

Sesión 5

Analizador Sintáctico

ASD Predictivo LL

Antonio Moratilla Ocaña

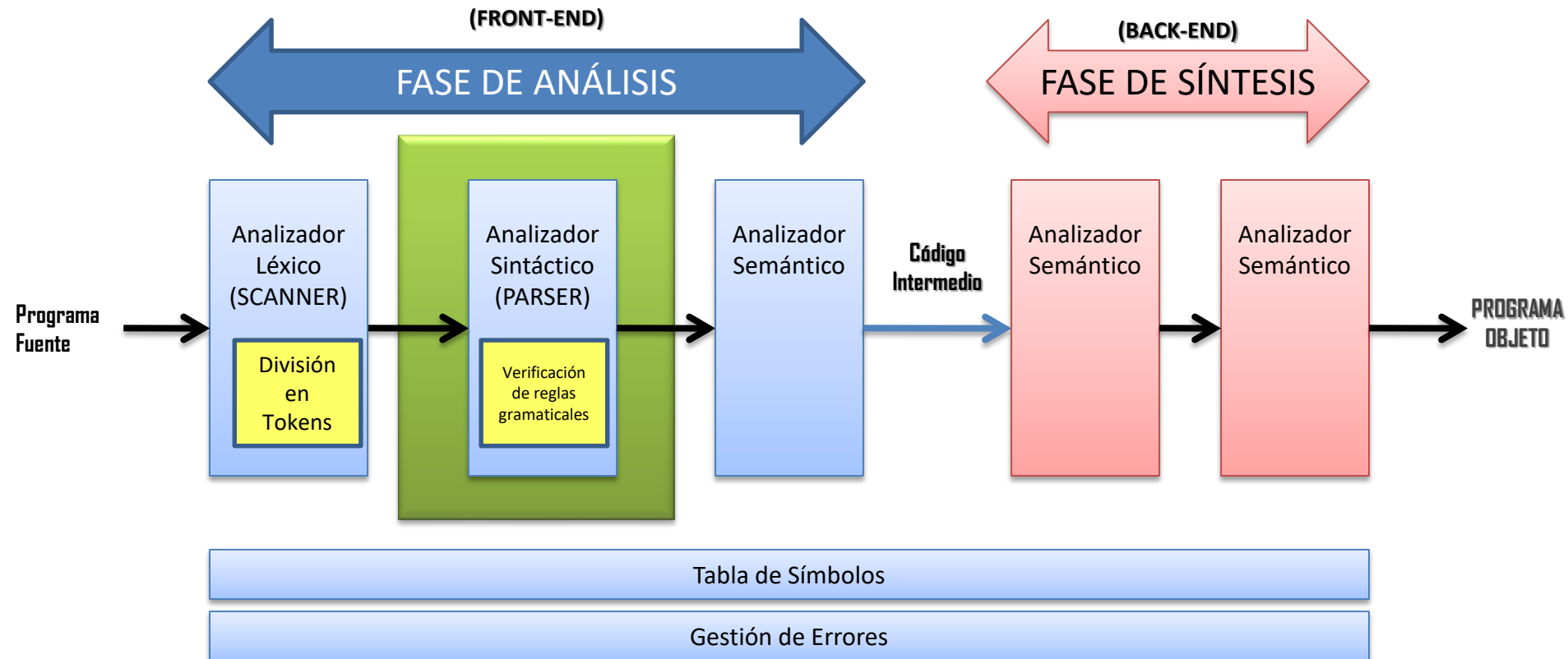


Resumen del tema

- Objetivo:
 - Estudiar cómo se produce el análisis descendente de una expresión para una gramática, tanto recursivo como predictivo.



Posición en el diagrama



Retomando el hilo...

