underline{F T' E'}$$

$$\xRightarrow[10]{} \text{identificador} = \text{identificador} + \underline{\text{número}} T' E'$$

$$\xRightarrow[6]{} \text{identificador} = \text{identificador} + \text{número} * \underline{F T' E'}$$

$$\xRightarrow[9]{} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}} T' E'$$

$$\xRightarrow[7]{} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in E'}$$

$$\xRightarrow[4]{} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in}$$

$$= \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda)

2 / 2

$$\xRightarrow[5]{} \text{identificador} = \text{identificador} + \underline{F T' E'}$$

$$\xRightarrow[10]{} \text{identificador} = \text{identificador} + \underline{\text{número}} T' E'$$

$$\xRightarrow[6]{} \text{identificador} = \text{identificador} + \text{número} * \underline{F T' E'}$$

$$\xRightarrow[9]{} \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}} T' E'$$

$$\xRightarrow[7]{} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in E'}$$

$$\xRightarrow[4]{} \text{identificador} = \text{identificador} + \text{número} * \text{identificador} \underline{\in}$$

$$= \text{identificador} = \text{identificador} + \text{número} * \underline{\text{identificador}}$$

Tipos de análisis sintáctico

Métodos descendentes

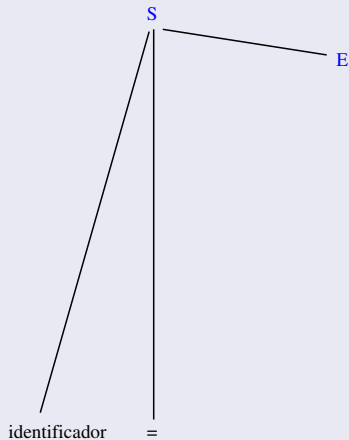
Ejemplo (Derivación por la izquierda: árbol sintáctico 1 / 13)

S

Tipos de análisis sintáctico

Métodos descendentes

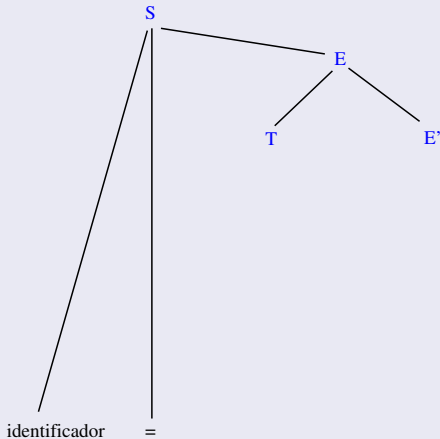
Ejemplo (Derivación por la izquierda: árbol sintáctico 2 / 13)



Tipos de análisis sintáctico

Métodos descendentes

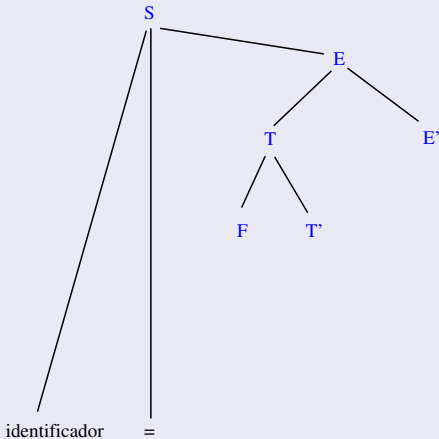
Ejemplo (Derivación por la izquierda: árbol sintáctico 3 / 13)



Tipos de análisis sintáctico

Métodos descendentes

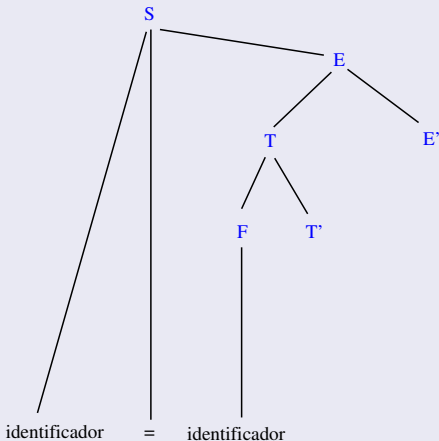
Ejemplo (Derivación por la izquierda: árbol sintáctico 4 / 13)



