

Autómatas y Lenguajes

3^{er} curso
1^{er} cuatrimestre

Alfonso Ortega: alfonso.ortega@uam.es



UNIDAD 2: Procesadores de lenguaje

TEMA Análisis sintáctico

Introducción al análisis sintáctico



Tema Analizador sintáctico

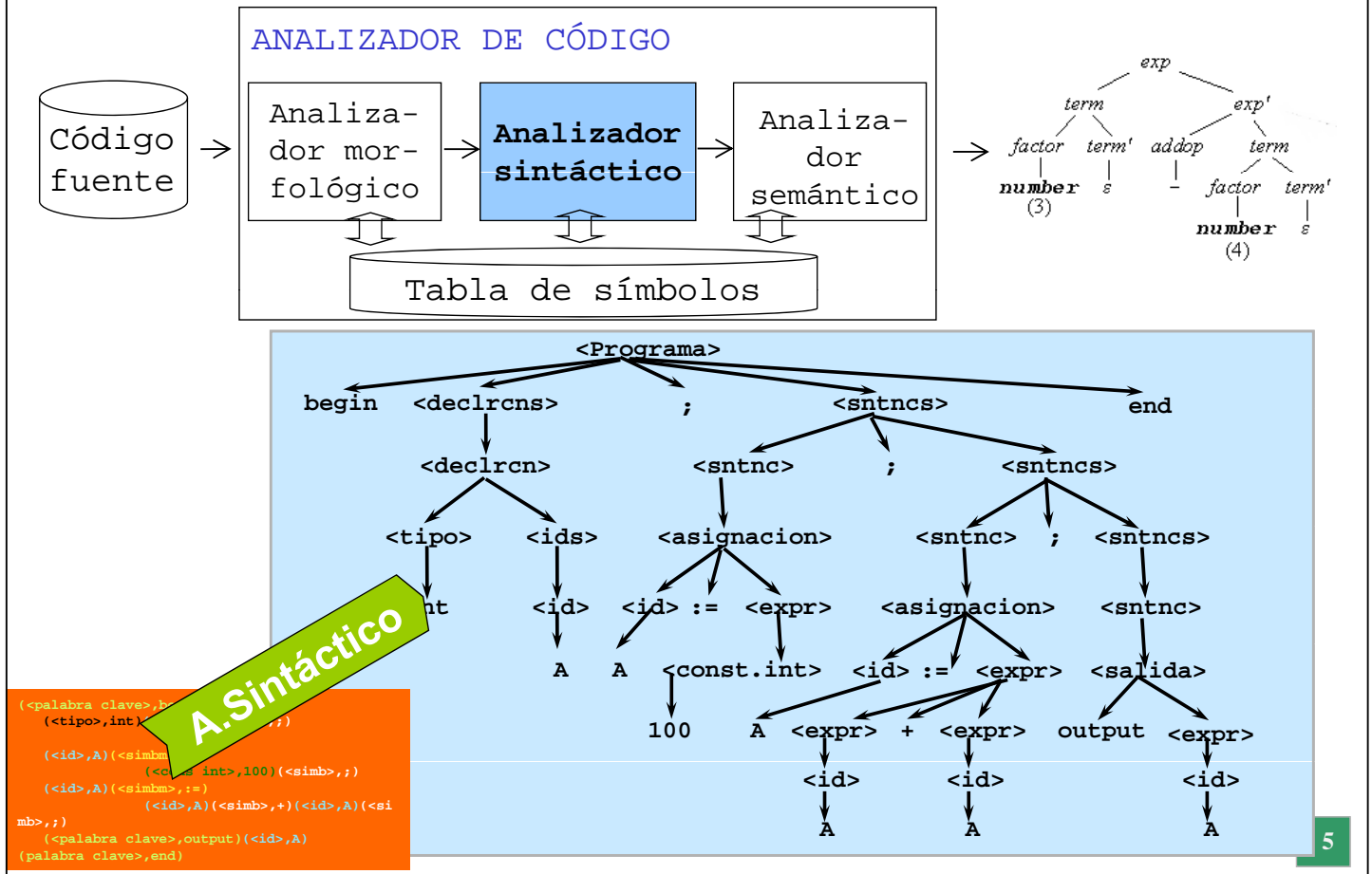
- 1 Introducción y objetivos
- 2 Estrategias de análisis sintáctico
 - 2.1 Ascendente (Top-down)
 - 2.2 Descendente (Bottom-up)
- 3 Tipos de analizadores
- 4 Ejemplo de analizador muy sencillo:
 - 4.1 Ascendente (Top-down)
 - 4.2 Descendente (Bottom-up)



