Tema 9

Análisis Semántico

Universidad Europea

LAUREATE INTERNATIONAL UNIVERSITIES

Gramáticas de Atributos

Prof. Leopoldo Santos Santos 2020

Análisis Semántico

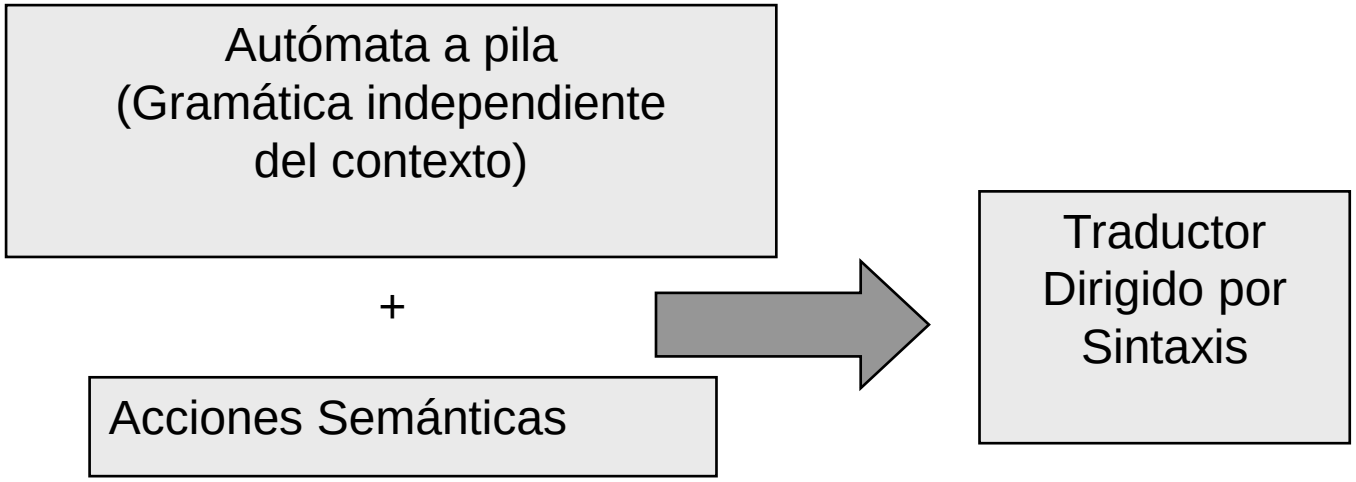
- Hay que extender el análisis sintáctico para llegar la **comprensión** del programa y comprobar que tiene sentido
- Las gramáticas independientes del contexto (G2) no son suficientes para realizar el análisis semántico. No basta comprobar que se deriva de la gramática para la especificación del lenguaje
- Es necesario definir un tipo de gramática más rica como las gramáticas de atributo
- Esta fase también modifica la tabla de símbolos y suele estar mezclada con la generación de código intermedio (traducción)

Análisis Semántico

- Comprobaciones adicionales (estáticas)
  - Comprobación de tipos
    - La aplicación de los operadores y operandos deben ser compatibles
  - Comprobaciones del flujo del control
    - Las proposiciones que hacen que se abandone el flujo del control de una construcción debe tranferirse a otro punto. (*break, exit ...*)
  - Comprobaciones de unicidad
    - Hay situaciones en los que un objeto solo puede definirse una vez exclusivamente. Las etiquetas de una sentencia case no deben repetirse, declaraciones de objetos,..
  - Comprobaciones relacionadas con nombres
    - El mismo nombre debe aparecer dos o más veces. Ej.: en Ada el nombre que aparece en un bloque puede aparecer al principio y final. Debe comprobarse que se utiliza el mismo el ambos sitios

El Analizador Semántico

- Traducción dirigida por sintaxis



Traducción dirigida por sintaxis

- Definición
  - Las gramáticas de atributo son gramáticas G2 a las que se añaden atributos y reglas de evaluación de atributos (funciones/reglas semánticas)
  - Cada atributo es una variable que representa una propiedad del símbolo X (terminal o no terminal)
    - Ej: X.Tipo, X.Valor, ...
    - Puede ser una cadena, número, tipo, posición de memoria, etc
- Reglas semánticas
  - Se asocian a las producciones sintácticas. Se definen en función de los atributos de los símbolos en la producción
  - Además existen condiciones semánticas que se ejecutan sobre estos atributos

Traducción dirigida por sintaxis

- Efectos de las acciones semánticas
  - Cálculo de valores de atributos
  - Guardar/Consultar información de la Tabla de Símbolos
  - Generación de código
  - Modificar variables globales
  - Notificación de mensajes de error
- Valores de atributos:
  - cada producción A=α se asocia con un conjunto de acciones semánticas representadas como una función:
    - X.atr=f (Y_1.atr, ..., Y_n.atr)
  - También se conocen estas acciones como “ecuaciones de atributos”