Tipos de análisis sintáctico

Métodos descendentes

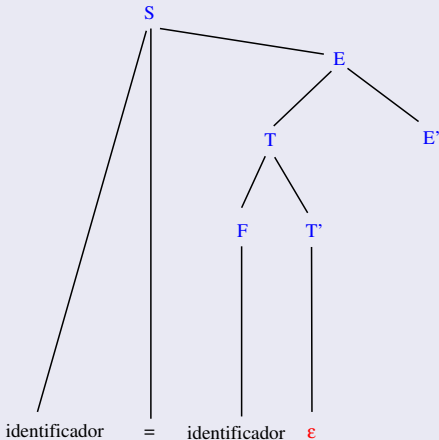
Ejemplo (Derivación por la izquierda: árbol sintáctico 5 / 13)



Tipos de análisis sintáctico

Métodos descendentes

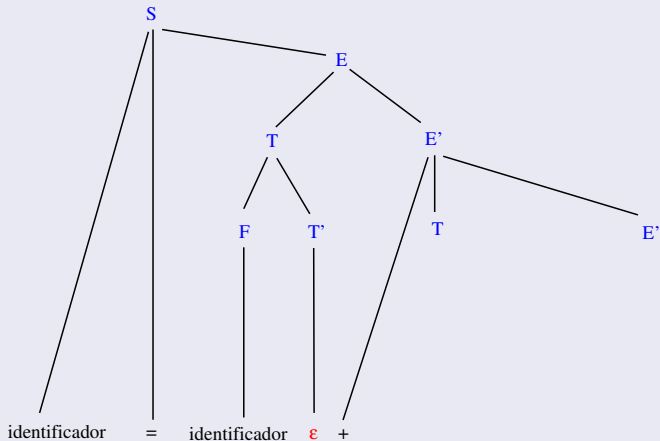
Ejemplo (Derivación por la izquierda: árbol sintáctico 6 / 13)



Tipos de análisis sintáctico

Métodos descendentes

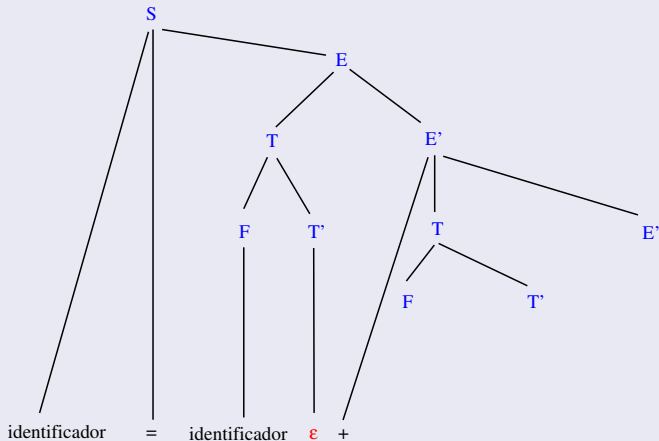
Ejemplo (Derivación por la izquierda: árbol sintáctico 7 / 13)



Tipos de análisis sintáctico

Métodos descendentes

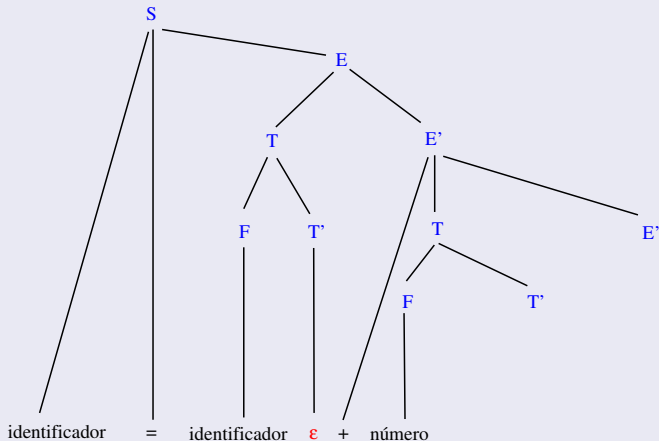
Ejemplo (Derivación por la izquierda: árbol sintáctico 8 / 13)



Tipos de análisis sintáctico

Métodos descendentes

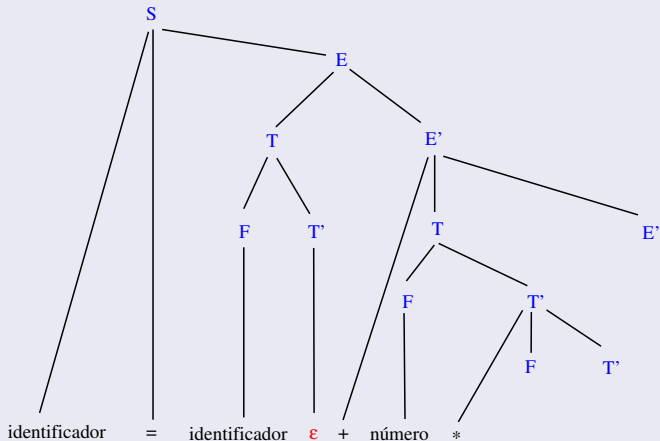
Ejemplo (Derivación por la izquierda: árbol sintáctico 9 / 13)