-
- Ya sabemos
 - Que el ASD Recursivo tiene limitaciones por los retrocesos
 - Que el ASD Predictivo elimina esos retrocesos mediante una tabla de análisis sintáctico donde se predice el salto para un símbolo de entrada.
 - Lo que queremos saber
 - ¿Cómo construimos esa tabla de análisis sintáctico para el ASD Predictivo?



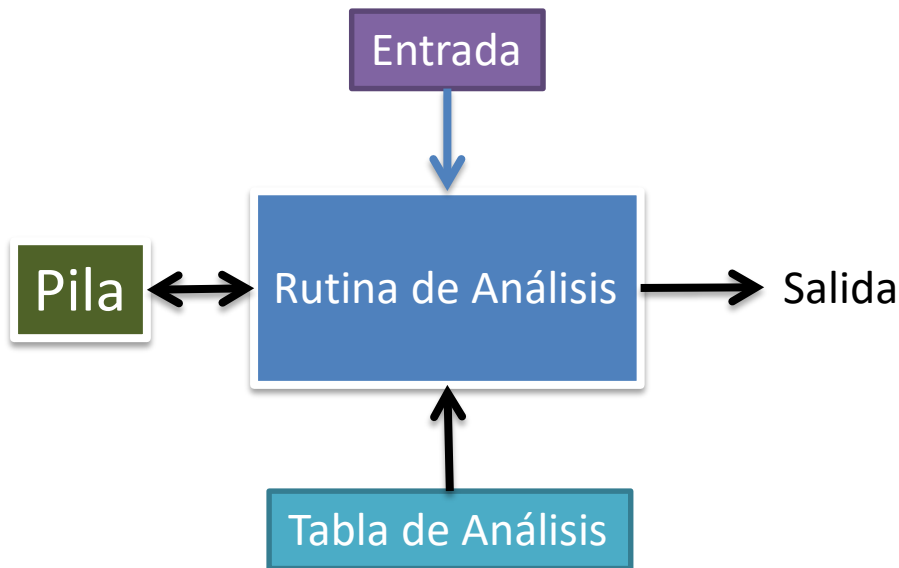
RECORDANDO: ASD Predictivo

- Intentan predecir la siguiente construcción a aplicar leyendo uno o más componentes léxicos por adelantado
 - “Saben” con exactitud qué regla deben expandir para llegar a la entrada
- Este tipo de analizadores se denomina LL(k)
 - Leen la entrada de izquierda a derecha (Left to right)
 - Aplican derivaciones por la izquierda para cada entrada (Left)
 - Utilizan k componentes léxicos de la entrada para predecir la dirección del análisis.
- Están formados por:
 - Un buffer para la entrada.
 - Una pila de análisis.
 - Una tabla de análisis sintáctico.
 - Una rutina de control.



RECORDANDO: ASD Predictivo

- En función de la entrada, de la tabla de análisis y de la pila decide la acción a realizar.



Posibles Acciones:

1. **Aceptar la cadena.**
2. **Aplicar Producción.**
3. **Pasar al siguiente símbolo de entrada.**
4. **Notificar Error.**



RECORDANDO: Tabla de análisis sintáctico

- Se trata de una matriz **M** [**V_n**, **V_t**] donde se representan las producciones a expandir en función del estado actual del análisis y del símbolo de la entrada.
- Las entradas en blanco indican errores

	a	b	c	d	e	\$
S	$S \rightarrow BA$	error	error	$S \rightarrow BA$	error	error
A	error	$A \rightarrow bSC$	error	error	$A \rightarrow \epsilon$	$A \rightarrow \epsilon$
B	$B \rightarrow DC$	error	error	$B \rightarrow DC$	error	error
C	error	$C \rightarrow \epsilon$	$C \rightarrow cDC$	error	$C \rightarrow \epsilon$	$C \rightarrow \epsilon$
D	$D \rightarrow a$	error	error	$D \rightarrow dSe$	error	error



