Tema 4 | Métodos de Análisis Sintáctico Descendentes

Contenidos

Bibliografía básica

20-Feb-2011

[Aho90] Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman
Compiladores. Principios, técnicas y herramientas. Addison-Wesley Iberoamericana 1990.

[Broo93] J.G. Brookshear
Teoría de la Computación. Lenguajes formales, autómatas y complejidad. Addison-Wesley Iberoamericana 1993.

4.1 Método intuitivo (fuerza bruta)

Se trata de un método con vuelta atrás.

Algoritmo del método intuitivo

1. Expandir la primera producción con el **símbolo inicial** de la gramática en su parte izquierda
2. Tomar el símbolo no terminal más a la izquierda *B* y aplicar el proceso de derivación con la primera producción en donde *B* aparezca en la parte izquierda de la producción.
3. Continuar el proceso (2) hasta que todos los símbolos sean terminales, pudiendo ocurrir:
   (a) Que alcancemos la secuencia a analizar.
   (b) Que no coincida con la secuencia a analizar.
4. En el caso (b) se buscará la subcadena de la subcadena alcanzada que no coincide con la que se desea analizar y se vuelve de nuevo a aplicar la siguiente producción con el mismo símbolo no terminal en la parte izquierda de la producción.

Problema: Cuando tenemos producciones definidas **recursivamente a la izquierda**.

Ejemplo: ***A*** → ***AcB***

4.2 Eliminación de la recursividad

Una **gramática es recursiva** a la izquierda si $\exists X \in N \,/\, X \Rightarrow_G^* X\alpha$

La recursividad a la izquierda puede **eliminarse** aplicando las siguiente transformación:

$$\boxed{\begin{array}{l} A \to A\alpha \\ A \to \beta \end{array}} \Longrightarrow \boxed{\begin{array}{l} A \to \beta A' \\ A' \to \alpha A' \\ A' \to \varepsilon \end{array}}$$

Ejemplo 4.1: Sea la gramática con recursividad a la izquierda definida por pos las producciones siguientes:

$$\begin{array}{l} E \to E + T \mid T \\ T \to T * F \mid F \\ F \to (E) \mid id \end{array}$$

Se puede apreciar que existe recursividad a la izquierda en los símbolos E y T. La gramática equivalente sin recursividad a la izquierda sería:

$$\begin{array}{l} E \to TE' \\ E' \to +TE' \mid \varepsilon \\ T \to FT' \\ T' \to *FT' \mid \varepsilon \\ F \to (E) \mid id \end{array}$$

4.3 Método descendente recursivo

Variante del método de la fuerza bruta. Consiste en considerar los símbolos no terminales como procedimientos.

Fundamento: Para cada símbolo no terminal ***A*** existe un procedimiento que recoge todas las producciones con ***A*** en la parte izquierda y genera los símbolos terminales que aparecen en la parte derecha de la producción y cuando alcanza a un símbolo no terminal se genera una llamada al procedimiento que lo define.

Problema: *General de los métodos de la fuerza bruta: Demasiados retrocesos (vuelta atrás).*

4.4 Analizadores predictivos

Son una variante de los anteriores evitando los múltiples retrocesos (vueltas atrás).

Requisitos:

- **No recursiva a la izquierda**: Que no sea recursiva a la izquierda. (problema solucionado en la sección 4.2)
- **Factorizada**: Con tan solo leer un símbolo de la entrada podamos decidir qué producción es la adecuada para aplicar la derivación (nuevo problema)

Problema: Que existan más de una producción cuyos símbolos de la parte derecha sean coincidentes a la izquierda.

La **factorización a la izquierda** consiste en sustituir aquellas producciones que tengan el citado problema por otras preservando la equivalencia:

$$A \rightarrow \alpha\beta_1 \mid \alpha\beta_2 \quad \Rightarrow \quad \begin{array}{l} A \rightarrow \alpha A' \\ A' \rightarrow \beta_1 \mid \beta_2 \end{array}$$

EN PRINCIPIO, TODA GRAMÁTICA FACTORIZADA Y NO RECURSIVA A LA IZQUIERDA PODRÍA SER RECONOCIDA POR EL MÉTODO DESCENDENTE PREDICTIVO RECURSIVO.