Tipos de análisis sintáctico

Métodos descendentes

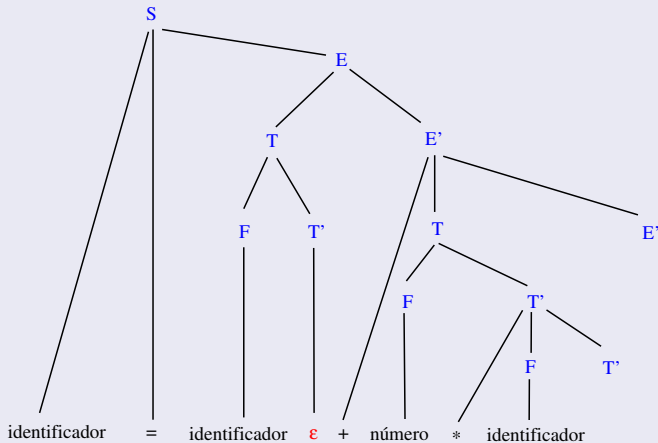
Ejemplo (Derivación por la izquierda: árbol sintáctico 10 / 13)



Tipos de análisis sintáctico

Métodos descendentes

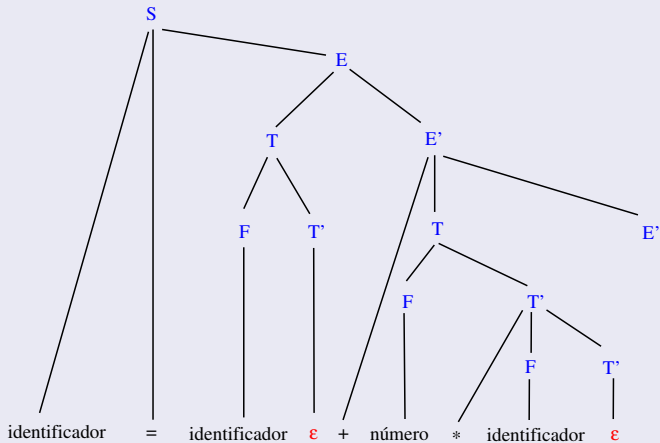
Ejemplo (Derivación por la izquierda: árbol sintáctico 11 / 13)



Tipos de análisis sintáctico

Métodos descendentes

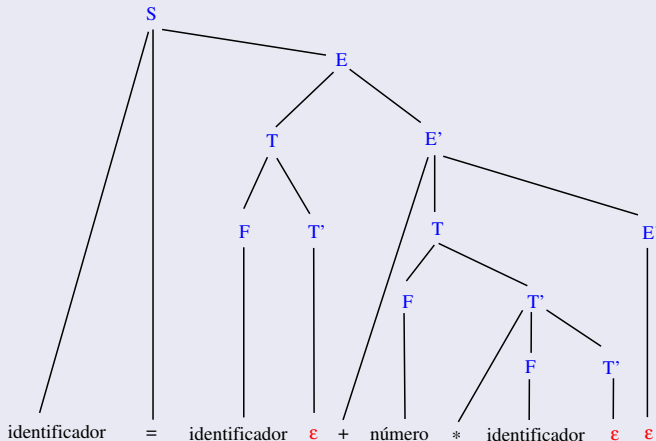
Ejemplo (Derivación por la izquierda: árbol sintáctico 12 / 13)



Tipos de análisis sintáctico

Métodos descendentes

Ejemplo (Derivación por la izquierda: árbol sintáctico 13 / 13)



Tipos de análisis sintáctico

Métodos descendentes

Métodos descendentes

- **Ventaja**
 - Los métodos de descenso **predictivo** tienen una **complejidad computacional** lineal: $O(n)$

- **Inconvenientes**

Los métodos descendentes

- No se puede aplicar a gramáticas con **recursividad por la izquierda**.
- No se puede aplicar a gramáticas **no** factorizadas por la izquierda.
- Estas condiciones son **necesarias** pero **no suficientes**:
 - Hay gramáticas **sin** recursividad por la izquierda y factorizadas por la izquierda que **no admiten** un análisis sintáctico descendente.