Ejemplo 1

Evaluación de expresiones

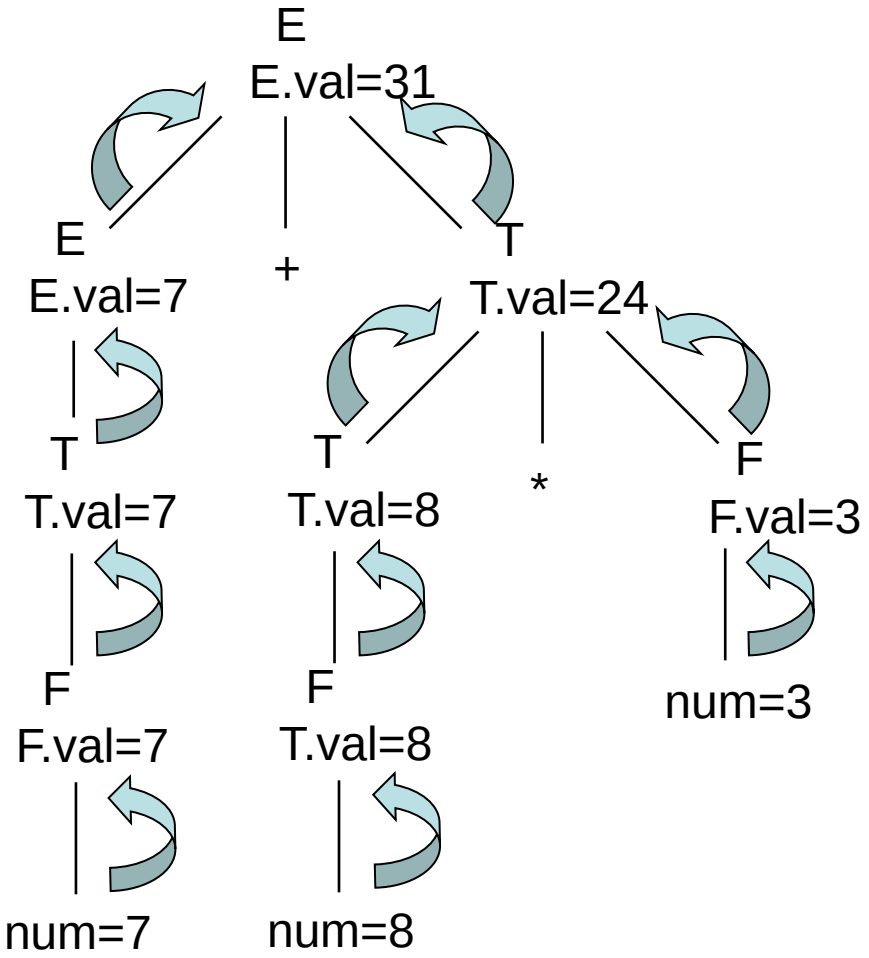
E::= E·+·T
E::= T
T::= T·*·F
T::= F
F::= num

1 atributo: valor

Acc. sem.: op aritm.

Sentencia: 7+8*3

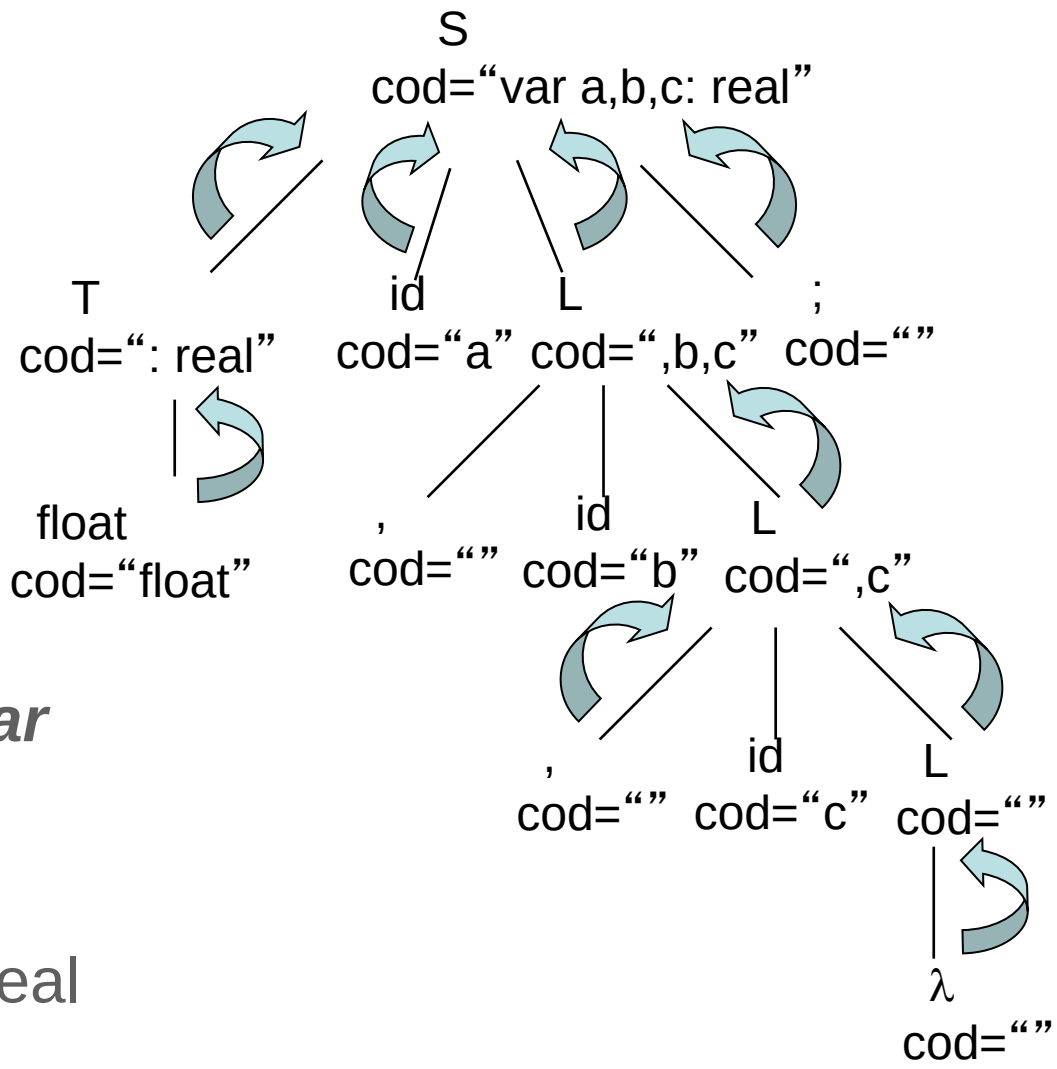
Resultado: 31



Ejemplo 2

Traductor C->Pascal

S::=T id L ;
L::= , id L | λ
T::= float | int



1 atributo: código

Acc. sem.: concatenar

Sentencia: float a,b,c;

