INGENIERÍA EN SISTEMAS

Creamos, Transformamos y Simplificamos

Traducción dirigida por la sintaxis

Ing. Alex Moncada

UNAH

UNIVERSIDAD NACIONAL
AUTÓNOMA DE HONDURAS

LU
CEM
ASPI
CIO

Traducción orientada a la sintaxis

La traducción orientada a la sintaxis se realiza uniendo reglas o fragmentos de un programa a las producciones en una gramática. Por ejemplo, considere una expresión *expr* generada por la siguiente producción:

$$expr \rightarrow expr_1 + term$$

Aquí, *expr* es la suma de las dos subexpresiones $expr_1$ y *term*. El subíndice en $expr_1$ se utiliza sólo para diferenciar la instancia de *expr* en el cuerpo de la producción, del encabezado de la producción. Podemos traducir *expr* explotando su estructura, como en el siguiente seudocódigo:

```
traducir expr1;
traducir term;
manejar +;
```

Traducción orientada a la sintaxis

Conceptos relacionados con la traducción orientada a la sintaxis:

- *Atributos*. Un atributo es cualquier cantidad asociada con una construcción de programación.

  Algunos ejemplos de atributos son los tipos de datos de las expresiones, el número de instrucciones en el código generado, o la ubicación de la primera instrucción en el código generado para una construcción, entre muchas otras posibilidades. Como utilizamos símbolos de la gramática (no terminales y terminales) para representar las construcciones de programación, extendemos la noción de los atributos, de las construcciones a los símbolos que las representan.

Traducción orientada a la sintaxis

- *Esquemas de traducción (orientada a la sintaxis).* Un esquema de traducción es una notación para unir los fragmentos de un programa a las producciones de una gramática. Los fragmentos del programa se ejecutan cuando se utiliza la producción durante el análisis sintáctico. El resultado combinado de todas estas ejecuciones de los fragmentos, en el orden inducido por el análisis sintáctico, produce la traducción del programa al cual se aplica este proceso de análisis/síntesis.

Notación postfija

La notación postfija para una expresión E puede definirse de manera inductiva, como se muestra a continuación:

1. Si E es una variable o constante, entonces la notación postfija para E es la misma E.
2. Si E es una expresión de la forma E_1 **op** E_2, en donde **op** es cualquier operador binario, entonces la notación postfija para E es E'_1 E'_2 **op**, en donde E'_1 E'_2 son las notaciones postfija para E_1 E_2 , respectivamente.
3. Si E es una expresión con paréntesis de la forma (E_1), entonces la notación postfija para E es la misma que la notación postfija para E_1.

Notación postfija

Ejemplo 2.8: La notación postfija para `(9-5)+2` es `95-2+`. Es decir, las traducciones de `9`, `5` y `2` son las mismas constantes, en base a la regla (1). Entonces, la traducción de `9-5` es `95-` en base a la regla (2). La traducción de `(9-5)` es la misma en base a la regla (3), Habiendo traducido la subexpresión con paréntesis, podemos aplicar la regla (2) a toda la expresión, con `(9-5)` en el papel de E_1 y `2` en el papel de E_2, para obtener el resultado `95-2+`.

Como otro ejemplo, la notación postfija para `9-(5+2)` es `952+-`. Es decir, primero se traduce `5+2` a `52+`, y esta expresión se convierte en el segundo argumento del signo negativo. □

No se necesitan paréntesis en la notación postfija, debido a que la posición y la *aridad* (número de argumentos) de los operadores sólo permiten una decodificación de una expresión postfija. El "truco" es explorar varias veces la cadena postfija partiendo de la izquierda, hasta encontrar un operador. Después, se busca a la izquierda el número apropiado de operandos, y se agrupa este operador con sus operandos. Se evalúa el operador con los operandos y se sustituyen por el resultado. Después se repite el proceso, continuando a la derecha y buscando otro operador.