Tipos de análisis sintáctico

Métodos descendentes

Nota (Métodos descendentes)

- *El tema nº 4 explica las características de los métodos de análisis sintáctico descendente.*

Contenido de la sección

- 7 Tipos de análisis sintáctico
 - Métodos universales
 - Métodos descendentes
 - Métodos ascendentes

Tipos de análisis sintáctico

Métodos ascendentes

Métodos descendentes

- Generan una derivación por la **derecha** de la cadena de entrada, pero dicha derivación se genera en **orden inverso**
- El **árbol sintáctico** se genera de **abajo hacia arriba**:
 - desde las hojas hasta la raíz

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha: gramática)

- $P = \{$
 - (1) $S \rightarrow \text{identificador} = E$
 - (2) $E \rightarrow E + E$
 - (3) $E \rightarrow E * E$
 - (4) $E \rightarrow (E)$
 - (5) $E \rightarrow \text{número}$
 - (6) $E \rightarrow \text{identificador}$ $\}$

Nota

Esta gramática es *recursiva por la izquierda* y *no* está factorizada por la izquierda.

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha en orden inverso)

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{3} \text{identificador} = E + \underline{E * E}$$

$$\xRightarrow{6} \text{identificador} = E + E * \underline{\text{identificador}}$$

$$\xRightarrow{5} \text{identificador} = E + \underline{\text{número}} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha en orden inverso)

$$S \xRightarrow{1} \text{identificador} = E$$

$$\xRightarrow{2} \text{identificador} = E + E$$

$$\xRightarrow{3} \text{identificador} = E + E * E$$

$$\xRightarrow{6} \text{identificador} = E + E * \text{identificador}$$

$$\xRightarrow{5} \text{identificador} = E + \text{número} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \text{identificador}$$

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha en orden inverso)

$$S \xRightarrow{1} \text{identificador} = E$$

$$\xRightarrow{2} \text{identificador} = E + E$$

$$\xRightarrow{3} \text{identificador} = E + E * E$$

$$\xRightarrow{6} \text{identificador} = E + E * \text{identificador}$$

$$\xRightarrow{5} \text{identificador} = E + \text{número} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \text{identificador}$$

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha en orden inverso)

$$S \xRightarrow{1} \underline{\text{identificador}} = E$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{3} \text{identificador} = E + \underline{E * E}$$

$$\xRightarrow{6} \text{identificador} = E + E * \underline{\text{identificador}}$$

$$\xRightarrow{5} \text{identificador} = E + \underline{\text{número}} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha en orden inverso)

$$S \xRightarrow{1} \text{identificador} = E$$

$$\xRightarrow{2} \text{identificador} = E + E$$

$$\xRightarrow{3} \text{identificador} = E + E * E$$

$$\xRightarrow{6} \text{identificador} = E + E * \text{identificador}$$

$$\xRightarrow{5} \text{identificador} = E + \text{número} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \text{identificador} + \text{número} * \text{identificador}$$

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha en orden inverso)

$$S \xRightarrow{1} \text{identificador} = \underline{E}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{3} \text{identificador} = E + \underline{E * E}$$

$$\xRightarrow{6} \text{identificador} = E + E * \underline{\text{identificador}}$$

$$\xRightarrow{5} \text{identificador} = E + \underline{\text{número}} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha en orden inverso)

$$S \xRightarrow{1} \text{identificador} = \underline{E}$$

$$\xRightarrow{2} \text{identificador} = \underline{E + E}$$

$$\xRightarrow{3} \text{identificador} = E + \underline{E * E}$$

$$\xRightarrow{6} \text{identificador} = E + E * \underline{\text{identificador}}$$

$$\xRightarrow{5} \text{identificador} = E + \underline{\text{número}} * \text{identificador}$$

$$\xRightarrow{6} \text{identificador} = \underline{\text{identificador}} + \text{número} * \text{identificador}$$

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha: árbol sintáctico 1 / 7)

identificador = identificador + número * identificador

Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha: árbol sintáctico 2 / 7)

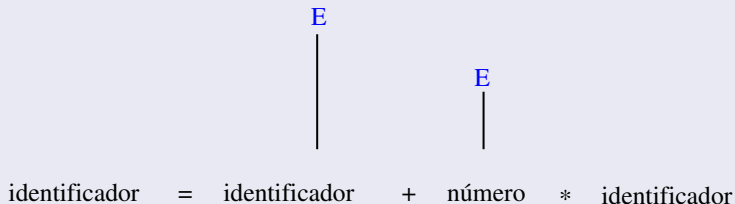
E

identificador = identificador + número * identificador

Tipos de análisis sintáctico

Métodos ascendentes

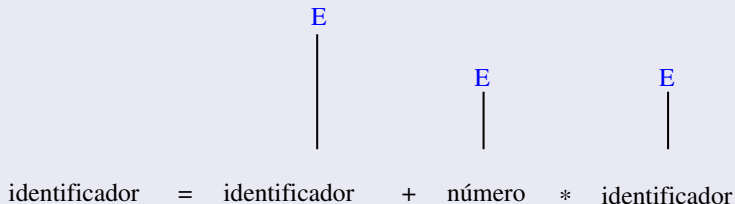
Ejemplo (Derivación por la derecha: árbol sintáctico 3 / 7)