Resultado: var a,b,c: real

Ejemplo 3

Traductor infija->postfija

E::=T E'
E' ::= op T {*escribe op.lex*} E'
E' ::=λ
T::= num {*escribe num.lex*}

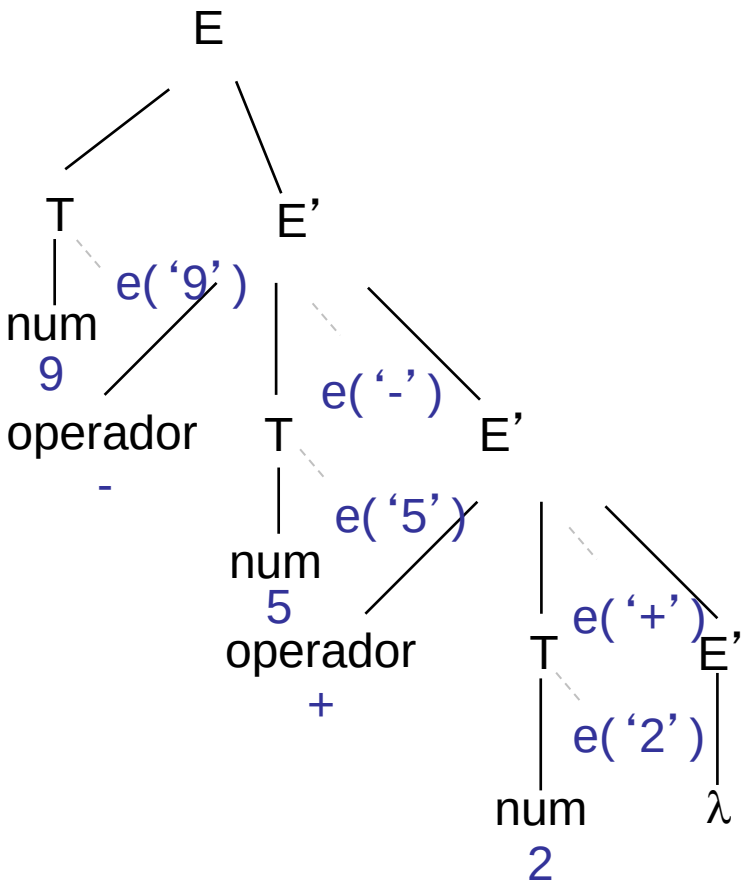
¡No hay atributos!

Acc. sem.: escribir código

(esquema de traducción)

Sentencia: 9-5+2

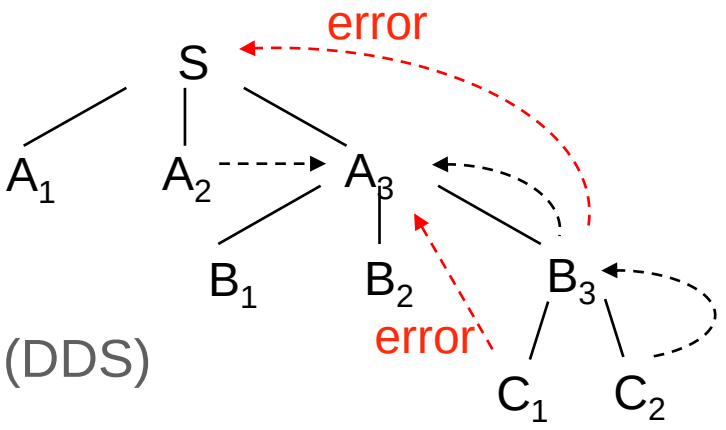
Resultado: 9 5 – 2 +



¿Traductor equivalente con atributos?

Gramáticas de Atributos

- Unión de gramática con atributos y reglas semánticas
  - Árboles “adornados”
  - Las relaciones entre atributos de cada acción semántica solo entre símbolos en la regla



- Notaciones
  - Definición dirigida por la sintaxis (DDS)
    - Acciones después de cada producción (sin orden especificado)
  - Esquema de Traducción (EDT)
    - Acciones después de cada símbolo. Notación para el traductor

Gramáticas de Atributos. DDS

- Ejemplo (calculadora):

producción	Acciones semánticas
E::= E·+·T	E_0 .val =E_1 .val +T.val
E::= T	E.val =T.val
T::= T·*·F	T_0 .val =T_1 .val *F.val
T::= F	T.val =F.val
F::= num	F.val =num.val

Gramáticas de Atributos. DDS

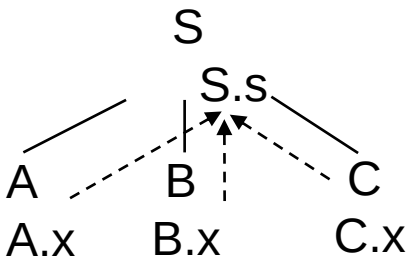
- Ejemplo (traductor C->Pascal):

producción	**Acciones semánticas**
S::= T id L ;	S.cod =“var ” + id.cod + “:” + L.cod
L::= , id L	L_0.cod = “, “ + id.cod+ L_1.cod
L::=λ	L.cod = “”
T::= float	T.cod = “real”
T::= int	T.cod = “int”