Notación postfija

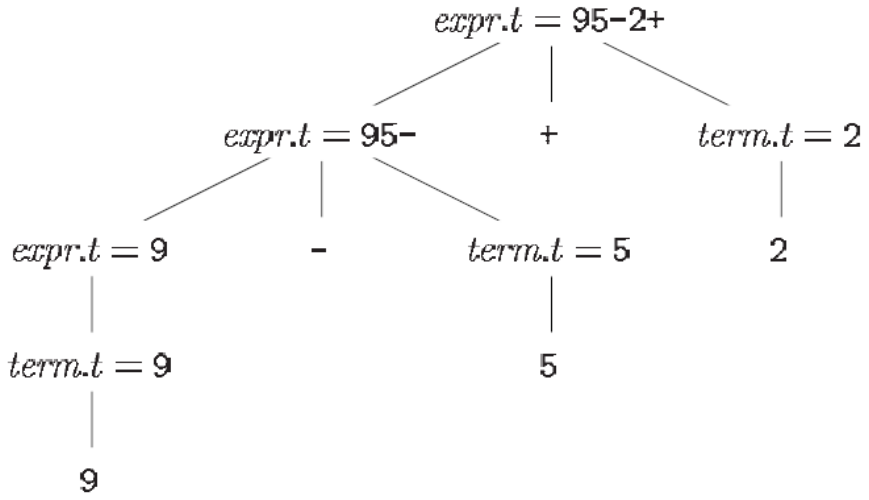

Figura 2.9: Valores de los atributos en los nodos de un árbol de análisis sintáctico

PRODUCCIÓN	REGLAS SEMÁNTICAS
$expr \rightarrow expr_1$ + $term$	$expr.t = expr_1.t \mid\mid term.t \mid\mid '+'$
$expr \rightarrow expr_1$ − $term$	$expr.t = expr_1.t \mid\mid term.t \mid\mid '-'$
$expr \rightarrow term$	$expr.t = term.t$
$term \rightarrow$ 0	$term.t = '0'$
$term \rightarrow$ 1	$term.t = '1'$
...	...
$term \rightarrow$ 9	$term.t = '9'$

Figura 2.10: Definición orientada a la sintaxis para la traducción de infija a postfija

Notación postfija

Ejemplo 2.9: Considere la expresión postfija `952+-3*`. Si exploramos partiendo de la izquierda, primero encontramos el signo positivo. Si buscamos a su izquierda encontramos los operandos `5` y `2`. Su suma, `7`, sustituye a `52+` y tenemos la cadena `97-3*`. Ahora, el operador de más a la izquierda es el signo negativo, y sus operandos son `9` y `7`. Si los sustituimos por el resultado de la resta nos queda `23*`. Por último, el signo de multiplicación se asigna a `2` y `3`, lo cual produce el resultado de `6`. □

Definiciones dirigidas por la sintaxis

- Una definición dirigida por la sintaxis es una gramática libre de contexto, junto con atributos y reglas.
- Los atributos sintetizados se asocian con los símbolos gramaticales y las reglas se asocian con las producciones. Si *X* es un símbolo y ***a*** es uno de sus atributos, entonces escribimos *X.a* para denotar el valor de ***a*** en el nodo específico de un árbol de análisis sintáctico, etiquetado como *X*.
- Si implementamos los nodos del árbol de análisis sintáctico mediante registros u objetos, entonces los atributos de *X* pueden implementarse mediante campos de datos en los registros, que representen los nodos para *X*. Los atributos pueden ser de cualquier tipo: por ejemplo, números, tipos, referencias de tablas o cadenas.

Atributos heredados y sintetizados

Vamos a manejar dos tipos de atributos para los no terminales:

1. Un *atributo sintetizado* para un no terminal *A* en un nodo *N* de un árbol sintáctico se define mediante una regla semántica asociada con la producción en N.

   Un atributo sintetizado en el nodo N se define sólo en términos de los valores de los atributos en el hijo de N, y en el mismo N.

2. Un *atributo heredado* para un no terminal B en el nodo N de un árbol de análisis sintáctico se define mediante una regla semántica asociada con la producción en el padre de N.

   Un atributo heredado en el nodo N se define sólo en términos de los valores de los atributos en el padre de N, en el mismo N y en sus hermanos.

Atributos heredados y sintetizados

Los terminales pueden tener atributos sintetizados, pero no atributos heredados. Los atributos para los terminales tienen valores léxicos que suministra el analizador léxico; no hay reglas semánticas en la misma definición dirigida por la sintaxis para calcular el valor de un atributo para un terminal.

	PRODUCCIÓN	REGLAS SEMÁNTICAS
1)	$L \rightarrow E\ \mathbf{n}$	$L.val = E.val$
2)	$E \rightarrow E_1\ +\ T$	$E.val = E_1.val + T.val$
3)	$E \rightarrow T$	$E.val = T.val$
4)	$T \rightarrow T_1\ *\ F$	$T.val = T_1.val \times F.val$
5)	$T \rightarrow F$	$T.val = F.val$
6)	$F \rightarrow (\ E\ )$	$F.val = E.val$
7)	$F \rightarrow \mathbf{digito}$	$F.val = \mathbf{digito}$.lexval

Figura 5.1: Definición orientada por la sintaxis de una calculadora de escritorio simple

Atributos heredados y sintetizados

A una definición dirigida por la sintaxis que sólo involucra atributos sintetizados se le conoce como definición dirigida por la sintaxis con *atributos sintetizados*; la definición dirigida por la sintaxis en la figura 5.1 tiene esta propiedad.