Tipos de análisis sintáctico

Métodos ascendentes

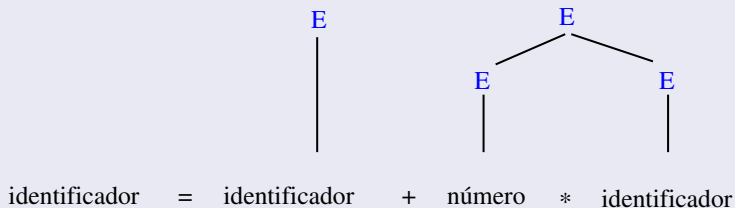
Ejemplo (Derivación por la derecha: árbol sintáctico 4 / 7)



Tipos de análisis sintáctico

Métodos ascendentes

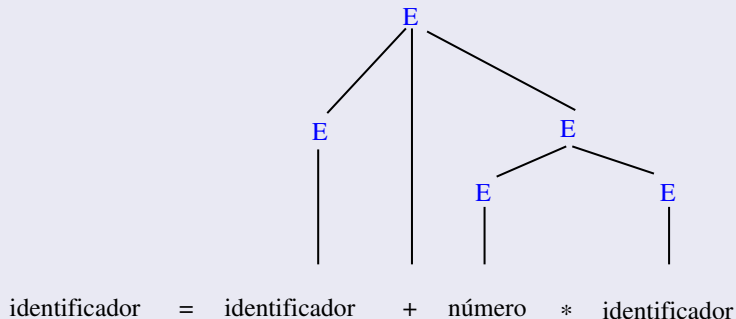
Ejemplo (Derivación por la derecha: árbol sintáctico 5 / 7)



Tipos de análisis sintáctico

Métodos ascendentes

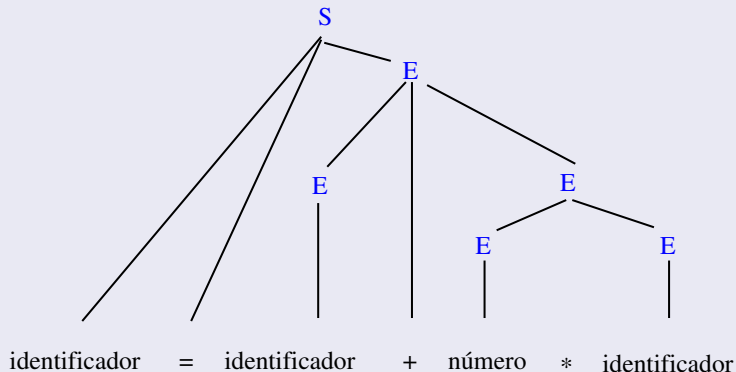
Ejemplo (Derivación por la derecha: árbol sintáctico 6 / 7)



Tipos de análisis sintáctico

Métodos ascendentes

Ejemplo (Derivación por la derecha: árbol sintáctico 7 / 7)



Tipos de análisis sintáctico

Métodos ascendentes

Métodos ascendentes

- **Ventajas**
 - Su **complejidad computacional** es lineal: $O(n)$
 - Son **más potentes** que los métodos descendentes
 - Los métodos ascendentes **se pueden** aplicar a gramáticas con recursividad por la izquierda o no factorizadas por la izquierda.
- **Inconveniente**
 - Algunas gramáticas **no admiten** un análisis sintáctico ascendente.

Tipos de análisis sintáctico

Métodos ascendentes

Nota (Métodos ascendentes)

- *El tema nº 5 explica las características de los métodos de análisis sintáctico ascendente.*

Detección y tratamiento de errores sintácticos

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad
- 4 Operaciones de limpieza
- 5 Recursividad y factorización
- 6 Formas normales
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos**
- 9 Generadores de analizadores sintácticos

Detección y tratamiento de errores sintácticos

- 8 Detección y tratamiento de errores sintácticos
 - Detección de errores sintácticos
 - Tratamiento de los errores sintácticos

Contenido de la sección

- 8 Detección y tratamiento de errores sintácticos
 - Detección de errores sintácticos
 - Tratamiento de los errores sintácticos

Detección y tratamiento de errores sintácticos

Detección de errores sintácticos

Detección de errores sintácticos

- Se origina al procesar un **componente léxico correcto** pero **inesperado**.
- Se **infringe** alguna regla sintáctica de lenguaje de programación.