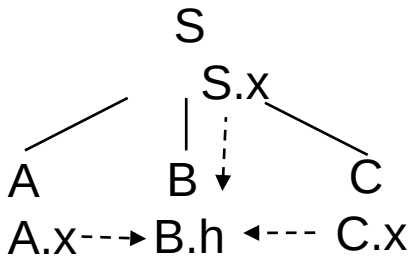
NOTA: + representa concatenar

Traducción dirigida por sintaxis

- Dos tipos de atributos
  - Sintetizados (locales)
    - El valor a asignar a un nodo depende del valor de los nodos hijos
  - Heredados
    - Se pasan a niveles inferiores del árbol. Su valor depende del valor de los hermanos y del padre



S.s=f(A.x, B.x, C.x)



B.h=g(A.x, S.x, C.x)

- El atributo mantiene el carácter en toda la gramática
- Los tokens sólo tienen atributos sintetizados

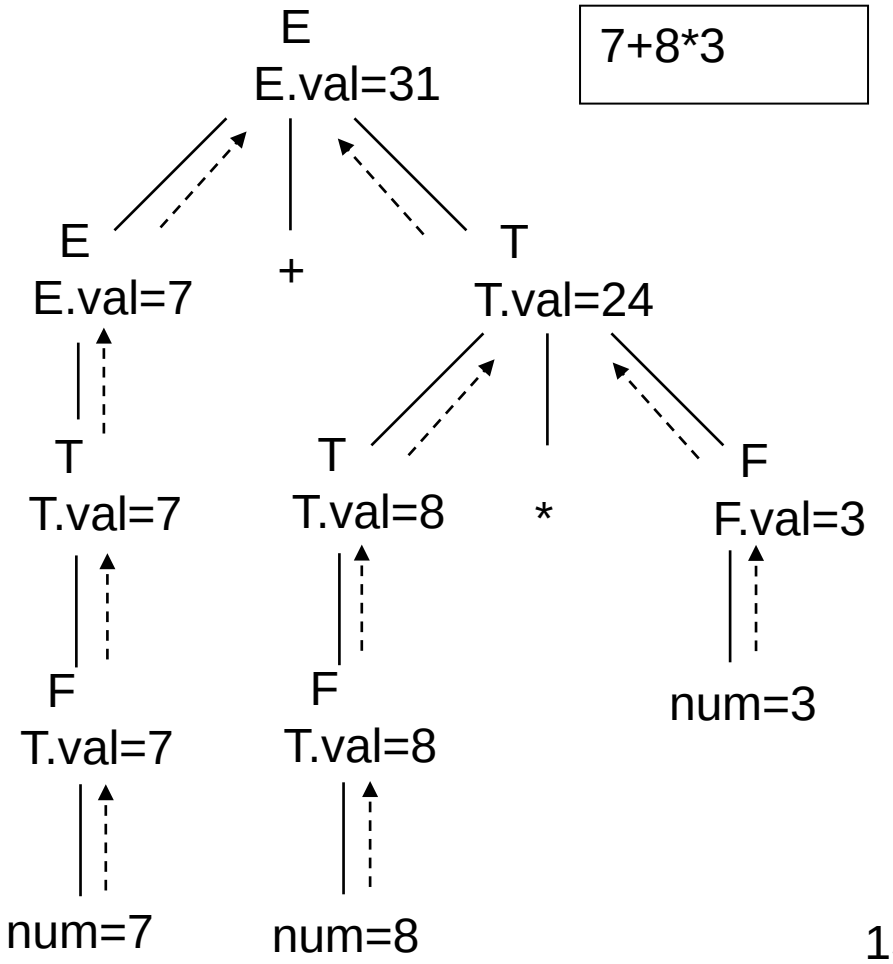
Traducción dirigida por sintaxis

- Ejemplo
  - Sintetizados, CALCULADORA, Análisis Ascendente

Producción	Reglas Semánticas
L→E **n**	*print* (*E.val*)
E→E + T	$E_0.val := E_{11}val + T.val$
E→T	*E.val* := *T.val*
T→T * F	$T_0.val := T_{1.}val * F.val$
T→F	*T.val* := *F.val*
F→(E)	*F.val* := *E.val*
F→**dígito**	*F.val* := ***dígito***.*valex*