A una definición dirigida por la sintaxis sin efectos adicionales se le llama algunas veces *gramática atribuida*. Las reglas en una gramática atribuida definen el valor de un atributo, sólo en términos de los valores de otros atributos y constantes.

Evaluación de una definición dirigida por la sintaxis en los nodos de un árbol de análisis sintáctico

Para visualizar la traducción especificada por una definición dirigida por la sintaxis, es útil trabajar con los árboles de análisis sintáctico.

A un árbol sintáctico, que muestra el (los) valor(es) de su(s) atributo(s) se le conoce como *árbol de análisis sintáctico anotado*.

Para las definiciones dirigidas por la sintaxis con atributos heredados y sintetizados, no hay garantía de que haya siquiera un orden en el que se puedan evaluar los atributos en los nodos. Por ejemplo, considere los no terminales A y B, con los atributos sintetizados y heredados A.s y B.i, respectivamente, junto con la siguiente producción y las siguientes reglas:

PRODUCCIÓN	REGLAS SEMÁNTICAS
$A \rightarrow B$	$A.s = B.i;$
	$B.i = A.s + 1$

Evaluación de una definición dirigida por la sintaxis en los nodos de un árbol de análisis sintáctico

Estas reglas son circulares; es imposible evaluar A.s en un nodo N o B.i como el hijo de N sin primero evaluar el otro. La dependencia circular de A.s y B.i en cierto par de nodos en un árbol sintáctico se sugiere mediante la figura.

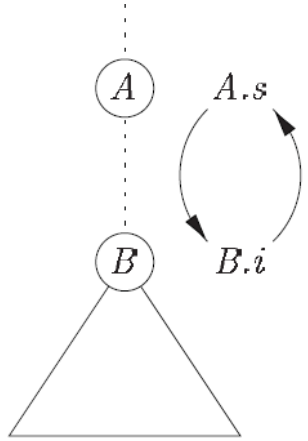

Figura 5.2: La dependencia circular de $A.s$ y $B.i$, uno del otro

En términos computacionales, es difícil determinar si existen o no circularidades en cualquiera de los árboles de análisis sintáctico que tenga que traducir una definición dirigida por la sintaxis dada

Evaluación de una definición dirigida por la sintaxis en los nodos de un árbol de análisis sintáctico

Ejemplo 5.2: La figura 5.3 muestra un árbol de análisis sintáctico anotado para la cadena de entrada 3 ∗ 5 + 4 n, construida mediante el uso de la gramática y las reglas de la figura 5.1.