1

Introducción y objetivos

Analizador sintáctico (I)



Analizador sintáctico (II)

Introducción (I)

- También se le puede llamar **parser**.
- Recuérdese que, en el paso de análisis morfológico, se trataba la parte regular del lenguaje fuente y se identificaban las **unidades sintácticas**.
- Estas unidades sintácticas son los símbolos terminales de la gramática que trata la parte que es **independiente del contexto** del lenguaje fuente.
- El **analizador sintáctico** se ocupa específicamente de este tratamiento y suele ser el que **gobierna todo el proceso de análisis**.
- No todas las construcciones de los lenguajes de programación son regulares (ya se vio en el tema de análisis morfológico).
- Ni independientes del contexto
 - Declaración de variables antes de su uso.
 - Acuerdo entre la declaración de funciones y su llamada.
- Para procesar las construcciones de los lenguajes de programación que no son **independientes del contexto** (más asociadas a la "semántica" del lenguaje) hace falta otras herramientas (gramáticas de atributos) no únicamente la gramática independiente del contexto, entre las que están estructuras de datos adicionales como la **tabla de símbolos**.

Analizador sintáctico (III)

Introducción (II): ejercicio

- Considere el siguiente fragmento de una gramática de ASPL

```
7 : <stm train> ::= <statement>
8 :               | <statement> ; <stm train>
  ...
22 : <cond stm> ::= if <exp> then <stm train> fi
23 :               | if <exp> then <stm train> else <stm train> fi
24 : <loop stm> ::= while <exp> do <stm train> end
25 :               | repeat <stm train> until <exp>
```

- Estas reglas representan respectivamente las construcciones de control de flujo de programa de la programación estructurada:
 - Bloque de sentencias
 - Sentencia de control de flujo condicional
 - Sentencia de control de flujo iterativa
- ¿Son construcciones dependientes o independientes del contexto? ¿Por qué?

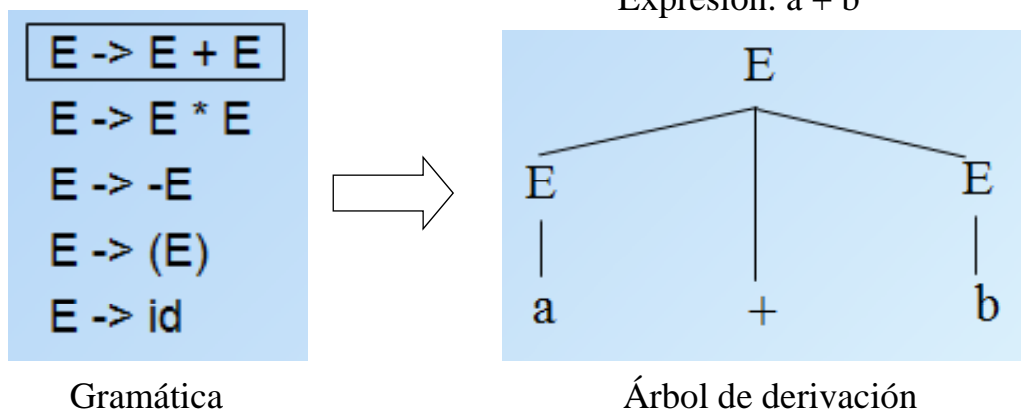
Autómatas y Lenguajes (A.O.P.)

7

Analizador sintáctico (III)

Introducción (III)

- El proceso de análisis puede entenderse como el mecanismo para obtener el “árbol sintáctico de derivación del programa” (en caso de ser correcto) que es una manera de poder **reducir el programa completo al axioma** de la gramática.
- Por ejemplo:



Autómatas y Lenguajes (A.O.P.)

8

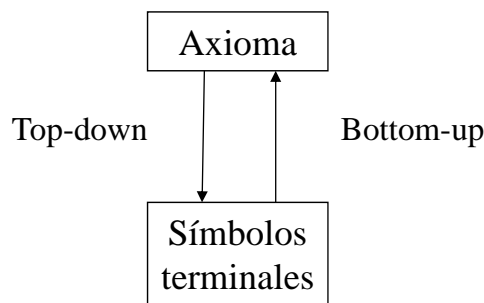
- El principal objetivo del analizador sintáctico es:
 - **Producir un árbol de derivación** que será usado por el procesador.
- Un objetivo secundario es:
 - **Detectar errores sintácticos** – indicar al usuario dónde hay un error (no se cumple la gramática) en el código fuente.
- El analizador sintáctico se puede **diseñar** teniendo en cuenta:
 - Ambos objetivos y que el analizador no pare ante el primer error sintáctico que se encuentre, sino que produzca una **lista de errores**.
 - Sólo la fase de compilación y que el analizador se **pare ante el primer error** que se encuentre.