Procesadores de Lenguajes
Tema 4. Métodos de Análisis Sintáctico Descendentes
Grado en Ingeniería Informática
4º Curso / 1º Cuatrimestre

4.4 Analizadores predictivos

Se puede factorizar la gramática:

D → M b A | c D F | d

M → c M | ε

A → M B j | F

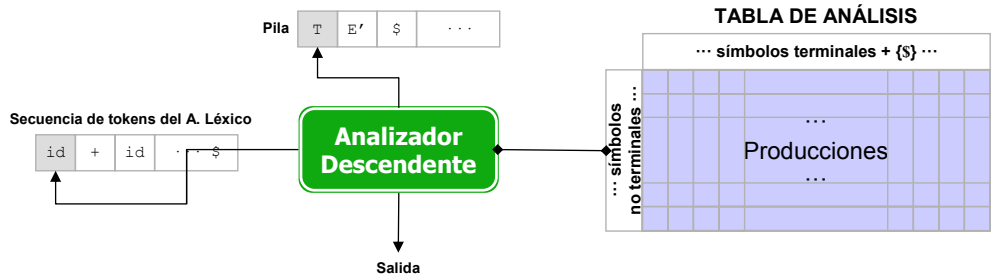
B → c | ε

F → f A | ε

4.5 Analizadores predictivos no recursivos

Se utiliza un autómata de pila en base a:

- **Alfabeto de la pila:** está formado por los símbolos terminales y no terminales de la gramática, junto con el símbolo **$** delimitador de la cadena de entrada.
- **Alfabeto de entrada:** está formado por los símbolos terminales.
- **Función de transición:** se construye en base a una tabla de análisis y al proceso de derivación de las producciones de la gramática.

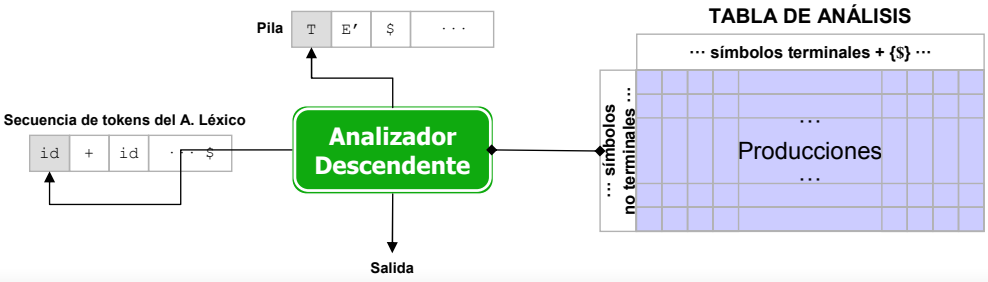


La tabla de análisis es un array de la forma **M[A,a]**, donde **A** representa un símbolo no terminal y **a** representa un símbolo terminal.

4.5.1 Descripción funcional del método predictivo no recursivo

Descripción funcional

1. Sea a el símbolo de entrada y X el símbolo situado en el tope de la pila.
2. Si $X = a = \$$, termina correctamente el analizador y se para.
3. Si $X = a \neq \$$, entonces se saca X de la pila y se toma otro símbolo de entrada.
4. Si X es un símbolo no terminal, entonces:
   - Si $M[X, a]$ tiene *vacío*, entonces ERROR.
   - Si $M[X, a]$ tiene la regla $X \rightarrow rst$, entonces se saca X del tope de la pila y se introduce rst, en orden inverso, en el tope de la pila.



Ejemplo 4.2: Dada la siguiente gramática no recursiva y la tabla de análisis asociada para el método predictivo no recursivo, se muestra una evolución del análisis sintáctico para una secuencia compuesta por **id+id*id**

$$
\begin{array}{l}
E \rightarrow TE' \\
E' \rightarrow +TE' \mid \varepsilon \\
T \rightarrow FT' \\
T' \rightarrow *FT' \mid \varepsilon \\
F \rightarrow (E) \mid id
\end{array}
$$

TABLA DE ANÁLISIS

	id	+	*	(	)	$
E	$E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \varepsilon$	$E' \rightarrow \varepsilon$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \varepsilon$	$T' \rightarrow *FT$		$T' \rightarrow \varepsilon$	$T' \rightarrow \varepsilon$
F	$F \rightarrow id$			$F \rightarrow (E)$		

PILA	ENTRADA	SALIDA
$E	id +id*id$	
$E'T	id +id*id$	$E \rightarrow TE'$
$E'T'F	id +id*id$	$T \rightarrow FT'$
$E'T'id	id +id*id$	$F \rightarrow id$
$E'T'	+ id*id$	
$E'	+ id*id$	$T \rightarrow \varepsilon$
$E'T+	+ id*id$	$E' \rightarrow +TE'$
$E'T	id *id$	
$E'T'F	id *id$	$T \rightarrow FT'$
$E'T'id	id *id$	$F \rightarrow id$
$E'T'	* id$	
$E'T'F*	* id$	$T \rightarrow *FT'$
$E'T'F	id $	
$E'T'id	id $	$F \rightarrow id$
$E'T'	$	
$E'	$	$T \rightarrow \varepsilon$
$	$	$E' \rightarrow \varepsilon$

4.5.2 Algoritmo del método predictivo no recursivo

Algoritmo

Entrada: Secuencia de símbolos **ω** y tabla de análisis **M**.

Salida: Si **ω** es o no correcto sintácticamente.