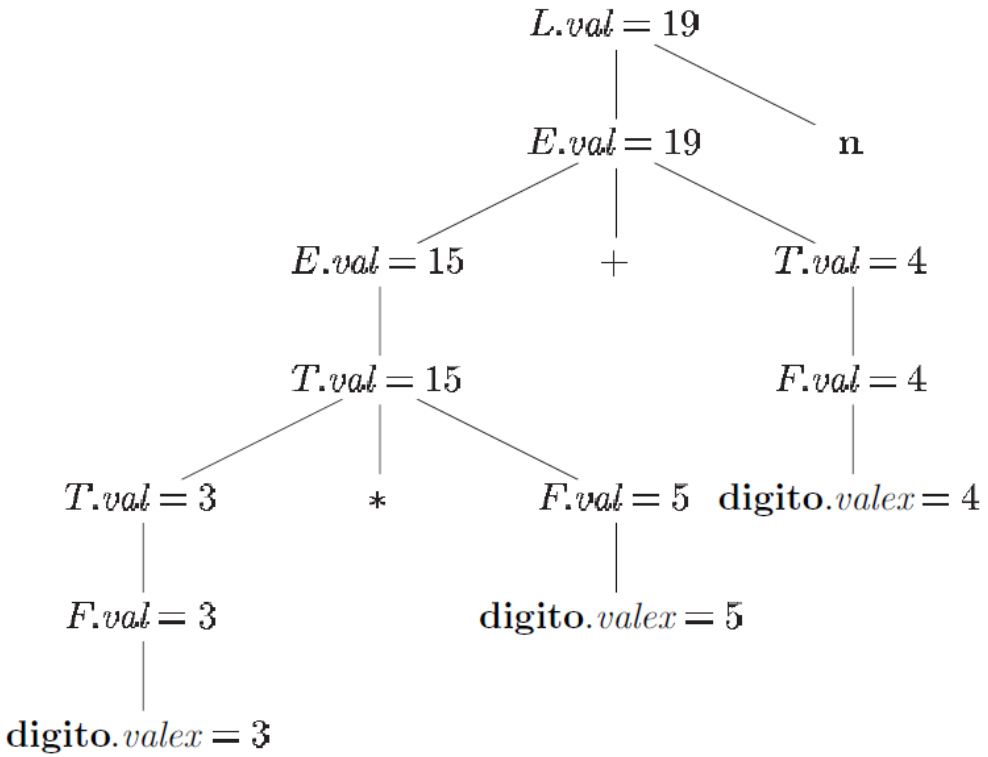

Figura 5.3: Árbol de análisis sintáctico anotado para 3 ∗ 5 + 4 **n**

Evaluación de una definición dirigida por la sintaxis en los nodos de un árbol de análisis sintáctico

Ejemplo 5.3: La definición dirigida por la sintaxis en la figura 5.4 calcula términos como $3 * 5$

Realice el árbol de análisis sintáctico.

	PRODUCCIÓN	REGLAS SEMÁNTICAS
1)	$T \rightarrow F\, T'$	$T'.her = F.val$ $T.val = T'.sin$
2)	$T' \rightarrow *\, F\, T'_1$	$T'_1.her = T'.her \times F.val$ $T'.sin = T'_1.sin$
3)	$T' \rightarrow \epsilon$	$T'.sin = T'.her$
4)	$F \rightarrow \mathbf{digito}$	$F.val = \mathbf{digito}.valex$

Figura 5.4: Una definición dirigida por la sintaxis basada en una gramática adecuada para el análisis sintáctico de descendente

Evaluación de una definición dirigida por la sintaxis en los nodos de un árbol de análisis sintáctico

Ejemplo 5.3: La definición dirigida por la sintaxis en la figura 5.4 calcula términos como $3 * 5$

Realice el árbol de análisis sintáctico.