Traducción dirigida por sintaxis

Ejemplo atributos sintetizados: flujo ascendente



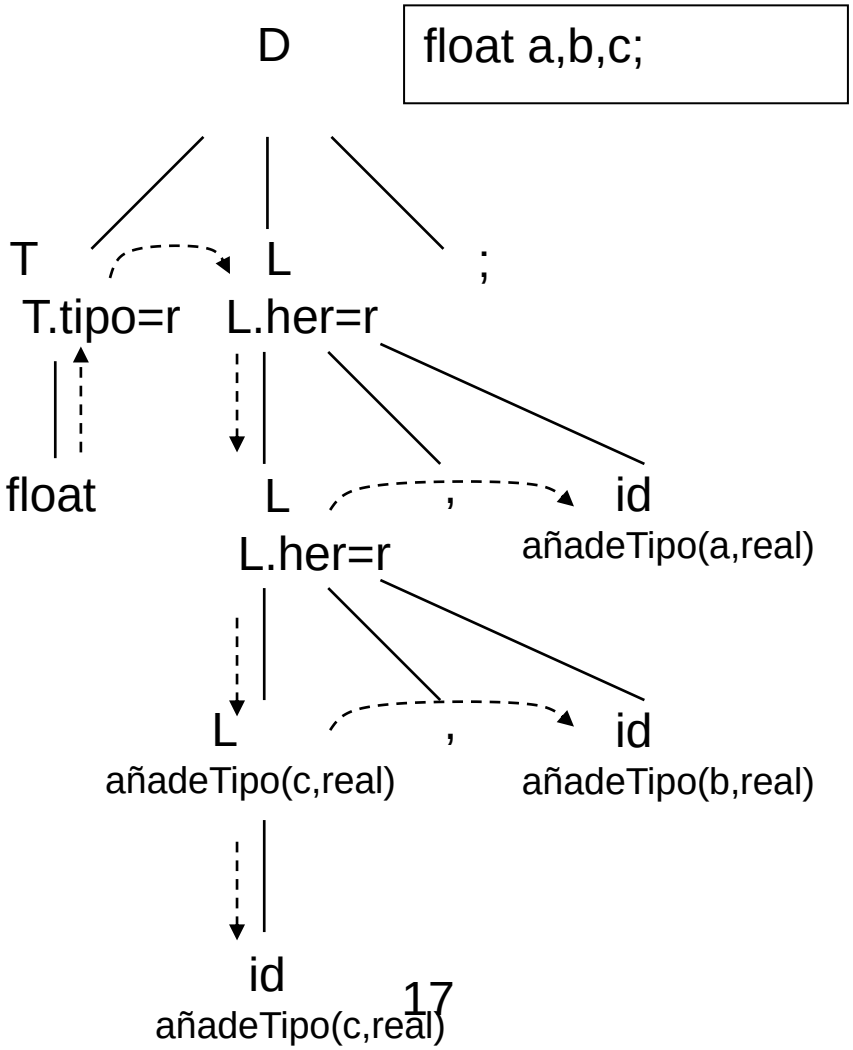
Traducción dirigida por sintaxis

- Ejemplo
  - Heredados, INFORMACIÓN DE TIPOS

Producción	Reglas Semánticas
D→T L	*L.her* := *T.tipo*
T→**int**	*T.tipo* := *integer*
T→**float**	*T.tipo* := *real*
L→L , **id**	L_1*.her* := L_0*.her*
	añadetipo (***id****.entrada, L.her*)
L→**id**	*añadetipo* (***id****.entrada, L.her*)

Traducción dirigida por sintaxis

Ejemplo con atributos heredados: flujo horizontal y descendente



Grafos de Dependencias

- Los atributos no pueden evaluarse en cualquier orden
  - Si un atributo *b* depende de un atributo *c*, entonces se debe evaluar la regla semántica para *b* después de la regla semántica que define a *c*
- Las interdependencias entre atributos heredados y sintetizados de un árbol de análisis sintáctico se pueden representar mediante un grafo dirigido llamado **Grafo de Dependencias**

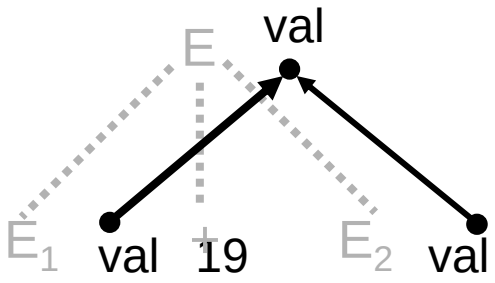
Grafo de Dependencias

- **Creación**: cada producción $A->X_1...X_n$ define una parte del grafo, se construye a partir de la cadena concreta

1. Se crea un nodo por cada atributo $X_i.a_j$ de cada símbolo de la producción
   - Los atributos heredados van a la izquierda y los sintetizados a la derecha
2. Para cada regla semántica $X_i.a_j=f(..., X_k.a_l,...)$ se hacen arcos desde cada nodo $X_k.a_l$ hacia el nodo $X_i.a_j$
   - Esto se repite para cada k y l afectados por la regla

- Ejemplo:

Producción	Regla Semántica
$E \rightarrow E+E$	$E_0.val:=E_1.val+E_2.val$



Evaluación de la gramática

- Evaluación de las reglas semánticas
  - Métodos con árbol de análisis sintáctico
    - Se realiza en el momento de compilación
    - El orden se obtiene de un ordenamiento topológico del grafo de dependencias construido según el árbol de análisis sintáctico para cada entrada
    - Si hay ciclos no funciona (gramáticas circulares)
  - Métodos de evaluación durante análisis sintáctico (hacen la evaluación en una pasada)
    - No necesita construir un grafo de dependencias de forma explícita
    - Hay métodos basados en reglas, las acciones semánticas asociadas con las producciones se analizan a mano
    - Hay métodos “sin recuerdo”: el orden de evaluación no tiene en cuenta las reglas semánticas

Esquema de Traducción (ETDS)

- Es una notación para hacer un traductor
- Las acciones semánticas se intercalan con los símbolos del consecuente de la producción
  - X::=ab {accion();} b
- Orden de evaluación fijo
- Dos tipos
  - EDT sólo con atributos sintetizados
    - Acciones al final de la producción
  - EDT con atributos sintetizados y heredados
    - Atributos heredados de un símbolo del consecuente
    - Atributos sintetizados utilizados en acciones
    - Atributos sintetizados del antecedente

Esquema de Traducción

- Restricciones sobre un ETDS
  - Un atributo heredado para un símbolo en el lado derecho de una producción se calculará en una acción antes que dicho símbolo
    - A->α_1 · B α_2 {B.h=f(...)}
  - Una acción no debe referirse a un atributo sintetizado de un símbolo que esté a la derecha de la acción
    - A->$\alpha_1\alpha_2$ {A.a= α_3.s} α_3
  - Un atributo sintetizado para el NO terminal de la izquierda solo puede calcularse después de que se hayan calculado todos los atributos a los que hace referencia. (La acción se sitúa al final del lado derecho de la producción)