2

Estrategias de análisis

Estrategias de análisis sintáctico

- Una posible estrategia, como se verá a continuación, construye los árboles de derivación empezando por los símbolos terminales de la cadena que se está analizando, pasando por los no terminales hasta llegar al axioma.
 - Este tipo de estrategia de análisis sintáctico se llama: **bottom-up**.
- Otra estrategia posible es comenzar por el axioma, e ir aplicando reglas hasta transformar todos los símbolos no terminales en terminales y llegar a la expresión a partir del axioma.
 - Este tipo de estrategia de análisis sintáctico se llama: **top-down**.



Autómatas y Lenguajes (A.O.P.)

13

Bottom-up

- A partir de la cadena de “tokens” devueltos por el analizador morfológico, se comienza a leer intentando identificar las partes derechas de las reglas de la gramática.
- Cuando esto ocurre se sustituye la cadena de “tokens” por el no terminal de la parte izquierda de la regla y se continúa el mismo análisis teniendo en cuenta esta sustitución.
- Si en algún momento ninguna secuencia de la cadena se corresponde con la parte derecha de ninguna regla tal vez se tenga que deshacer alguna elección para probar si con otra decisión se consigue terminar el análisis.
- Cuando se han probado sistemáticamente todas las opciones sin poder terminar el análisis se concluye que el programa es sintácticamente incorrecto.
- La otra forma de terminar el análisis es cuando se consigue obtener por el mecanismo descrito el axioma de la gramática.
- En este último caso se concluye que el programa es sintácticamente correcto.

Autómatas y Lenguajes (A.O.P.)

14

- A partir del axioma de la gramática y el programa (transformado en secuencia de “tokens” por el analizador morfológico):
 - Probar con las distintas opciones de derivación para cada no terminal que encontremos.
 - Siempre que haya coincidencia entre el terminal actual en el programa y el terminal de la hoja actual del árbol:
 - Ir avanzando simultáneamente en el programa y en las hojas del árbol de derivación.
 - Hasta que:
 - O bien se recorra por completo la cadena y coincida con las hojas del árbol y, por tanto, el programa es sintácticamente correcto.
 - O bien se intentan todas las opciones posibles para cada no terminal (incluido el axioma) y no se puede completar el análisis por lo que el programa es sintácticamente incorrecto.

Secuencias de derivación (I)

Derivación

- Una ‘derivación’ muestra la secuencia de sustituciones desde el axioma hasta la entrada.
- Una ‘leftmost derivation’ (derivación más a la izquierda) es aquella en la que trabajando de arriba a abajo, se expande siempre el símbolo no terminal situado más a la izquierda.
- Una ‘rightmost derivation’ (derivación más a la derecha) es aquella en la que trabajando de arriba a abajo, se expande siempre el símbolo no terminal situado más a la derecha.

Secuencias de derivación (II)

E
E * E
E * id
(E) * id
(E + E) * id
(E + id) * id
(id + id) * id

E
E * E
(E) * E
(E + E) * E
(id + E) * E
(id + id) * E
(id + id) * id

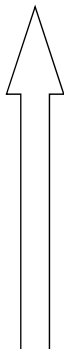
¿Cuál es la derivación más a la derecha?

¿Por qué?

Secuencias de derivación (III)

- Como se puede apreciar la aplicación más a la izquierda de reglas en orden ascendente de análisis produce una derivación más a la derecha:

E
E * E
E * id
(E) * id
(E + E) * id
(E + id) * id
↑
(id + id) * id



3

Tipos de analizadores

Tipos de analizadores sintácticos

- Según los enfoques estudiados, los analizadores sintácticos se pueden clasificar como:
 - Top Down:
 - Analizadores LL
 - Bottom Up
 - Analizadores LR
 - LR(0)
 - SLR(1) (Simple LR)
 - LR(1) (Canonical LR)
 - LALR(1) (LookAhead LR)
 - LR(k)
 - Analizadores de precedencia de operadores (aplicados en muchos lenguajes algebraicos – por ejemplo para expresar consultas -)

4

Ejemplos de analizadores muy sencillos

4.1

Ascendente

Ejemplo de analizador ascendente muy sencillo

- Aplica un análisis ascendente (bottom-up).
- A medida que se avanza en el reconocimiento de la cadena de entrada, va guardando los tokens en una pila.
- Es muy poco eficiente porque en cada paso comprueba todas las reglas de la gramática.
- Se basa en dos operaciones principales:
 - Desplazamiento (shift)
 - Reducción (reduce)