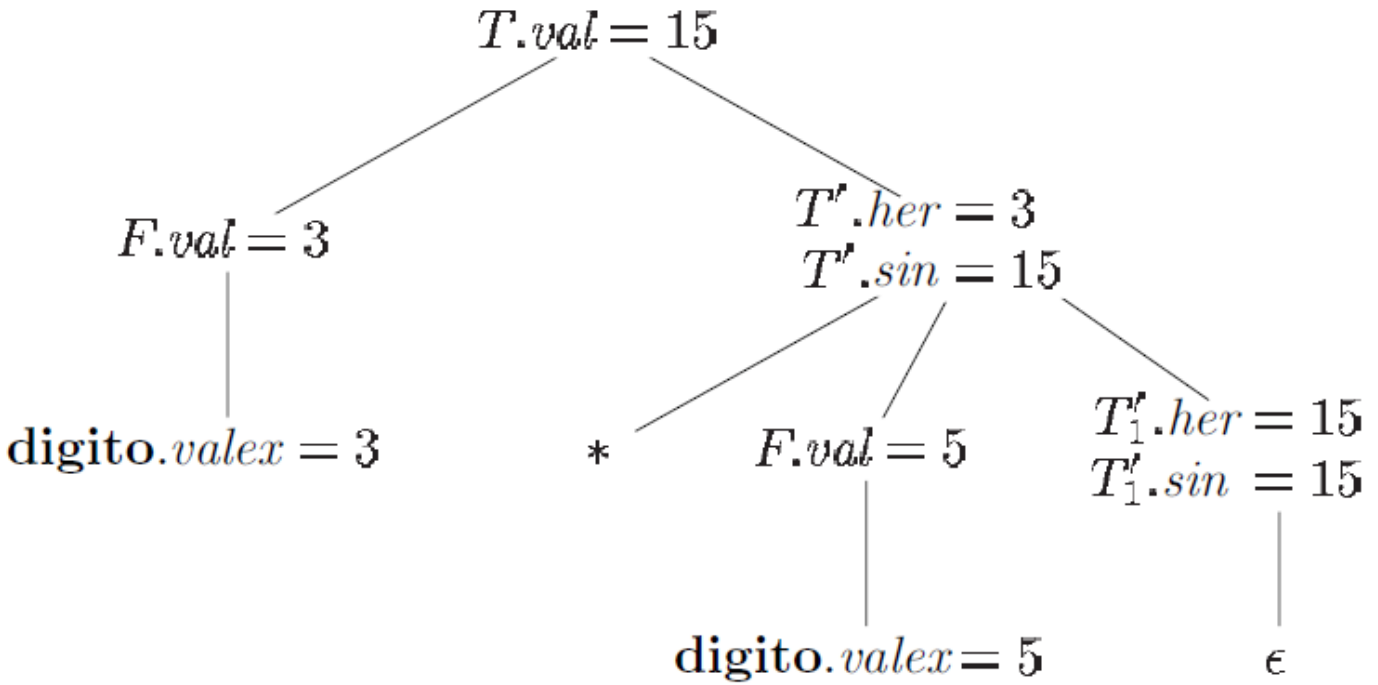

Figura 5.5: Árbol de análisis sintáctico anotado para $3 * 5$

Ejercicio

Proporcione árboles de análisis sintáctico anotado para la siguiente expresión: (3 + 4) ∗ (5 + 6) n utilizando la definición dirigida por la sintaxis de la figura 5.1.

Construcción de árboles de análisis sintáctico

Un nodo de árbol sintáctico que representa a una expresión $E_1 + E_2$ tiene la etiqueta + y dos hijos que representan las subexpresiones E_1 y E_2.

Vamos a implementar los nodos de un árbol sintáctico mediante objetos con un número adecuado de campos. Cada objeto tendrá un campo *op* que es la etiqueta del nodo. Los objetos tendrán los siguientes campos adicionales:

- Si el nodo es una hoja, un campo adicional contiene el valor léxico para esa hoja. Un constructor *Hoja(op, val)* crea un objeto hoja. De manera alternativa, si los nodos se ven como registros, entonces Hoja devuelve un apuntador a un nuevo registro para una hoja.
- Si el nodo es interior, hay tantos campos adicionales como hijos que tiene el nodo en el árbol sintáctico. Un constructor Nodo recibe dos o más argumentos: *Nodo(op, c1, c2, …, ck)* crea un objeto con el primer campo *op* y *k* campos adicionales para los *k* hijos *c1, …, ck*.

Ejemplo 5.11

	PRODUCCIÓN	REGLAS SEMÁNTICAS
1)	$E \rightarrow E_1 + T$	$E.nodo =$ **new** $Nodo('+', E_1.nodo, T.nodo)$
2)	$E \rightarrow E_1 - T$	$E.nodo =$ **new** $Nodo('-', E_1.nodo, T.nodo)$
3)	$E \rightarrow T$	$E.nodo = T.nodo$
4)	$T \rightarrow (E)$	$T.nodo = E.nodo$
5)	$T \rightarrow$ **id**	$T.nodo =$ **new** $Hoja($**id**, **id**$.entrada)$
6)	$T \rightarrow$ **num**	$T.nodo =$ **new** $Hoja($**num**, **num**$.val)$