Ejemplo ETDS

El siguiente ETD no cumple los criterios:

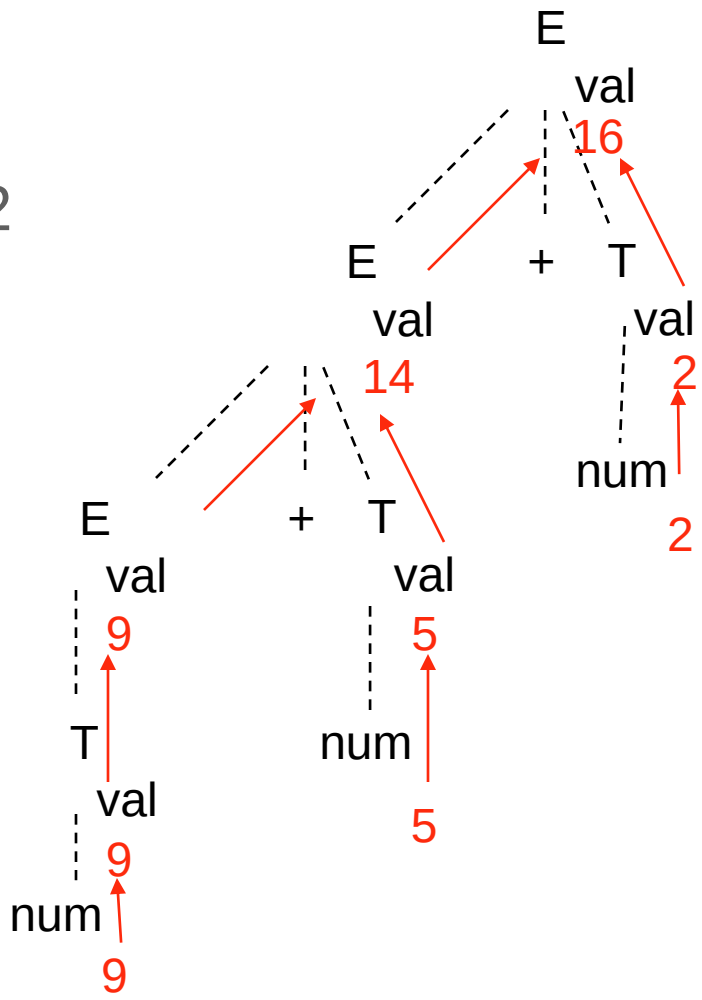
S::= AA	{A_1.her=1; A_2.her=2}
A::= a	{imprimir(A.her)}

Modificación:

S::=	{A_1.her=1;}
A	{A_2.her=1;}
A	
A::= a	{imprimir(A.her)}

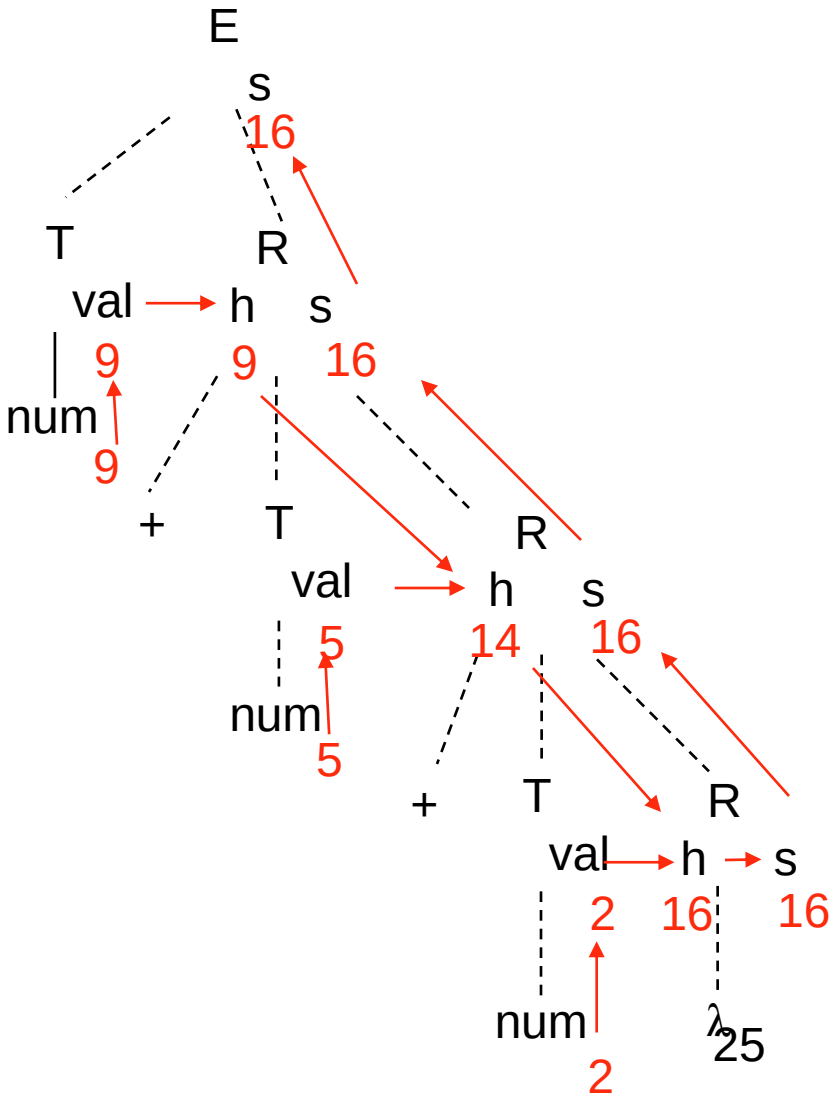
Gramática Original

9+5+2



Con Evaluación Descendente

9+5+2



Eliminación de acciones intercaladas en un ETDS

- Cuando no hay atributos heredados, la transformación es inmediata:
- $A \rightarrow X_1 \ldots X_{j-1}$ {accion} X_j ⇒ $\begin{cases} A \rightarrow X_1 \ldots X_{j-1} M X_j \\ M \rightarrow \lambda \ \{accion\} \end{cases}$