Ejemplo de analizador ascendente muy sencillo

La operación de reducción

- Cuando los tokens en la cima de la pila coincidan con la parte derecha de una regla de la gramática:
 - Reemplaza los tokens de la cima de la pila con la parte izquierda de la regla.
- Ejemplo:

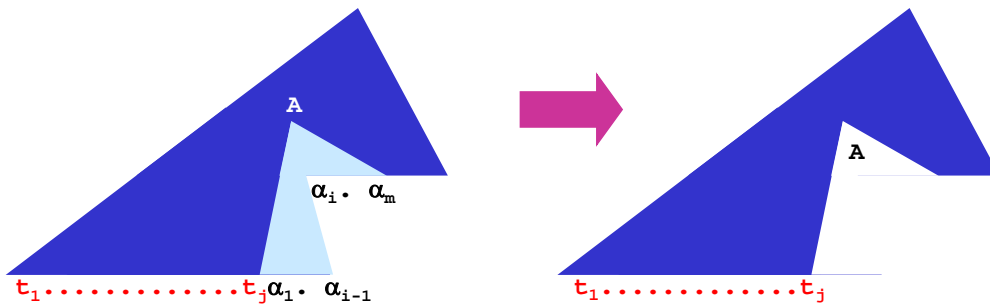


E -> E + E

Ejemplo de analizador ascendente muy sencillo

Concepto de reducción

- Gráficamente
 - Si la regla reducida es
$$A \rightarrow \alpha_1 \dots \alpha_m$$
 - Podría imaginarse su efecto en el árbol de análisis de la siguiente manera



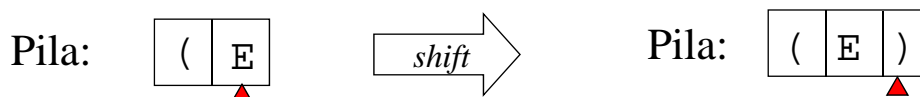
Autómatas y Lenguajes (A.O.P.)

25

Ejemplo de analizador ascendente muy sencillo

La operación de desplazamiento

- Cuando no se puede realizar una operación de reducción, entonces hay que aplicar la operación de desplazamiento:
 - Avanzar una posición en la cadena de entrada.
 - Copiar el símbolo en la pila
- Ejemplo:



Input: (id + id) * id



Autómatas y Lenguajes (A.O.P.)

26

Ejemplo de analizador ascendente muy sencillo

Concepto de desplazamiento

- Intuitivamente, el efecto sobre el analizador es el siguiente:
 - Corresponde al caso en el que los terminales encontrados en la cadena que se está analizando “van correspondiendo” con lo que se espera como posible parte derecha de alguna regla.
 - Los analizadores toman nota de la circunstancia de forma que guardan la suficiente información para actualizar la posición en la parte derecha de todas las reglas que podrían finalmente reducirse con la entrada analizada
- Gráficamente se muestra a continuación como algunas reglas se descartan y otras avanzan como la entrada

$N_h \rightarrow \dots t_j t_k N \dots$	\rightarrow	$N_h \rightarrow \dots t_j t_k N \dots$
$N_v \rightarrow \dots t_j t_{j+1} N \dots$		$N_v \rightarrow \dots t_j t_{j+1} N \dots$
$N_b \rightarrow \dots t_j t_{j+1} N \dots$		$N_b \rightarrow \dots t_j t_{j+1} N \dots$
$N_g \rightarrow \dots t_j t_1 N \dots$		$N_g \rightarrow \dots t_j t_1 N \dots$
$N_t \rightarrow \dots t_j t_m N \dots$		$N_t \rightarrow \dots t_j t_m N \dots$
$N_y \rightarrow \dots t_j t_{j+1} N \dots$		$N_y \rightarrow \dots t_j t_{j+1} N \dots$
$N_q \rightarrow \dots t_j t_p N \dots$		$N_q \rightarrow \dots t_j t_p N \dots$

$\dots, t_j t_{j+1} t_{j+2} t_{j+3} t_{j+4} t_{j+5} t_{j+6}$
 t_{j+7}, t_j, \dots

Autómatas y Lenguajes (A.O.P.)