Figura 5.10: Construcción de árboles sintácticos para expresiones simples

Ejemplo 5.11

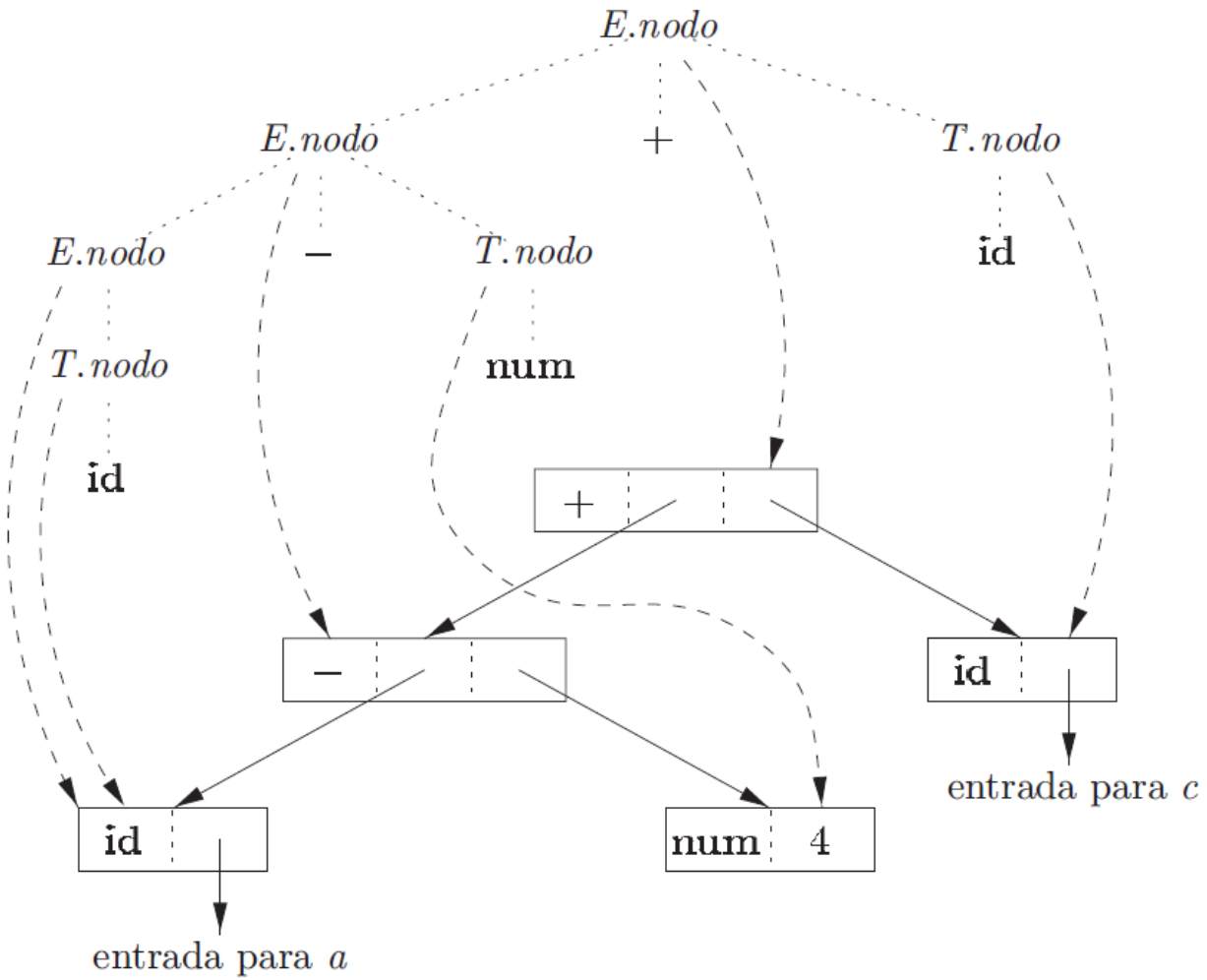

Figura 5.11: Árbol sintáctico para $a - 4 + c$

Ejemplo 5.11

1) $p_1 = \textbf{new}\ \mathit{Hoja}(\textbf{id},\ \mathit{entrada\text{-}a});$
2) $p_2 = \textbf{new}\ \mathit{Hoja}(\textbf{num},\ 4);$
3) $p_3 = \textbf{new}\ \mathit{Nodo}('-',\ p_1,\ p_2);$
4) $p_4 = \textbf{new}\ \mathit{Hoja}(\textbf{id},\ \mathit{entrada\text{-}c});$
5) $p_5 = \textbf{new}\ \mathit{Nodo}('+',\ p_3,\ p_4);$

Figura 5.12: Pasos en la construcción del árbol sintáctico para $a - 4 + c$

Ejemplo 5.12

	PRODUCCIÓN	REGLAS SEMÁNTICAS
1)	$E \rightarrow T\ E'$	$E.nodo = E'.sin$ $E'.her = T.nodo$
2)	$E' \rightarrow +\ T\ E'_1$	$E'_1.her =$ **new** $Nodo('+', E'.her, T.nodo)$ $E'.sin = E'_1.sin$
3)	$E' \rightarrow -\ T\ E'_1$	$E'_1.her =$ **new** $nodo('-', E'.her, T.nodo)$ $E'.sin = E'_1.sin$
4)	$E' \rightarrow \epsilon$	$E'.sin = E'.her$
5)	$T \rightarrow (\ E\ )$	$T.nodo = E.nodo$
6)	$T \rightarrow$ **id**	$T.nodo =$ **new** $Hoja($**id**, **id**$.entrada)$
7)	$T \rightarrow$ **num**	$T.nodo =$ **new** $Hoja($**num**, **num**$.entrada)$

Figura 5.13: Construcción de árboles sintácticos durante el análisis sintáctico descendente

