Compiladores: Práctica 3

Roberto Ocampo Villegas

23 de Septiembre de 2024

Determinación de los conjuntos N, Σ y el símbolo inicial S

Dada la gramática $G = (N, \Sigma, P, S)$, con las producciones P:

- programa → declaraciones sentencias
- declaraciones → declaraciones declaracion | declaracion
- declaracion → tipo lista-var ;
- tipo → int | float
- lista-var → lista-var , identificador | identificador
- sentencias → sentencias sentencia | sentencia
- sentencia → identificador = expresion ; | if (expresion) sentencias else sentencias | while (expresion) sentencias
- expresion → expresion + expresion | expresion - expresion | expresion * expresion | expresion / expresion | identificador | numero
- expresion → (expresion)

- **Conjunto de no terminales** N:

$$N = \{\text{programa}, \text{declaraciones}, \text{declaracion}, \text{tipo}, \text{lista-var}, \text{sentencias}, \text{sentencia}, \text{expresion}\}$$

- **Conjunto de terminales** Σ:

$$\Sigma = \{\text{int}, \text{float}, \text{identificador}, ;, ,, =, \text{if}, \text{else}, \text{while}, (,), +, -, *, /, \text{numero}\}$$

- **Símbolo inicial** S:

$$S = \text{programa}$$

Eliminación de ambigüedad

Podemos ver que en la gramática dada hay una ambigüedad en la producción de expresiones:

expresion → expresion + expresion | expresion - expresion | expresion * expresion | expresion / expresion | identificador | numero

Esta ambigüedad ocurre debido a que no se especifica el orden de las operaciones, es decir, las operaciones de suma, resta, multiplicación y división se pueden interpretar de distintas formas.
Para resolver esta ambigüedad, reorganizamos las producciones de las expresiones para reflejar la precedencia y la asociatividad:

Producciones actualizadas para las operaciones aritméticas

$$
\begin{aligned}
\text{expresion} &\rightarrow \text{expresion\_suma} \\
\text{expresion\_suma} &\rightarrow \text{expresion\_suma} \ + \ \text{expresion\_mult} \\
&\mid \text{expresion\_suma} \ - \ \text{expresion\_mult} \\
&\mid \text{expresion\_mult} \\
\text{expresion\_mult} &\rightarrow \text{expresion\_mult} \ * \ \text{termino} \\
&\mid \text{expresion\_mult} \ / \ \text{termino} \\
&\mid \text{termino} \\
\text{termino} &\rightarrow \text{identificador} \\
&\mid \text{numero} \\
&\mid (\text{expresion})
\end{aligned}
$$

Gramática sin ambigüedad

$$
\begin{aligned}
\text{programa} &\rightarrow \text{declaraciones sentencias} \\
\text{declaraciones} &\rightarrow \text{declaraciones declaracion} \\
&\mid \text{declaracion} \\
\text{declaracion} &\rightarrow \text{tipo lista-var ;} \\
\text{tipo} &\rightarrow \text{int} \\
&\mid \text{float} \\
\text{lista-var} &\rightarrow \text{lista-var , identificador} \\
&\mid \text{identificador} \\
\text{sentencias} &\rightarrow \text{sentencias sentencia} \\
&\mid \text{sentencia} \\
\text{sentencia} &\rightarrow \text{identificador} = \text{expresion ;} \\
&\mid \text{if (expresion) sentencias else sentencias} \\
&\mid \text{while (expresion) sentencias} \\
\text{expresion} &\rightarrow \text{expresion\_suma} \\
\text{expresion\_suma} &\rightarrow \text{expresion\_suma} \ + \ \text{expresion\_mult} \\
&\mid \text{expresion\_suma} \ - \ \text{expresion\_mult} \\
&\mid \text{expresion\_mult} \\
\text{expresion\_mult} &\rightarrow \text{expresion\_mult} \ * \ \text{termino} \\
&\mid \text{expresion\_mult} \ / \ \text{termino} \\
&\mid \text{termino} \\
\text{termino} &\rightarrow \text{identificador} \\
&\mid \text{numero} \\
&\mid (\text{expresion})
\end{aligned}
$$

Eliminación de recursividad izquierda

En la gramática dada, podemos identificar casos de recursividad izquierda en las producciones de *declaraciones*, *lista-var*, *sentencias*, *expresion_suma* y *expresion_mult*. Utilizando el algoritmo visto en clase para eliminar la recursivdad, hacemos la actualización.