RECORDANDO: ASD Predictivo - Ejemplo

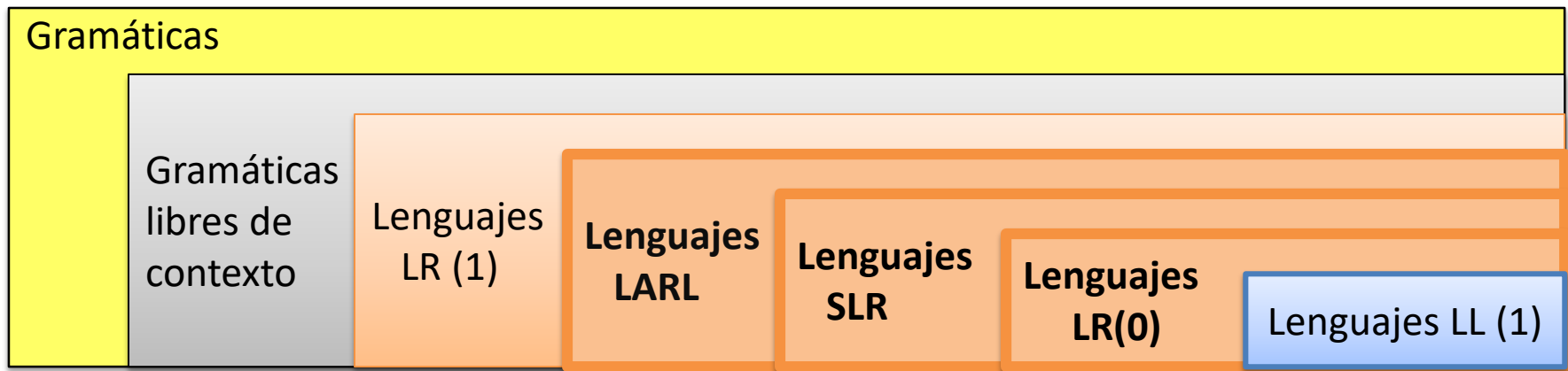
GRAMATICA		Cadena entrada	Símbolo en la entrada	Regla a aplicar *	Pila	Derivaciones aplicadas
A → B A → aBc A → xC B → bA C → c	1	babxccc	b	A→B	B	B
	2	babxccc	b	B→bA	bA	bA
	3	abxccc	a	A→aBc	aBc	baBc
	4	bxccc	b	B→bA	bAc	babAc
	5	xccc	x	A→xC	xCc	babxCc
	6	cc	c	C→c	cc	babxccc
	7	c	c		c	babxccc
	8					babxccc

* La regla a aplicar vendría dada por la tabla de análisis sintáctico.



Clasificación de gramáticas y lenguajes

- Los lenguajes definidos por las gramáticas se clasifican en función de las posibilidades de los métodos de análisis de los mismos:



Analizador LL(1)

- Una gramática es **LL(1)** si la tabla de análisis sintáctico asociada tiene como máximo una producción en cada entrada de la tabla
 - Es decir, sólo se necesita mirar el primer token (testigo) de la entrada para decidir qué regla aplicar
- El análisis descendente más común es LL(1) pues con $k > 1$ la tabla de análisis es mucho más complicada
 - No se usan LL(2), LL(3), etc. en la práctica
- Sin embargo, no todas las gramáticas se pueden analizar mediante métodos descendentes predictivos.

Para determinar si a una gramática se le puede aplicar el método **LL(1)**, debe comprobarse que:

- No es recursiva por la izquierda
- Si existe un conjunto de reglas de la forma $A \rightarrow A\alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_n$, es posible decidir la opción a escoger mirando un token de la entrada, i.e., dos productores de un mismo terminal no pueden dar lugar al mismo testigo (símbolo terminal) inicial.