Ejemplo 5.12

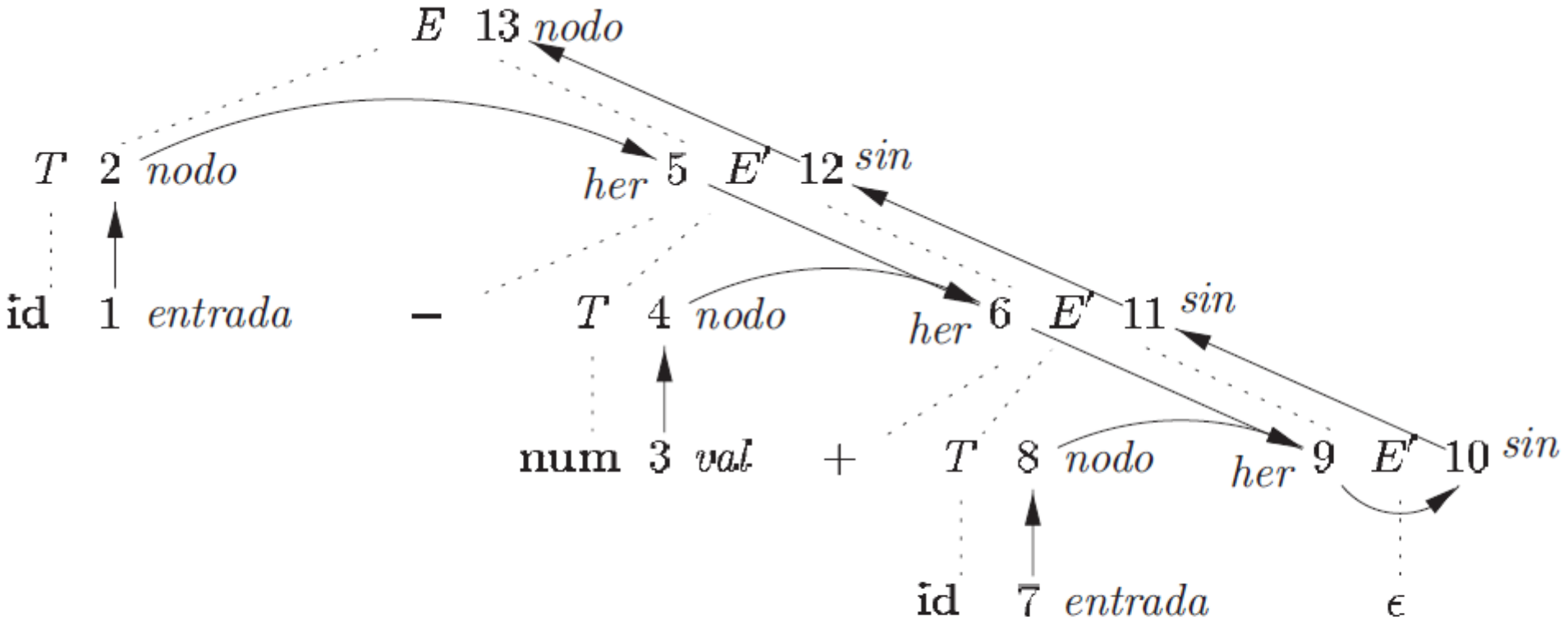

Figura 5.14: Gráfico de dependencias para $a - 4 + c$, con la definición dirigida por la sintaxis de la figura 5.13

La estructura de tipos

Los atributos heredados son útiles cuando la estructura del árbol de análisis sintáctico difiere de la sintaxis abstracta de la entrada; entonces, los atributos pueden usarse para llevar información de una parte del árbol de análisis sintáctico a otra.

El siguiente ejemplo muestra cómo un conflicto en la estructura puede deberse al diseño del lenguaje, y no a las restricciones que impone el método de análisis sintáctico.

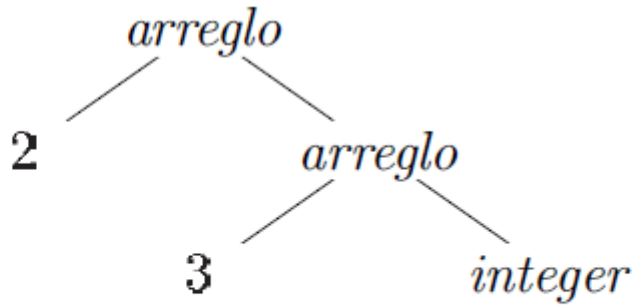

Figura 5.15: Expresión de los tipos para **int**[2][3]

La estructura de tipos

Ejemplo 5.13: En C, el tipo int [2][3] puede leerse como “arreglo de 2 arreglos de 3 enteros”. La expresión del tipo correspondiente arreglo(2, arreglo(3, integer)) se representa mediante el árbol en la figura 5.15. El operador arreglo recibe dos parámetros, un número y un tipo. Si los tipos se representan mediante árboles, entonces este operador devuelve un nodo de árbol etiquetado como arreglo, con dos hijos para el número y el tipo.