Detección y tratamiento de errores sintácticos

Detección de errores sintácticos

Ejemplo (Detección de errores sintácticos)

1 / 3

- *Palabra reservada **mal** escrita, pero es un identificador correcto*

fi $(x \geq 0)$ $y = \text{sqrt}(x);$

- *Ubicación errónea de palabras reservadas*

$(x \geq 0)$ **if** $y = \text{sqrt}(x);$

- *Repetición de palabras reservadas*

if if $(x \geq 0)$ $y = \text{sqrt}(x);$

- *Omisión de una palabra reservada*

_ $(x \geq 0)$ $y = \text{sqrt}(x);$

- *Palabra reservada **inesperada***

if for $(x \geq 0)$ $y = \text{sqrt}(x);$

Detección y tratamiento de errores sintácticos

Detección de errores sintácticos

Ejemplo (Errores sintácticos)

2 / 3

- *Uso de una palabra reservada de otro lenguaje (Pascal)*
`if (x ≥ 0) then y = sqrt(x);`
- *Paréntesis no emparejados o balanceados*
`a = (2 * 3) + 5) ;`
`a = (2 * 3 + 5 ;`
- *Ausencia de operador*
`a = (2 _ 3) + 5 ;`
- *Ausencia de argumento*
`a = (2 * _) + 5 ;`

Detección y tratamiento de errores sintácticos

Detección de errores sintácticos

Ejemplo (Errores sintácticos)

3 / 3

- *Operador de asignación de otro lenguaje (Pascal)*
 $a := (2 * 3) + 5 ;$
- **Omisión** de *punto y coma de fin de sentencia*
 $a = (2 * 3) + 5 _$
- **Omisión** de *separadores (comas)*
 $a = atan2(x _ y)$
- **Omisión** de *argumento*
 $a = atan2(_ , y)$
- *Etc.*

Detección y tratamiento de errores sintácticos

Detección de errores sintácticos

Ejemplo (Error sintáctico no detectado)

```
while (n > 1) ;  
  {  
    factorial = factorial * n;  
    n = n - 1;  
  }
```

Contenido de la sección

- 8 Detección y tratamiento de errores sintácticos
 - Detección de errores sintácticos
 - Tratamiento de los errores sintácticos

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Criterios **generales** para el tratamiento de los errores

- Informar del error
 - + Localización: ubicación del error dentro del código.
 - + Descripción: motivo o características del error
- Evitar la cascada de errores
 - + Informar una sola vez de cada error.
 - + Si un error es producido por otro entonces sólo se debe informar del primero.
- Reparar el error, si es posible, e informar de la corrección realizada.
- Continuar con el proceso de traducción para detectar otros posibles errores.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Criterios **generales** para el tratamiento de los errores

- Informar del error
 - + Localización: ubicación del error dentro del código.
 - + Descripción: motivo o características del error
- Evitar la cascada de errores
 - + Informar una sola vez de cada error.
 - + Si un error es producido por otro entonces sólo se debe informar del primero.
- Reparar el error, si es posible, e informar de la corrección realizada.
- Continuar con el proceso de traducción para detectar otros posibles errores.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Criterios **generales** para el tratamiento de los errores

- Informar del error
 - + Localización: ubicación del error dentro del código.
 - + Descripción: motivo o características del error
- Evitar la cascada de errores
 - + Informar una sola vez de cada error.
 - + Si un error es producido por otro entonces sólo se debe informar del primero.
- Reparar el error, si es posible, e informar de la corrección realizada.
- Continuar con el proceso de traducción para detectar otros posibles errores.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Criterios **generales** para el tratamiento de los errores

- Informar del error
 - + Localización: ubicación del error dentro del código.
 - + Descripción: motivo o características del error
- Evitar la cascada de errores
 - + Informar una sola vez de cada error.
 - + Si un error es producido por otro entonces sólo se debe informar del primero.
- Reparar el error, si es posible, e informar de la corrección realizada.
- Continuar con el proceso de traducción para detectar otros posibles errores.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Nota (Reparación del error)

- **Siempre** *debe ser revisada después por el programador.*
- Sólo **propone** *una solución, que no tiene por qué ser la correcta.*
- Sólo *pretende que el proceso de traducción* **continúe** ... *para detectar más errores.*

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Métodos de recuperación de errores sintácticos

- (a) Modo de pánico
- (b) Método de nivel de frase
- (c) Reglas de producción de control de errores
- (d) Corrección global