$\dots, t_j t_{j+1} t_{j+2} t_{j+3} t_{j+4} t_{j+5} t_{j+6}$
 t_{j+7}, t_j, \dots

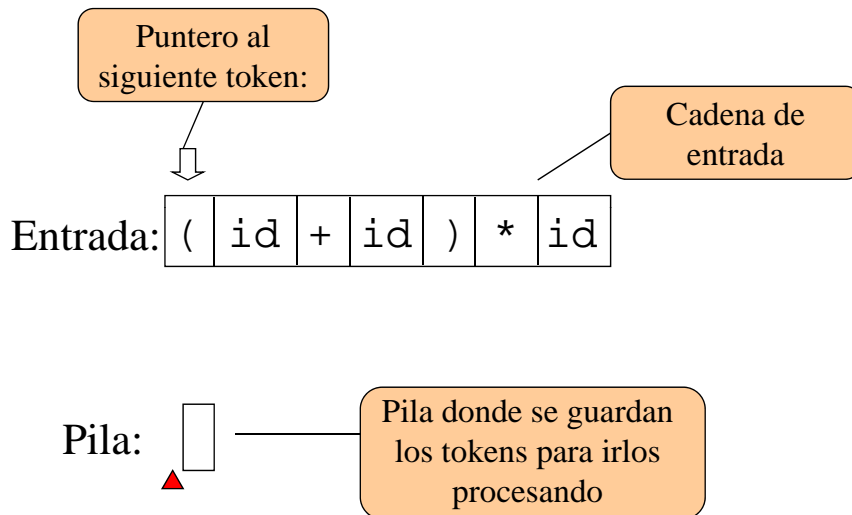
27

Ejemplo de analizador ascendente muy sencillo

Algoritmo

- Se crea la pila vacía.
- Se desplaza el primer token de la cadena a la cima de la pila.
- Mientras quede cadena de entrada:
 - Si se puede reducir:
 - Reducir
 - Si no se puede reducir:
 - Desplazarse al siguiente token de la cadena entrada

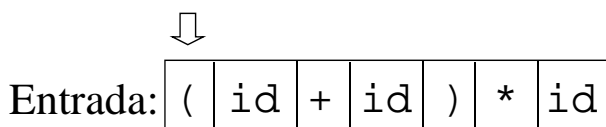
Ejemplo de analizador ascendente muy sencillo



Gramática:

$E \rightarrow E + E$
 $E \rightarrow E * E$
 $E \rightarrow -E$
 $E \rightarrow (E)$
 $E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo



Acción: Desplazar

Mover al siguiente token de la cadena de entrada y guardar éste en la pila

Gramática:

$E \rightarrow E + E$
 $E \rightarrow E * E$
 $E \rightarrow -E$
 $E \rightarrow (E)$
 $E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

Pila: (

Acción: Desplazar

Mover al siguiente token de la cadena de entrada y guardar éste en la pila

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

Pila: (

¿Se puede reducir?

¿coincide la cima de la pila con la parte derecha de alguna regla de la gramática?

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

↓

Pila: (

¿Se puede reducir?

No.

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

↓

Pila: (

¿Se puede reducir?

No.

Acción: desplazar.

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

↓

Pila: (id

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

↓

Pila: (id

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

¿Se puede reducir?

Sí.

Acción: reducir.

Ejemplo de analizador ascendente muy sencillo

Ojo: no se cambia el puntero de la cadena

Entrada: (id + id) * id

Pila: (E

id reemplazado por E

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

Pila: (E

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

¿Se puede reducir?

No.

Acción: desplazar

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

↓

Pila: (E +

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

↓

Pila: (E +

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

¿Se puede reducir?

No.

Acción: desplazar.

Ejemplo de analizador ascendente muy sencillo

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Entrada: (id + id) * id

Pila: (E + id

¿Se puede reducir?

Sí.

Acción: Reducir.

Ejemplo de analizador ascendente muy sencillo

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Entrada: (id + id) * id

Pila: (E + E

Ejemplo de analizador ascendente muy sencillo

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Entrada: (id + id) * id

Pila: (E + E

¿Se puede reducir?

Sí.

Acción: Reducir.

Ejemplo de analizador ascendente muy sencillo

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Entrada: (id + id) * id

Pila: (E

¿Se puede reducir?

No.

Acción: desplazar.

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id



Pila: (E)

¿Se puede reducir?

Sí.

Acción: reducir.