Analizador LL(1) Conjuntos de Predicción

Para construir la Tabla de Análisis y probar si la gramática es LL(1) se construyen dos conjuntos:

A. Conjuntos de **Primeros**.

B. Conjuntos de **Siguientes**.

- Los conjuntos **Primeros** se calculan para todos los símbolos de la gramática (terminales y no terminales). Los Primeros de los símbolos terminales son triviales, al ser el mismo símbolo.
- Los conjuntos **Siguiente** se definen sólo para los no terminales.



Analizador LL(1) - Conjunto Primero

Dado un símbolo de la gramática A, su **conjunto Primero** es el conjunto de terminales por los que comienza cualquier frase que se genere a partir de A mediante derivaciones por la izquierda de las producciones.

$$a \in \text{Primero}(\alpha_p) \text{ si } a \in (V_t \cup \varepsilon) \mid \alpha_p \rightarrow^* a \beta$$

- Algoritmo de cálculo del conjunto primero:
 - Si $A \in V_t \Rightarrow \text{Prim}(A) = A$
 - Si $\exists A \rightarrow \varepsilon \Rightarrow \varepsilon \in \text{Prim}(A)$
 - Si $\exists A \rightarrow Y_1 Y_2 \dots Y_n \Rightarrow \{\text{Prim}(Y_1) - \varepsilon\} \in \text{Prim}(A)$
 - Si $\varepsilon \in \text{Prim}(Y_1) \Rightarrow \{\text{Prim}(Y_2) - \varepsilon\} \in \text{Prim}(A)$
 - Si $\varepsilon \in \text{Prim}(Y_2) \Rightarrow \{\text{Prim}(Y_3) - \varepsilon\} \in \text{Prim}(A)$
 - ...
 - Si $\forall i, \varepsilon \in \text{Prim}(Y_i) \Rightarrow \varepsilon \in \text{Prim}(A)$.



Analizador LL(1) - Conjunto Primero

Ejemplo 1

Para la gramática:

$A \rightarrow aBc$

$A \rightarrow xC$

$A \rightarrow B$

$B \rightarrow bA$

$C \rightarrow c$

Solo símbolos terminales o épsilon que puedan aparecer en la primera posición aplicando las reglas de derivación

Conjuntos *Primeros*:

- $\text{Prim}(A) = \{a, x\} \cup \{\text{Prim}(B) - \epsilon\} = \{a, x, b\}$
- $\text{Prim}(B) = \{b\}$
- $\text{Prim}(C) = \{c\}$



Analizador LL(1) - Conjunto Primero

Ejemplo 2

Para la gramática:

$$E \rightarrow T E'$$

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

$$T \rightarrow F T'$$

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

$$F \rightarrow (E) \mid n$$

Conjuntos *Primeros*:

- $\text{PRIMERO}(E) = \text{PRIMERO}(TE')$ (*en adelante escribiremos simplemente $\text{PRIMERO}(E)$*).
- $\text{PRIMERO}(E) = \text{PRIMERO}(T)$, así que pasamos a calcular $\text{PRIMERO}(T)$.
- $\text{PRIMERO}(T) = \text{PRIMERO}(F)$, así que pasamos a calcular $\text{PRIMERO}(F)$.
- $\text{PRIMERO}(F) = \{ (, n \}$.

En este punto ya sabemos que:

- $\text{PRIMERO}(E) = \text{PRIMERO}(T) = \text{PRIMERO}(F) = \{ (, n \}$
- $\text{PRIMERO}(E') = \{ + , \varepsilon \}$.
- $\text{PRIMERO}(T') = \{ * , \varepsilon \}$.



