

### 3. Análisis Sintáctico

#### 3.1. Conceptos fundamentales

- Especificación sintáctica de los Lenguajes de Programación
- Introducción al Análisis Sintáctico

#### 3.2. Análisis Sintáctico Descendente

- Condición LL(1): Gramáticas LL(1)
- Construcción de Analizadores Sintácticos LL(1)

#### 3.3. Análisis Sintáctico Ascendente

- Conceptos fundamentales: ASA por “Desplazamiento-Reducción”
- Gramáticas LR(0) y Gramáticas SLR(1)
- Relación entre gramáticas y resolución de conflictos
- Tratamiento y recuperación de errores

#### ➤ Gramáticas incontextuales: $G = (N, \Sigma, P, S)$

$$S \in N; \quad V = N \cup \Sigma; \quad N \cap \Sigma = \emptyset; \quad (A \rightarrow \alpha) \in P; \quad A \in N; \quad \alpha \in V^*$$

#### ➤ Derivación directa

$$\delta A \gamma \Rightarrow \delta \alpha \gamma \quad \text{sii} \quad \exists (A \rightarrow \alpha) \in P; \quad \delta, \gamma \in V^*$$

#### ➤ Derivación

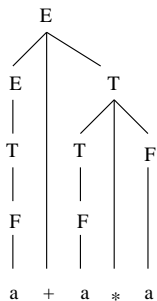
$$\alpha \xRightarrow{*} \beta \quad \text{sii} \quad \exists \alpha_0, \dots, \alpha_m \in V^* : \quad \alpha_0 = \alpha \Rightarrow \alpha_1 \Rightarrow \dots \Rightarrow \alpha_{m-1} \Rightarrow \alpha_m = \beta$$

#### ➤ Forma Sentencial: $\alpha \in V^* \quad \text{sii} \quad \exists S \xRightarrow{*} \alpha$

#### ➤ Lenguaje Generado: $L(G) = \{x \mid x \in \Sigma^* : S \xRightarrow{+} x\}$

### Ejemplo-1

- 1)  $E ::= E + T$
- 2)  $E ::= T$
- 3)  $T ::= T * F$
- 4)  $T ::= F$
- 5)  $F ::= ( E )$
- 6)  $F ::= a$



#### Secuencia de Derivación a Izquierdas

$$\begin{aligned}
 E &\Rightarrow E + T \Rightarrow T + T \Rightarrow F + T \Rightarrow \\
 &a + T \Rightarrow a + T * F \Rightarrow \\
 &a + F * F \Rightarrow a + a * F \Rightarrow \\
 &a + a * a
 \end{aligned}$$

#### Secuencia de Derivación a Derechas

$$\begin{aligned}
 E &\Rightarrow E + T \Rightarrow E + T * F \Rightarrow \\
 &E + T * a \Rightarrow E + F * a \Rightarrow \\
 &E + a * a \Rightarrow T + a * a \Rightarrow \\
 &F + a * a \Rightarrow a + a * a
 \end{aligned}$$

#### ➤ Árbol de Derivación. Dado $G = (N, \Sigma, P, S)$ :

- la raíz está etiquetada con el símbolo inicial  $S$ ,
- cada hoja está etiquetada con un símbolo de  $\Sigma \cup \{\epsilon\}$ ,
- cada nodo interior está etiquetado con un símbolo de  $N$ ,
- sea  $A \in N$  la etiqueta de un nodo, y  $x_1, \dots, x_n$  las etiquetas de sus nodos hijos (de izquierda a derecha); entonces:  $A \rightarrow x_1 \dots x_n \in P$ .

#### ➤ Teorema

Dada  $G(S)$ ,  $S \xRightarrow{*} \alpha$  (con  $\alpha \in V^*$ ) **sii** existe un árbol de derivación que produce  $\alpha$ .

#### ➤ Gramática ambigua

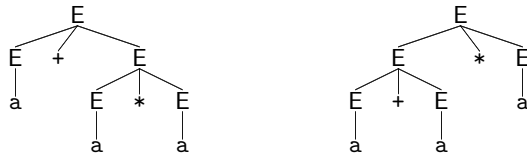
Una gramática es ambigua **sii** dado  $x \in \Sigma^*$  existen más de un árbol de derivación que produce  $x$ .

#### ➤ Proposición

Determinar si una gramática es ambigua, es un problema indecidible.

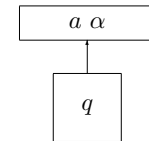
## Ejemplo-2

$$\begin{array}{l} E \rightarrow E + E \\ E \rightarrow E * E \\ E \rightarrow ( E ) \\ E \rightarrow a \end{array}$$



## Máquinas aceptoras: Autómatas

## &gt; Autómata de Estados Finitos



$$AEF = (Q, \Sigma, \delta, q_0, F)$$

$$q_0 \in Q; \quad F \subseteq Q; \quad \delta : Q \times \Sigma \rightarrow \wp(Q)$$

$$\text{Configuración} \Rightarrow (q, a\alpha)$$

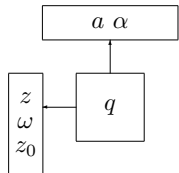
$$> \text{Movimiento:} \quad (q, a\alpha) \vdash (q', \alpha) \quad \text{sii} \quad \exists q' \in \delta(q, a)$$

$$\text{con:} \quad q, q' \in Q; \quad a \in \Sigma; \quad \alpha \in \Sigma^*$$

## &gt; Lenguaje aceptado por un AEF

$$L(AEF) = \{x \mid x \in \Sigma^* : (q_0, x) \vdash^* (q, \epsilon) \quad q \in F\}.$$