- Ej:

E::=T E'
E' ::=+ T {*escribe* +} E'
E' ::= - T {*escribe* -} E'
E' ::=λ
T::= num {*escribe num.lex*}

⇒

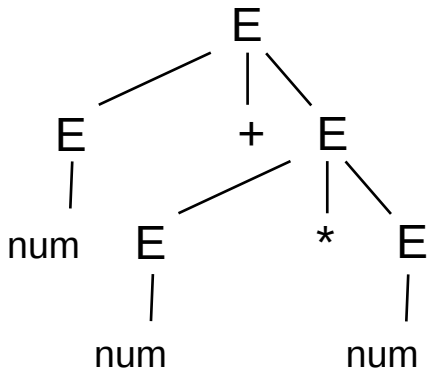
E::=T E'
E' ::= + T M E'
E' ::= - T N E'
E' ::=λ
T::= num {*escribe num.lex*}
M::=λ {*escribe* +}
N::=λ {*escribe* - }

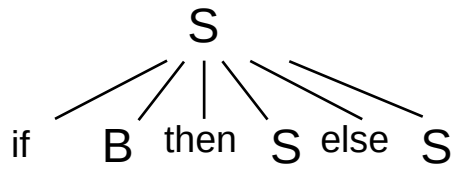
Creación del árbol sintáctico

- Si no se puede evaluar la gramática al tiempo que se hace el análisis sintáctico
- Evaluación en varias pasadas:
  - construir el árbol explícitamente (con una DDS)
  - evaluación en función del orden de recorrido (grafo de dependencias)
- Árbol sintáctico abstracto
  - Estructura de datos que condensa un árbol de análisis sintáctico
  - Los operadores y las palabras clave
    - No son hojas
    - Están asociadas con el nodo padre de dichas hojas

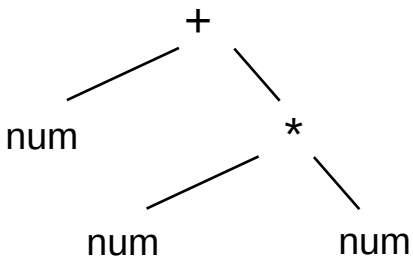
Creación del árbol sintáctico

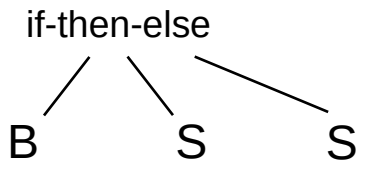
- Árboles sintácticos (parse tree)





- Árboles de sintaxis abstractos (sintax tree)





Creación del árbol sintáctico

- Construcción
  - Un nodo para cada operador y cada operando
  - Los hijos de un nodo operador son las raíces de los nodos que representan las subexpresiones que constituyen los operandos de dicho operador
- Funciones auxiliares
  - hazNodo (operador, izquierda, derecha)
  - hazHoja (id, entrada)
  - hazHoja (num, val)

Creación del árbol sintáctico

- DDS para construir árboles sintácticos
  - Definición con atributos sintetizados para construir un árbol sintáctico para una expresión con operadores + y –
- Utiliza
  - Atributo sintetizado apn conserva los apuntadores devueltos por la llamada a las funciones
  - HazNodo y HazHoja