Analizador LL(1) - Conjunto Siguientes

Dado un símbolo **A**, su **Conjunto SIGUIENTE** es el conjunto de terminales que aparecen inmediatamente después de **A** en alguna sentencia por la que se pasa al realizar una derivación por la izquierda.

$$a \in \text{Siguientes}(A) \text{ si } a \in (V_t \cup \{\$ \}) \mid S \rightarrow^* \dots Aa \dots$$

- Algoritmo de cálculo del conjunto de Siguientes aplicando las siguientes reglas :
 1. Siguiente (S) = \$, siendo S es el axioma.
 - Para cada símbolo **no-terminal** A de la gramática, añadir
 2. Si $\exists A \rightarrow \alpha B \beta \Rightarrow \{\text{Primero}(\beta) - \epsilon\} \in \text{Siguiente}(B)$
 3. Si $\exists A \rightarrow \alpha B$ ó $(A \rightarrow \alpha B \beta \text{ donde } \epsilon \in \text{Prim}(\beta)) \Rightarrow \text{Sig}(B) \in \text{Sig}(A)$

Nota : ϵ nunca es elemento de un conjunto siguiente.



Analizador LL(1) - Conjunto Siguientes

Ejemplo 1

Para la gramática:

$$E \rightarrow T E'$$

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

$$T \rightarrow F T'$$

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

$$F \rightarrow (E) \mid n$$

$$\text{Prim}(E)=\text{Prim}(T)=\text{Prim}(F)=\{(, n\}; \text{Prim}(E')=\{+, \varepsilon\}; \text{Prim}(T')=\{+, \varepsilon\}$$

Se aplica a los no terminales de la gramática. Devuelve el conjunto de terminales que aparecen a continuación de **A** en alguna forma sentencial derivada del símbolo inicial y el símbolo (\$) que representa el final de la cadena de entrada.

Conjuntos *Siguientes* :

- $\text{SIGUIENTE}(E) = \{\$ + \text{SIGUIENTE}(E)\} = \text{SIGUIENTE}(E) \cup \{) \} = \{ \$,) \}$, hemos buscado E en las partes derechas de las reglas y hemos encontrado $F \rightarrow (E)$, comprobando que $\text{PRIMERO}() = \{) \}$. Como E no aparece en la parte derecha de ninguna otra regla, cerramos su conjunto.
- $\text{SIGUIENTE}(E') = \text{SIGUIENTE}(E) = \{ \$,) \}$, E' aparece el último en la parte derecha de dos reglas de producción, aunque el No Terminal que las origina es el mismo: E. Por tanto, se añade $\text{SIGUIENTE}(E)$ a $\text{SIGUIENTE}(E')$ y, como E' no aparece más, cerramos su conjunto.
- $\text{SIGUIENTE}(T) = \text{PRIMERO}(E') \cup \text{SIGUIENTE}(E') = \{ + , \$,) \}$, hemos aplicado $E \rightarrow +TE'$ pues el No Terminal T podría ser el último a la derecha, ya que E' deriva la palabra vacía.
- $\text{SIGUIENTE}(T') = \text{SIGUIENTE}(T) = \{ + , \$,) \}$.
- $\text{SIGUIENTE}(F) = \text{PRIMERO}(T') \cup \text{SIGUIENTE}(T) \cup \text{SIGUIENTE}(T') = \{ * , + , \$,) \}$.

El resultado será:

$$\text{SIGUIENTE}(E) = \{ \$,) \}, \text{SIGUIENTE}(E') = \{ \$,) \}, \text{SIGUIENTE}(T) = \{ + , \$,) \}$$

$$\text{SIGUIENTE}(T') = \{ + , \$,) \}, \text{SIGUIENTE}(F) = \{ * , + , \$,) \}$$



Analizador LL(1) Construcción de la tabla de análisis

REGLAS de construcción de la Tabla de Análisis ($M[V_n, V_t]$)

Repita todo este proceso para cada producción $A \rightarrow \alpha$ de la gramática

1. Para cada terminal $a \in \text{Primero}(\alpha)$:
 - Agregue $A \rightarrow \alpha$ a la tabla $M[A, a]$
2. Si $\epsilon \in \text{Primero}(\alpha)$ (porque α deriva ϵ) :
 - Agregar la producción $A \rightarrow \alpha$ en la posición $M[A, b]$ de la tabla para cada terminal $b \in \text{Siguierte}(A)$.
3. Si $\epsilon \in \text{Primero}(\alpha)$ y $\$ \in \text{Siguierte}(A)$:
 - Agregue $A \rightarrow \alpha$ en la posición $M[A, \$]$

Notas:

- Cada entrada en blanco es una posición de error
- Si alguna de las entradas tiene más de una producción es porque la gramática no es LL(1)



Analizador LL(1) - Ejemplo

1. Comprobar si la siguiente gramática es LL(1) y construir la tabla, calculando todos los conjuntos (Siguiendo y Primero).