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Métodos de recuperación de errores sintácticos

- (a) Modo de pánico
- (b) Método de nivel de frase
- (c) Reglas de producción de control de errores
- (d) Corrección global

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Modo de pánico

- Al detectar un **error**, se **eliminan** componentes léxicos de la entrada hasta que se encuentra un **componente léxico** de **sincronización**.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Ejemplo (Componente léxico de sincronización)

- *Palabra clave que inicie una sentencia: **if, while, for, etc.***
- *Fin de sentencia: ;*
- *Etc.*

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Modo de pánico

- **Ventajas**
 - Sencillo de aplicar
 - No cae en bucles infinitos
 - Permite continuar el análisis para detectar más errores.
- **Inconvenientes**
 - Método de aplicación local
 - No es capaz de detectar todos los errores posibles cuando busca el componente léxico de sincronización.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Métodos de recuperación de errores sintácticos

- (a) Modo de pánico
- (b) Método de nivel de frase
- (c) Reglas de producción de control de errores
- (d) Corrección global

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Método de nivel de frase

- Al detectar un **error**,
intenta realizar una transformación simple que permita corregir o reparar el error y continuar el análisis.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Método de nivel de frase

- Tipos de transformaciones:
 - + **Eliminar** un componente léxico
 - + **Insertar** un componente léxico
 - + **Sustituir** un componente léxico

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Nota (Método de nivel de frase)

- *Se debe informar de la transformación realizada.*
- *La transformación no pretende corregir el error, sino continuar con el análisis sintáctico.*
- *Posteriormente, el programador deberá supervisar la transformación realizada.*

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Método de nivel de frase

- **Ventajas**

- Puede hacer correcciones de errores habituales conocidos a priori.
- Permite continuar el análisis para detectar más errores.

- **Inconvenientes**

- Requiere un buen conocimiento del lenguaje para predecir los errores que se pueden producir.
- Método de aplicación local.
- Puede provocar bucles infinitos
- Difícil de aplicar cuando el error se encuentra muy lejos del punto de detección.

Por ejemplo: llave final } no balanceada

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Nota (Método de nivel de frase)

- *Es uno de los métodos más utilizados para la recuperación de errores sintácticos*

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Métodos de recuperación de errores sintácticos

- (a) Modo de pánico
- (b) Método de nivel de frase
- (c) Reglas de producción de control de errores
- (d) Corrección global

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Reglas de producción de control de errores

- Se amplía la gramática con **nuevas reglas de producción** que **modelan** posibles situaciones de **error**.
- Si se **utilizan** dichas reglas de producción durante el análisis sintáctico entonces **se activa una función de recuperación de error**.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Ejemplo (Regla de control de error)

- $P = \{$
 - (1) $S \rightarrow \text{identificador} = E$
 - (1') $S \rightarrow \text{constante} = E$
 - (2) $E \rightarrow E + E$
 - (3) $E \rightarrow E * E$
 - (4) $E \rightarrow (E)$
 - (5) $E \rightarrow \text{número}$
 - (6) $E \rightarrow \text{identificador}$
 - (7) $E \rightarrow \text{constante}$ $\}$

Nota

- Detecta una asignación a una constante

$PI = 999999;$

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Reglas de producción de control de errores

- **Ventajas**
 - Pueden corregir errores habituales que son conocidos a priori.
 - Permite continuar el análisis sintáctico para detectar más errores.
- **Inconvenientes**
 - Requiere un buen conocimiento del lenguaje para predecir los errores que se pueden producir.
 - Método de aplicación local.
 - Al introducir nuevas reglas, puede que la gramática no admita el análisis sintáctico por la generación de **conflictos**.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Métodos de recuperación de errores sintácticos

- (a) Modo de pánico
- (b) Método de nivel de frase
- (c) Reglas de producción de control de errores
- (d) Corrección global

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Corrección global

- Es un método **teórico** basado en los métodos anteriores y, fundamentalmente, en el método de nivel de frase.
- Dado un programa con **errores**, intenta realizar el **menor número de transformaciones** para obtener otro programa **correcto**.

Detección y tratamiento de errores sintácticos

Tratamiento de los errores sintácticos

Reglas de producción de error

- **Ventajas**
 - Proporciona una escala para evaluar las técnicas de recuperación de errores.
- **Inconvenientes**
 - La complejidad temporal y espacial es muy alta.
 - El programa **sintácticamente correcto** que se genera puede ser **semánticamente diferente** del programa original.