Gramática actualizada sin recursividad izquierda

$$
\begin{aligned}
\text{programa} &\rightarrow \text{declaraciones sentencias} \\
\text{declaraciones} &\rightarrow \text{declaracion declaraciones'} \\
\text{declaraciones'} &\rightarrow \text{declaracion declaraciones'} \mid \varepsilon \\
\text{declaracion} &\rightarrow \text{tipo lista-var ;} \\
\text{tipo} &\rightarrow \text{int} \mid \text{float} \\
\text{lista-var} &\rightarrow \text{identificador lista-var'} \\
\text{lista-var'} &\rightarrow , \text{ identificador lista-var'} \mid \varepsilon \\
\text{sentencias} &\rightarrow \text{sentencia sentencias'} \\
\text{sentencias'} &\rightarrow \text{sentencia sentencias'} \mid \varepsilon \\
\text{sentencia} &\rightarrow \text{identificador} = \text{expresion ;} \mid \text{if (expresion) sentencias else sentencias} \mid \text{while (expresion) sentencias} \\
\text{expresion} &\rightarrow \text{expresion\_suma} \\
\text{expresion\_suma} &\rightarrow \text{expresion\_mult expresion\_suma'} \\
\text{expresion\_suma'} &\rightarrow + \text{ expresion\_mult expresion\_suma'} \mid - \text{ expresion\_mult expresion\_suma'} \mid \varepsilon \\
\text{expresion\_mult} &\rightarrow \text{termino expresion\_mult'} \\
\text{expresion\_mult'} &\rightarrow * \text{ termino expresion\_mult'} \mid / \text{ termino expresion\_mult'} \mid \varepsilon \\
\text{termino} &\rightarrow \text{identificador} \mid \text{numero} \mid \text{(expresion)}
\end{aligned}
$$

Proceso de factorización izquierda

No es necesario realizar el proceso de factorización izquierda, ya que la gramática no presenta factores comunes al inicio de las producciones.

Nuevos conjuntos N y P

Después de los procesos de eliminación de ambigüedad, recursividad izquierda y factorización izquierda, los conjuntos actualizados son los siguientes:

Conjunto de no terminales (N)

$$
\begin{aligned}
N = \{&\text{programa}, \text{declaraciones}, \text{declaraciones'}, \text{declaracion}, \text{tipo}, \\
&\text{lista-var}, \text{lista-var'}, \text{sentencias}, \text{sentencias'}, \text{sentencia}, \\
&\text{expresion}, \text{expresion\_suma}, \text{expresion\_suma'}, \text{expresion\_mult}, \\
&\text{expresion\_mult'}, \text{termino}\}
\end{aligned}
$$

Conjunto de producciones (P)

$$
P = \begin{array}{l}
\text{programa} \rightarrow \text{declaraciones sentencias} \\
\text{declaraciones} \rightarrow \text{declaracion declaraciones'} \\
\text{declaraciones'} \rightarrow \text{declaracion declaraciones'} \mid \varepsilon \\
\text{declaracion} \rightarrow \text{tipo lista-var ;} \\
\text{tipo} \rightarrow \text{int} \mid \text{float} \\
\text{lista-var} \rightarrow \text{identificador lista-var'} \\
\text{lista-var'} \rightarrow, \text{ identificador lista-var'} \mid \varepsilon \\
\text{sentencias} \rightarrow \text{sentencia sentencias'} \\
\text{sentencias'} \rightarrow \text{sentencia sentencias'} \mid \varepsilon \\
\text{sentencia} \rightarrow \text{identificador} = \text{expresion ;} \mid \text{if (expresion) sentencias else sentencias} \mid \text{while (expresion) sentencias} \\
\text{expresion} \rightarrow \text{expresion\_suma} \\
\text{expresion\_suma} \rightarrow \text{expresion\_mult expresion\_suma'} \\
\text{expresion\_suma'} \rightarrow + \text{ expresion\_mult expresion\_suma'} \mid - \text{ expresion\_mult expresion\_suma'} \mid \varepsilon \\
\text{expresion\_mult} \rightarrow \text{termino expresion\_mult'} \\
\text{expresion\_mult'} \rightarrow * \text{ termino expresion\_mult'} \mid / \text{ termino expresion\_mult'} \mid \varepsilon \\
\text{termino} \rightarrow \text{identificador} \mid \text{numero} \mid \text{(expresion)}
\end{array}
$$