PRODUCCIÓN	REGLAS SEMÁNTICAS
$T \rightarrow B\ C$	$T.t = C.t$
	$C.b = B.t$
$B \rightarrow$ **int**	$B.t = integer$
$B \rightarrow$ **float**	$B.t = float$
$C \rightarrow$ [**num**] C_1	$C.t = arreglo($ **num**$.val,\ C_1.t)$
	$C_1.b = C.b$
$C \rightarrow \epsilon$	$C.t = C.b$

Figura 5.16: T genera un tipo básico o un tipo de arreglo

La estructura de tipos

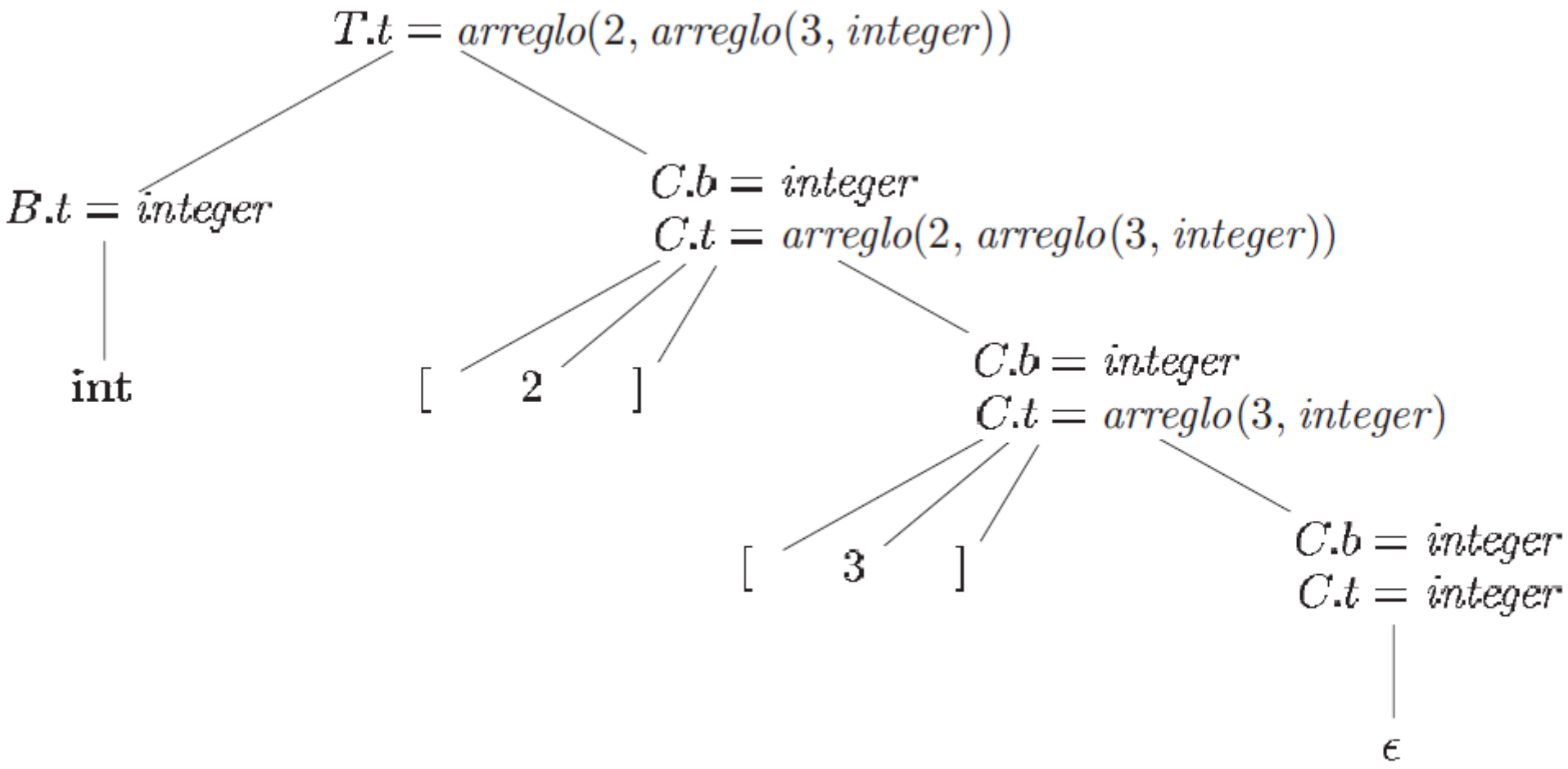

Figura 5.17: Traducción orientada por la sintaxis de los tipos de arreglos