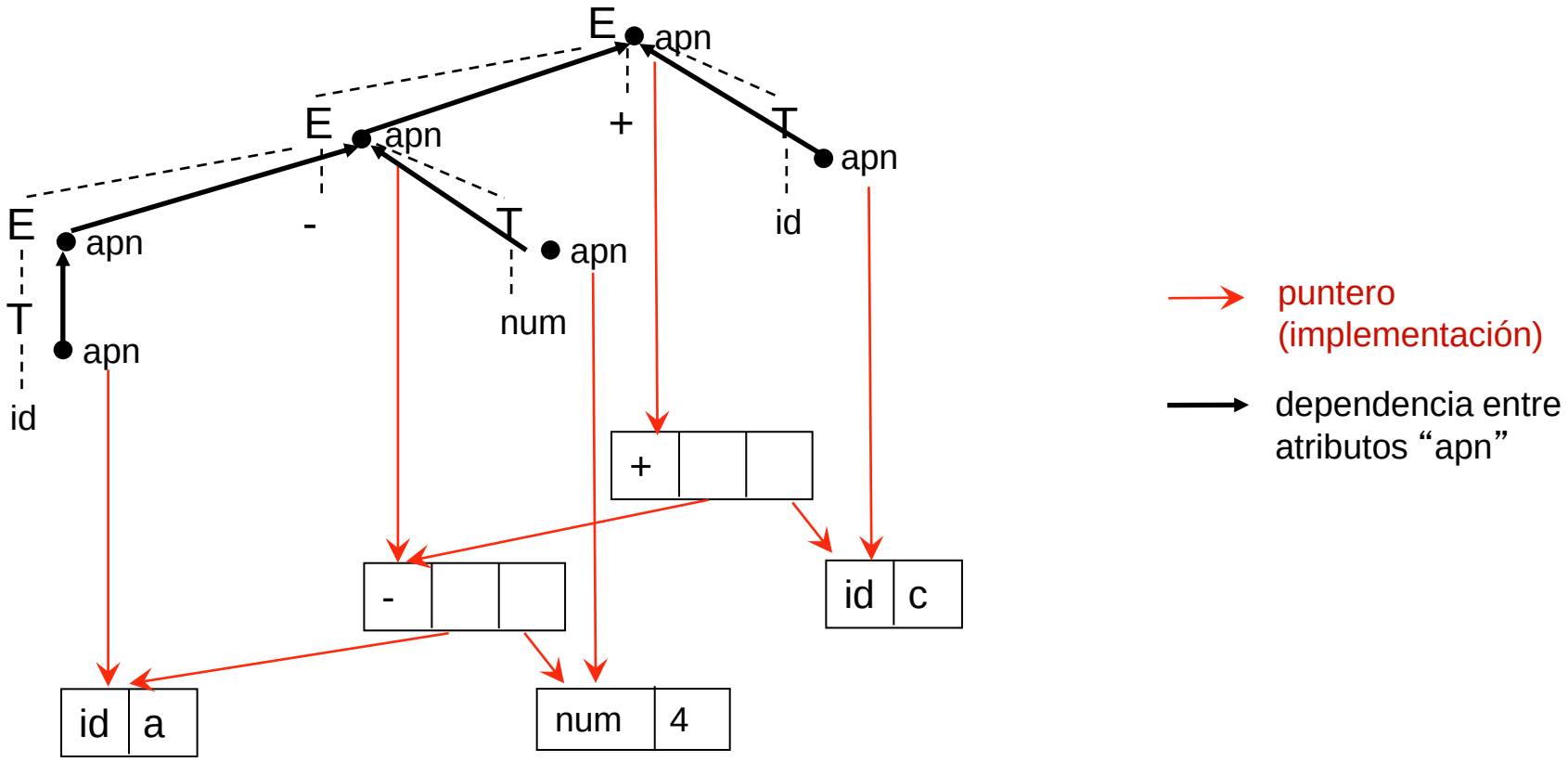
Creación del árbol sintáctico

- DDS para construir árboles sintácticos
  - Definición con atributos sintetizados para construir un árbol sintáctico para una expresión con operadores + y –
- Utiliza
  - Atributo sintetizado apn conserva los apuntadores devueltos por la llamada a las funciones
  - HazNodo y HazHoja

PRODUCCIÓN	REGLAS SEMÁNTICAS
$E \rightarrow E_1 + T$	E.apn := hazNodo ('+' , E_1.apn, T.apn)
$E \rightarrow E_1 - T$	E.apn := hazNodo ('-' , E_1.apn, T.apn)
$E \rightarrow T$	E.apn := T.apn
$T \rightarrow (E)$	T.apn := E.apn
$T \rightarrow id$	T.apn := hazHoja (id, id.entrada)
$T \rightarrow num$	T.apn := hazHoja (num, num.val)

Creación del árbol sintáctico

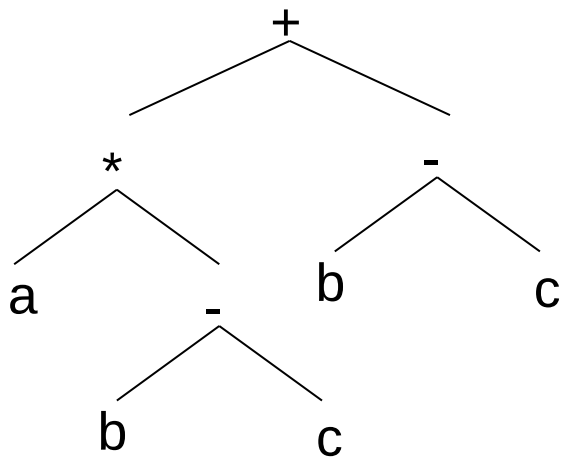
- Ej. (construcción ascendente): **a – 4 + c**



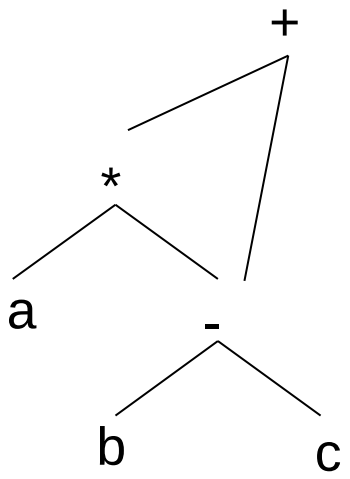
Grafos acíclicos para expresiones

- Grafos de expresiones
  - Se identifican subexpresiones comunes
  - Cada nueva ocurrencia de la subexpresión
    - No genera un nuevo subárbol
    - Referencia al subárbol ya existente o lo crea si no existe
- Ej.: a * (b – c) + (b –c)

árbol sintáctico



grafo acíclico



Orden de recorrido del árbol

- Un orden natural para los métodos de traducción descendente y ascendente es el “orden de evaluación en profundidad” (Ej.: gramáticas L-atribuidas)

```
procedimiento visitaprof (n:nodo)
    empezar
        para cada hijo m de n, de izquierda a derecha
            hacer empezar
                evaluar los atributos heredados de m;
                visitarprof (m)
            fin;
        evaluar los atributos sintetizados de n
    fin
```

- En general: orden según grafo de dependencias: evaluadores recursivos (gramáticas acíclicas)

Bibliografía

- Louden K.C. “Construcción de Compiladores. Principios y Práctica”. Thomson, 2004.

- Apuntes de la Universidad Carlos III de Madrid.