## &gt; Autómata a Pila



$$A_p = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$$

$$q_0 \in Q; \quad z_0 \in \Gamma; \quad F \subseteq Q;$$

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow \wp(Q \times \Gamma^*)$$

$$\text{Configuración} \Rightarrow (q, a\alpha, z\omega z_0)$$

$$> \text{Movimiento:} \quad (q, a\alpha, z\omega z_0) \vdash (q', \alpha, \beta\omega z_0) \quad \text{sii} \quad \exists (q', \beta) \in \delta(q, a, z)$$

$$\text{con:} \quad q, q' \in Q; \quad a \in \Sigma \cup \{\epsilon\}; \quad \alpha \in \Sigma^*; \quad z \in \Gamma; \quad \beta, \omega \in \Gamma^*$$

## &gt; Lenguaje aceptado a estado final:

$$L_F(A_p) = \{x \mid x \in \Sigma^* : (q_0, x, z_0) \vdash^* (q, \epsilon, \gamma); \quad q \in F\}$$

## &gt; Lenguaje aceptado a pila vacía

$$L_v(A_p) = \{x \mid x \in \Sigma^* : (q_0, x, z_0) \vdash^* (q, \epsilon, \epsilon)\}$$

## Aproximaciones al Análisis Sintáctico

## &gt; Análisis Sintáctico Descendente

Construye el árbol de análisis desde la raíz hasta las hojas

$\Rightarrow$  *Secuencia de Derivación a Izquierda*

## &gt; Análisis Sintáctico Ascendente

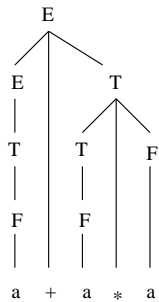
Construye el árbol de análisis desde las hojas hasta la raíz

$\Rightarrow$  *Inversa de la Secuencia de Derivación a Derecha*

## INTRODUCCIÓN AL AS

### Ejemplo-3 (ASD)

- 1)  $E ::= E + T$
- 2)  $E ::= T$
- 3)  $T ::= T * F$
- 4)  $T ::= F$
- 5)  $F ::= ( E )$
- 6)  $F ::= a$



### Secuencia de Derivación a Izquierdas

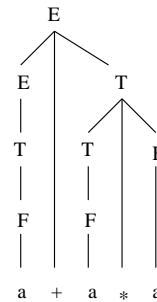
$$\begin{aligned}
 E &\Rightarrow E + T \Rightarrow T + T \Rightarrow F + T \Rightarrow \\
 &a + T \Rightarrow a + T * F \Rightarrow \\
 &a + F * F \Rightarrow a + a * F \Rightarrow \\
 &a + a * a
 \end{aligned}$$

A.S.D.	
Forma sentencial	Regla a derivar
<u>E</u>	r-1
<u>E</u> +T	r-2
<u>T</u> +T	r-4
<u>E</u> +T	r-6
a+ <u>T</u>	r-3
a+ <u>T</u> *F	r-4
a+ <u>F</u> *F	r-6
a+a* <u>F</u>	r-6
a+a*a	

## INTRODUCCIÓN AL AS

### Ejemplo-3 (ASA)

- 1)  $E ::= E + T$
- 2)  $E ::= T$
- 3)  $T ::= T * F$
- 4)  $T ::= F$
- 5)  $F ::= ( E )$
- 6)  $F ::= a$



### Secuencia de Derivación a Derechas

$$\begin{aligned}
 E &\Rightarrow E + T \Rightarrow E + T * F \Rightarrow \\
 &E + T * a \Rightarrow E + F * a \Rightarrow \\
 &E + a * a \Rightarrow T + a * a \Rightarrow \\
 &F + a * a \Rightarrow a + a * a
 \end{aligned}$$

A.S.A.	
Forma sentencial	Regla a reducir
<u>a</u> +a*a	r-6
<u>F</u> +a*a	r-4
<u>T</u> +a*a	r-2
<u>E</u> +a*a	r-6
<u>E</u> + <u>F</u> *a	r-4
<u>E</u> + <u>T</u> *a	r-6
<u>E</u> + <u>T</u> * <u>F</u>	r-3
<u>E</u> + <u>T</u>	r-1
<u>E</u>	

## INTRODUCCIÓN AL AS

### Tipos de Análisis Sintáctico: problema de indeterminismo

- Métodos con retroceso Complejidad exponencial
- Métodos tabulares: Complejidad cúbica
  - Algoritmo de *Cocke-Younger-Kasami*
  - Algoritmo de *Earley*
- Métodos deterministas: Complejidad lineal
  - ⇒ Caracterizar la clase de gramáticas deterministas
  - ⇒ Diseñar el algoritmo de AS determinista