$S \rightarrow c A$

$A \rightarrow a B$

$B \rightarrow b \mid \epsilon$

2. Reconocer la cadena “**cab**” con el analizador construido.



Analizador LL(1) - Ejemplo

Pasos para ver si es LL(1):

1. No es recursiva izquierda

2. $B \rightarrow b \mid \epsilon$

I. $\text{Prim}(b) \cap \text{Prim}(\epsilon) = \emptyset$

II. Si $\epsilon \in \text{Prim}(\epsilon) \Rightarrow \text{Prim}(b) \cap \text{Sig}(B) = \emptyset$

Conjuntos primero:

$\text{Prim}(S) = \{c\}$, $\text{Prim}(A) = \{a\}$, $\text{Prim}(B) = \{b, \epsilon\}$

Conjuntos siguiente:

$\text{Sig}(S) = \{\$ \}$, $\text{Sig}(A) = \text{Sig}(S) = \{\$ \}$, $\text{Sig}(B) = \text{Sig}(A) = \{\$ \}$

Para determinar si a una gramática se le puede aplicar el método LL(1), debe comprobarse que:

1. No es recursiva por la izquierda
2. Si existe un conjunto de reglas de la forma $A \rightarrow A\alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_n$ se cumple que no hay dos partes derechas que comiencen por un mismo terminal:

$\text{Primero}(\alpha_i) \cap \text{Primero}(\alpha_j) = \{\}, \forall i \neq j$

• A lo sumo un α_i puede derivar la cadena vacía .

• Si un no terminal $\alpha_i \Rightarrow \epsilon$, entonces no hay otro α_j que comience por un elemento que esté en

$\text{Siguiente}(A)$:

$\text{Primero}(\alpha_i) \cap \text{Siguiente}(A) = \emptyset$



Analizador LL(1) - Ejemplo

Gramática

$S \rightarrow c A$

$\text{Prim}(S) = \{c\}$, $\text{Prim}(A) = \{a\}$, $\text{Prim}(B) = \{b, \epsilon\}$

$A \rightarrow a B$

$\text{Sig}(S) = \{\$ \}$, $\text{Sig}(A) = \text{Sig}(S) = \{\$ \}$, $\text{Sig}(B) = \text{Sig}(A) = \{\$ \}$

$B \rightarrow b \mid \epsilon$

Tabla de Análisis

	c	a	b	\$
S	$S \rightarrow c A$			
A		$A \rightarrow a B$		
B			$B \rightarrow b$	$B \rightarrow \epsilon$

Regla1

Para cada terminal $a \in \text{Primero}(\alpha)$:
Agregue $A \rightarrow \alpha$ a la tabla $M[A, a]$

Si $\epsilon \in \text{Primero}(\alpha)$ y
 $\$ \in \text{Siguiete}(A)$:
Agregue $A \rightarrow \alpha$ en la
posición $M[A, \$]$

Regla 3



Analizador LL(1) - Ejemplo

Reconocer la cadena “cab”

Gramática

$S \rightarrow c A$

$A \rightarrow a B$

$B \rightarrow b \mid \epsilon$

Pila	Acción	Entrada
\$S		cab\$
\$A c	Reconocimiento	c ab\$
\$A	$A \rightarrow a B$	ab\$
\$B a	Reconocimiento	a b\$
\$B	$B \rightarrow b$	b\$
\$ b	Reconocimiento	b \$
\$	éxito	\$