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id



Pila: E

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id



Pila: E *

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id



Pila: E * id

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

Pila: E * E

Gramática:

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$

Ejemplo de analizador ascendente muy sencillo

Entrada: (id + id) * id

Pila: E

Gramática:

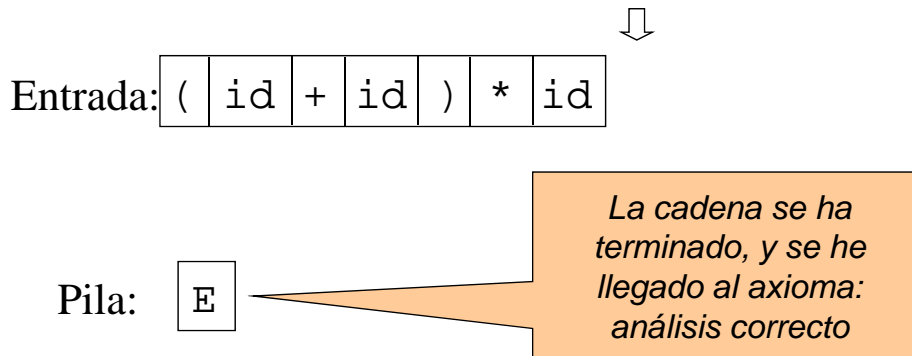
$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow -E$

$E \rightarrow (E)$

$E \rightarrow id$



¿Se ha terminado la cadena?

Sí.

¿El símbolo de la pila es el axioma?

Acción: aceptar.

Autómatas y Lenguajes (A.O.P.)

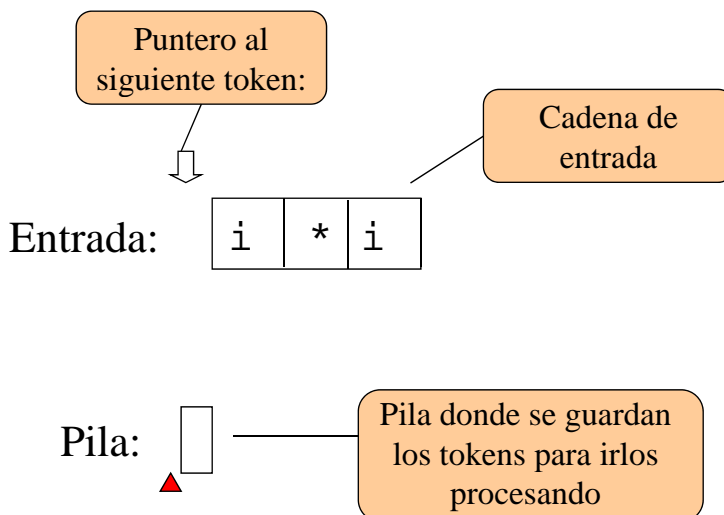
4.2

Descendente

Ejemplo de analizador descendente muy sencillo

- Aplica un análisis descendente (top-down).
- En este caso, es más sencillo porque se ha modificado la gramática para que sólo sea posible aplicar una regla en cada paso.
- Es objetivo de otro tema saber qué técnica permite elegir la regla adecuada en cada paso (según los correspondientes símbolos del árbol y de la entrada)
- Como en el análisis ascendente, a medida que se avanza en el reconocimiento de la cadena de entrada, se van guardando los tokens en una pila.
- Se basa en la operación principal ya estudiada:
 - Derivación

Ejemplo de analizador descendente muy sencillo



Gramática:

```
E -> TB
B -> +TB
    | λ
T -> FX
X -> *FX
    | λ
F -> i
    | (E)
```

Ejemplo de analizador descendente muy sencillo

Entrada: \downarrow

i	*	i
---	---	---

Pila:

E

Gramática:

$E \rightarrow TB$

$B \rightarrow +TB$

$\mid \lambda$

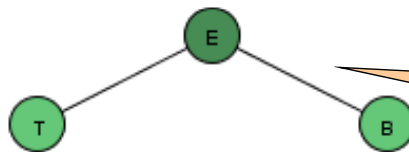
$T \rightarrow FX$

$X \rightarrow *FX$

$\mid \lambda$

$F \rightarrow i$

$\mid (E)$



Se realiza la primera derivación

Ejemplo de analizador descendente muy sencillo

Entrada: \downarrow

i	*	i
---	---	---

Pila:

T	B
---	---

Gramática:

$E \rightarrow TB$

$B \rightarrow +TB$

$\mid \lambda$

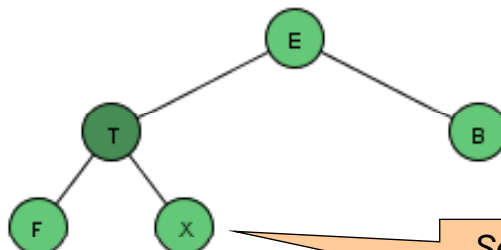
$T \rightarrow FX$

$X \rightarrow *FX$

$\mid \lambda$

$F \rightarrow i$

$\mid (E)$



Se realiza otra derivación

Ejemplo de analizador descendente muy sencillo

Entrada:

i	*	i
---	---	---

Pila:

i	X	B
---	---	---

Gramática:

$E \rightarrow TB$

$B \rightarrow +TB$

$\mid \lambda$

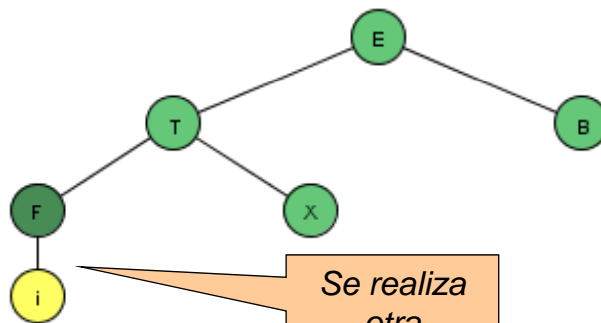
$T \rightarrow FX$

$X \rightarrow *FX$

$\mid \lambda$

$F \rightarrow i$

$\mid (E)$



Se realiza
otra
derivación

Autómatas y Lenguajes (A.O.P.)

57

Ejemplo de analizador descendente muy sencillo

Entrada:

i	*	i
---	---	---

Pila:

*	F	X	B
---	---	---	---

Gramática:

$E \rightarrow TB$

$B \rightarrow +TB$

$\mid \lambda$

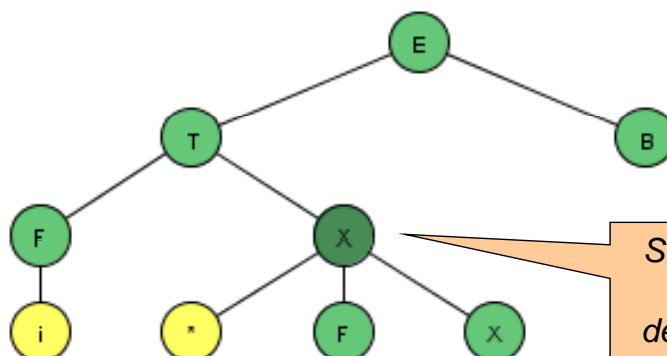
$T \rightarrow FX$

$X \rightarrow *FX$

$\mid \lambda$

$F \rightarrow i$

$\mid (E)$



Se realiza
otra
derivación

Autómatas y Lenguajes (A.O.P.)

58

Ejemplo de analizador descendente muy sencillo

Entrada:

i	*	i
---	---	---

Pila:

F	X	B
---	---	---

Gramática:

$E \rightarrow TB$

$B \rightarrow +TB$

$\mid \lambda$

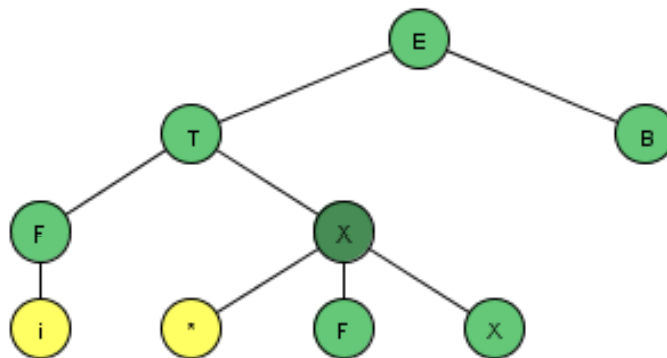
$T \rightarrow FX$

$X \rightarrow *FX$

$\mid \lambda$

$F \rightarrow i$

$\mid (E)$



Autómatas y Lenguajes (A.O.P.)

59

Ejemplo de analizador descendente muy sencillo

Entrada:

i	*	i
---	---	---

Pila:

i	X	B
---	---	---

Gramática:

$E \rightarrow TB$

$B \rightarrow +TB$

$\mid \lambda$

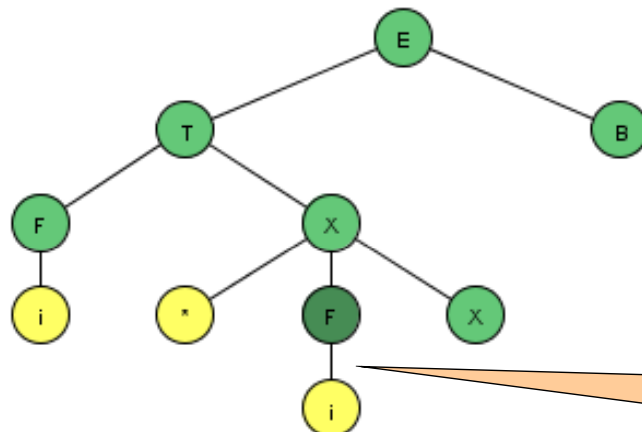
$T \rightarrow FX$

$X \rightarrow *FX$

$\mid \lambda$

$F \rightarrow i$

$\mid (E)$

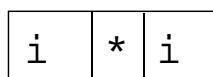


Se realiza
otra
derivación

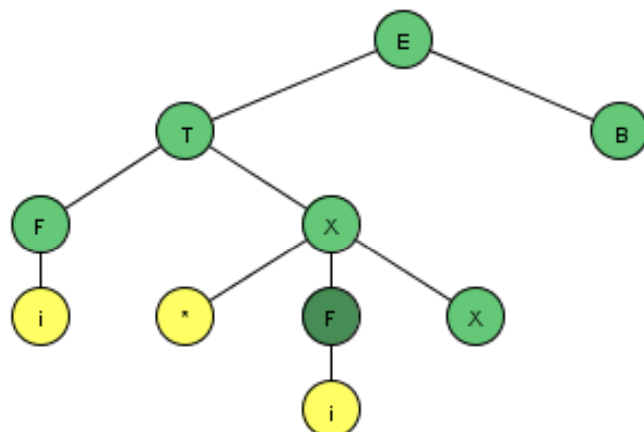
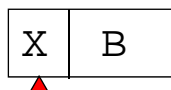
60

Ejemplo de analizador descendente muy sencillo

Entrada:



Pila:



Gramática:

$E \rightarrow TB$

$B \rightarrow +TB$

$\mid \lambda$

$T \rightarrow FX$

$X \rightarrow *FX$

$\mid \lambda$

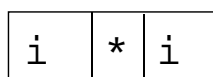
$F \rightarrow i$

$\mid (E)$

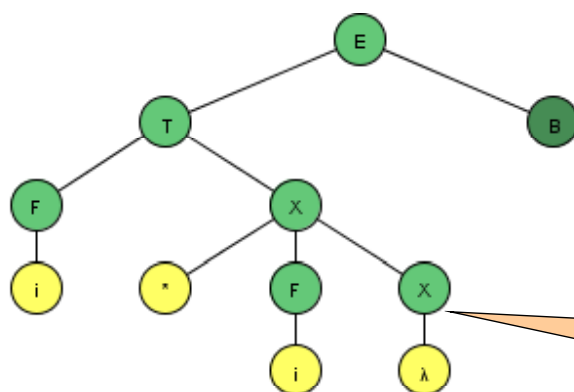
61

Ejemplo de analizador descendente muy sencillo

Entrada:



Pila:



Gramática:

$E \rightarrow TB$

$B \rightarrow +TB$

$\mid \lambda$

$T \rightarrow FX$

$X \rightarrow *FX$

$\mid \lambda$

$F \rightarrow i$

$\mid (E)$

Se realiza
otra
derivación

62

Ejemplo de analizador descendente muy sencillo

Entrada:

i	*	i
---	---	---



Aceptada

Pila:



Gramática:

E \rightarrow **T****B**

B \rightarrow **+****T****B**

	λ
--	-----------

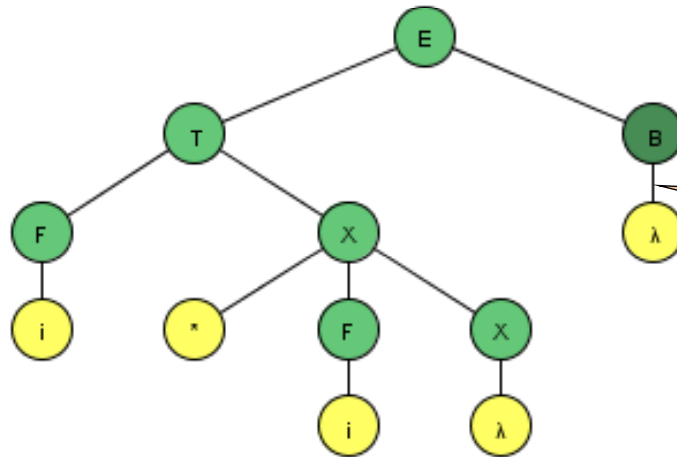
T \rightarrow **F****X**

X \rightarrow *******F****X**

λ

F \rightarrow **i**

λ (**E**)



Se realiza
otra
derivación