Generadores de analizadores sintácticos

- 1 Introducción
- 2 Gramáticas de contexto libre
- 3 Ambigüedad
- 4 Operaciones de limpieza
- 5 Recursividad y factorización
- 6 Formas normales
- 7 Tipos de análisis sintáctico
- 8 Detección y tratamiento de errores sintácticos
- 9 Generadores de analizadores sintácticos**

Generadores de analizadores sintácticos

9 Generadores de analizadores sintácticos

Generadores de analizadores sintácticos

Definición (Generador de analizador sintáctico)

- **Entrada:** *gramática de contexto libre.*
- **Salida:** *analizador sintáctico.*

Generadores de analizadores sintácticos

Tipos de generadores

- Generadores de analizadores sintácticos **descendentes**
- Generadores de analizadores sintácticos **ascendentes**

Generadores de analizadores sintácticos

Tipos de generadores

- Generadores de analizadores sintácticos **descendentes**
- Generadores de analizadores sintácticos **ascendentes**

Generadores de analizadores sintácticos

Generadores de analizadores sintácticos descendentes 1 / 2

- ANTLR:
 - **Nombre:** *Another Tool for Language Recognition*
 - **Tipo:** LL(K)
 - **Lenguaje:** Java y C++
 - **Web:** <http://www.antlr.org/>
- JavaCC
 - **Nombre:** *Java Compiler Compiler*
 - **Tipo:** LL(K)
 - **Lenguaje:** Java
 - **Web:** <https://javacc.dev.java.net/>

Generadores de analizadores sintácticos

Generadores de analizadores sintácticos descendentes 2 / 2

- LLGen:
 - **Nombre:**
 - **Tipo:** LL(K), descenso recursivo
 - **Lenguaje:** Java y C++
 - **Web:** <http://www.cs.vu.nl/~ceriel/LLgen.html>
- COCO/R
 - **Nombre:** *Java Compiler Compiler*
 - **Tipo:** LL(K), descenso recursivo
 - **Lenguaje:** C, C++, C#, Java
 - **Web:** <http://www.ssw.uni-linz.ac.at/Research/Projects/Coco/>

Generadores de analizadores sintácticos

Tipos de generadores

- Generadores de analizadores sintácticos **descendentes**
- Generadores de analizadores sintácticos **ascendentes**

Generadores de analizadores sintácticos

Generadores de analizadores sintácticos ascendentes

1 / 4

- YACC:
 - **Nombre:** *Yet Another Compiler – Compiler*
 - **Tipo:** LALR(1)
 - **Lenguaje:** C
 - **Web:** <http://dinosaur.compilertools.net/>
- Bison
 - **Nombre:** *GNU versión libre de YACC*
 - **Tipo:** LALR(1)
 - **Lenguaje:** C
 - **Web:** <http://www.gnu.org/software/bison/>

Generadores de analizadores sintácticos

Generadores de analizadores sintácticos ascendentes

2 / 4

- **AYACC:**
 - **Nombre:** *Yet Another Compiler – Compiler*
 - **Tipo:** LALR(1)
 - **Lenguaje:** Ada
 - **Web:** [www.ics.uci.edu/~ arcadia/Aflex-Ayacc/aflex-ayacc.html](http://www.ics.uci.edu/~arcadia/Aflex-Ayacc/aflex-ayacc.html)
- **BYACC**
 - **Nombre:** *Berkeley YACC*
 - **Tipo:** LALR(1)
 - **Lenguaje:** C
 - **Web:** <ftp://ftp.cs.berkeley.edu/ucb/4bsd/byacc.tar.Z>

Generadores de analizadores sintácticos

Generadores de analizadores sintácticos ascendentes

3 / 4

- PCYACC:

- **Nombre:** *Yet Another Compiler – Compiler* para MSDOS
- **Tipo:** LALR(1)
- **Lenguaje:** C, C++, Java, Delphi,...
- **Web:** <http://www.abxsoft.com/pcyacc.htm>

- LEMON

- **Nombre:** *Berkeley YACC*
- **Tipo:** LALR(1)
- **Lenguaje:** C
- **Web:** <http://www.hwaci.com/sw/lemon/>

Generadores de analizadores sintácticos

Generadores de analizadores sintácticos ascendentes

4 / 4

- CUP:
 - **Nombre:** *constructor of Useful Parser*
 - **Tipo:** LALR(1)
 - **Lenguaje:** Java
 - **Web:** <http://www2.cs.tum.edu/projects/cup/>

PROCESADORES DE LENGUAJES

TEMA III.- FUNDAMENTOS TEÓRICOS DEL ANÁLISIS SINTÁCTICO

Prof. Dr. Nicolás Luis Fernández García

Departamento de Informática y Análisis Numérico
Escuela Politécnica Superior
Universidad de Córdoba