Analizador LL(1) – Ejemplo 2

Pasos para ver si es LL(1):

1. Comprobar que No es recursiva por la izquierda

Cuando se da alguno de estos casos hay que modificar la gramática:

1. Eliminando la recursión por la izquierda
2. Sacando algún factor común por la izquierda

Para determinar si a una gramática se le puede aplicar el método LL(1), debe comprobarse que:

1. No es recursiva por la izquierda
2. Si existe un conjunto de reglas de la forma $A \rightarrow A\alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_n$ se cumple que no hay dos partes derechas que comiencen por un mismo terminal:

Primero $(\alpha_i) \cap \text{Primero}(\alpha_j) = \{\}, \forall i \neq j$

• A lo sumo un α_i puede derivar la cadena vacía.

• Si un no terminal $\alpha_i \Rightarrow \epsilon$, entonces no hay otro α_j que comience por un elemento que esté en $\text{Siguiente}(A)$:

Primero $(\alpha_i) \cap \text{Siguiente}(A) = \emptyset$

Gramática original	Gramática modificada
$A \rightarrow A a$ $A \rightarrow bB$	$A \rightarrow bB A'$ $A' \rightarrow aA' \mid \epsilon$ se elimina la recursividad
$B \rightarrow b c$ $B \rightarrow b b$ $B \rightarrow b$	$B \rightarrow b B'$ $B' \rightarrow c \mid b \mid \epsilon$ se saca los factores común y se agrupan



Resumen de Primeros y Siguientes

- **Cálculo del conjunto Primero**

$\text{Primero}(\alpha)$ es el conjunto de *tokens* que están al principio de las hileras derivadas a partir de α .

Si $\alpha \Rightarrow^* \epsilon$, entonces $\epsilon \in \text{Primero}(\alpha)$

X es cualquier símbolo gramatical Comience con $\text{Primero}(X) == \{ \}$

- Si $X == a$ es terminal $\text{Primero}(X) == \{X\} == \{a\}$
- Si $X \rightarrow \epsilon$ es una producción, agregue ϵ a $\text{Primero}(X)$
- Si X es no terminal y $X \rightarrow Y_1 Y_2 \dots Y_n$ es una producción
Repita desde 1 hasta n lo siguiente ($1 \leq i \leq n$)

- Agregue $\text{Primero}(Y_i) - \{ \epsilon \}$ a $\text{Primero}(X)$

- Si $\epsilon \notin \text{Primero}(Y_i)$ termine el cálculo

- Si llegó hasta ($i == n$)

- Agregue ϵ a $\text{Primero}(X)$ (pues Y_i deriva ϵ para todos los valores $1 \leq i \leq n$)

- Si $\epsilon \in \text{Primero}(X)$ es porque X deriva ϵ

<http://www.di-mare.com/pub/ayc/>



Resumen de Primeros y Siguientes

- **Cálculo del conjunto Siguiente**

Siguiente(B) se calcula únicamente para los no terminales B.

- Calcule $\text{Primero}(X)$ para todo símbolo gramatical X.
- Agregue \$ a $\text{Siguiente}(S)$ para el símbolo inicial S
- Si hay una producción $A \rightarrow \alpha B \beta$, todo lo que esté en $\text{Primero}(\beta)$ excepto ϵ se pone en $\text{Siguiente}(B)$:
$$\text{Siguiente}(B) \leftarrow \text{Siguiente}(B) \cup (\text{Primero}(\beta) - \{ \epsilon \})$$
- Si hay una producción $A \rightarrow \alpha B$ o una producción $A \rightarrow \alpha B \beta$ donde $\epsilon \in \text{Primero}(\beta)$ (porque β deriva ϵ):
Agregue en $\text{Siguiente}(B)$ todo lo que esté en $\text{Siguiente}(A)$
$$\text{Siguiente}(B) \leftarrow \text{Siguiente}(B) \cup \text{Siguiente}(A)$$

<http://www.di-mare.com/pub/ayc/>



Resumen de Primeros y Siguientes

Construcción de las Tablas LL(1)

- Repita todo este proceso para cada producción $A \rightarrow \alpha$ de la gramática para cada terminal $a \in \text{Primer}(\alpha)$
 1. Agregue $A \rightarrow \alpha$ a la tabla $M[A, a]$
 2. Si $\epsilon \in \text{Primer}(\alpha)$ (porque α deriva ϵ)
Agregue la producción $A \rightarrow \alpha$ en la posición $M[A, b]$ de la tabla $M[]$ para cada terminal $b \in \text{Sigui}(\alpha)$.
 3. Si $\epsilon \in \text{Primer}(\alpha)$ y $\$ \in \text{Sigui}(\alpha)$
Agregue $A \rightarrow \alpha$ en la posición $M[A, \$]$
- Cada entrada en blanco en $M[]$ es una posición de error
- Si alguna de las entradas de $M[]$ tiene más de una producción es porque la gramática no es LL(1)

<http://www.di-mare.com/pub/ayc/>



Ejercicios

- Dada la gramática $G(A \rightarrow a B b ; B \rightarrow c d \mid c)$
- Realice en proceso de análisis sintáctico descendente predictivo para reconocer las expresiones
 - acdb
 - abcd
 - acb



Ejercicios

- Dada la gramática
 - $A \rightarrow B C$
 - $B \rightarrow D E$
 - $C \rightarrow \varepsilon \mid z B C$
 - $D \rightarrow y A x \mid v$
 - $E \rightarrow w D E \mid \varepsilon$
- Realice en proceso de análisis sintáctico descendente predictivo para reconocer las expresiones
 - ywx
 - v
 - $yvxwvzv$



ASD Predictivo Tabla de análisis sintáctico

- Se trata de una matriz **M** [**V_n**, **V_t**] donde se representan las producciones a expandir en función del estado actual del análisis y del símbolo de la entrada.
- Las entradas en blanco indican errores

	a	b	c	d	e	\$
S	$S \rightarrow BA$	error	error	$S \rightarrow BA$	error	error
A	error	$A \rightarrow bSC$	error	error	$A \rightarrow \epsilon$	$A \rightarrow \epsilon$
B	$B \rightarrow DC$	error	error	$B \rightarrow DC$	error	error
C	error	$C \rightarrow \epsilon$	$C \rightarrow cDC$	error	$C \rightarrow \epsilon$	$C \rightarrow \epsilon$
D	$D \rightarrow a$	error	error	$D \rightarrow dSe$	error	error



ASD Predictivo - Ejemplo

GRAMATICA		Cadena entrada	Símbolo en la entrada	Regla a aplicar *	Pila	Derivaciones aplicadas
$A \rightarrow B$ $A \rightarrow aBc$ $A \rightarrow xC$ $B \rightarrow bA$ $C \rightarrow c$	1	babxccc	b	$A \rightarrow B$	B	B
	2	babxccc	b	$B \rightarrow bA$	bA	bA
	3	abxccc	a	$A \rightarrow aBc$	aBc	baBc
	4	bxccc	b	$B \rightarrow bA$	bAc	babAc
	5	xccc	x	$A \rightarrow xC$	xCc	babxCc
	6	ccc	c	$C \rightarrow c$	cc	babxccc
	7	c	c		c	babxccc
	8					babxccc

* La regla a aplicar vendría dada por la tabla de análisis sintáctico.



Fuentes

- Para la elaboración de estas transparencias se han utilizado:
 - Transparencias de cursos previos (elaboradas por los profesores Dr. D. Salvador Sánchez, Dr. D. José Luis Cuadrado).
 - Libros de referencia (en especial capítulos 2 y 4 de Aho).