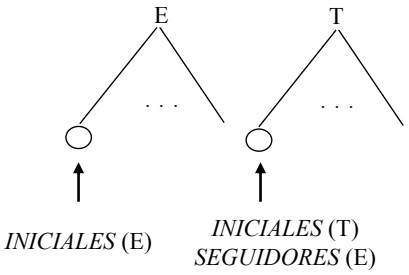
```
ip = puntero al primer token de ω
repeat
    Sea X el símbolo del tope de la pila y a=ip.
    if (X es terminal || X=$) then
        if (X=a) then
            POP (X); ip++;
        else
            ERROR;
    else
        if (M[X,a]="X→Yα") then
            POP (X);
            PUSH (αY);
        else
            ERROR;
until (X=$);
```

4.5.3 Procedimiento y algoritmo de construcción de la tabla de análisis

La **tabla de análisis** nos indica qué producción de la gramática se han de aplicar ante un determinado símbolo terminal de entrada (incluyendo el **$**).

Para construir la tabla de análisis se utilizan los conjuntos de **INICIALES** y **SEGUIDORES**.

Ejemplo 4.3: Dada la producción S → E T, podemos ver un ejemplo de árbol de análisis para apreciar los INICIALES y los SEGUIDORES.



4.5.3 Procedimiento y algoritmo de construcción de la tabla de análisis

INICIALES: Representa el conjunto de símbolos terminales que resultan al principio de la cadena resultante de derivar un símbolo no terminal.

$$INICIALES(\alpha) = \{a \in V_T \ / \ \alpha \Rightarrow^* a\beta, \beta \in (V_T \cup V_N)^*\}$$

Método para obtener el conjunto de INICIALES

1. Si X es terminal, entonces $X \in INICIALES(X)$.
2. Si $X \rightarrow \varepsilon$, entonces $\varepsilon \in INICIALES(X)$.
3. Si X es no terminal y existe la producción $X \rightarrow YZ$, tal que $Y \Rightarrow^* \varepsilon$, entonces $INICIALES(Z) \subset INICIALES(X)$.

4.5.3 Procedimiento y algoritmo de construcción de la tabla de análisis

SEGUIDORES: Contiene el conjunto de símbolos terminales que siguen a una cadena derivada.

$$SEGUIDORES(A) = \{a \,/\, S \rightarrow \gamma A \delta,\ \delta \in (V_T \cup V_N)^*,\ A \in V_N,\ a \in INICIALES(\delta)\}$$

Método para obtener el conjunto de SEGUIDORES

1. \$ pertenece al conjunto $SEGUIDORES(S)$, siendo S el símbolo inicial de la gramática.
2. $SEGUIDORES(B) = \{a \,/\, A \rightarrow \alpha B \beta,\ A, B \in V_N,\ \alpha, \beta \in (V_N \cup V_T)^*,\ a \in INICIALES(\beta),\ a \neq \varepsilon\}$
3. $SEGUIDORES(B) = \{a \,/\, \exists A \rightarrow \alpha B \vee \exists A \rightarrow \alpha B \beta,\ A, B \subset V_N,\ \alpha, \beta \subset (V_N \cup V_T)^*,\ \beta \Rightarrow^* \varepsilon,\ a \subset SEGUIDORES(A)\}$

Ejemplo 4.4: Sea la gramática sin recursividad a la izquierda que se muestra a continuación, se calculan los conjuntos INICIALES y SEGUIDORES de cada símbolo no terminal.

$$\begin{aligned} E &\rightarrow TE' \\ E' &\rightarrow +TE' \mid \varepsilon \\ T &\rightarrow FT' \\ T' &\rightarrow *FT' \mid \varepsilon \\ F &\rightarrow (E) \mid id \end{aligned}$$

El cálculo de los INICIALES es el siguiente:

$$\begin{aligned} INICIALES(E) &= \{(, id\} \\ INICIALES(T) &= \{(, id\} \\ INICIALES(F) &= \{(, id\} \\ INICIALES(E') &= \{+, \varepsilon\} \\ INICIALES(T') &= \{*, \varepsilon\} \end{aligned}$$

1. Si X es terminal, entonces $X \subset INICIALES(X)$.
2. Si $X \rightarrow \varepsilon$, entonces $\varepsilon \in INICIALES(X)$.
3. Si X es no terminal y existe la producción $X \rightarrow YZ$, tal que $Y \Rightarrow^* \varepsilon$, entonces $INICIALES(Z) \subset INICIALES(X)$.

Y el cálculo de los SEGUIDORES es el siguiente:

$$SEGUIDORES(A) = \{a \,/\, S \rightarrow \gamma A \delta,\ \delta \in (V_T \cup V_N)^*,\ A \in V_N,\ a \in INICIALES(\delta)\}$$

$$\begin{aligned} SEGUIDORES(E) &= \{\$,)\} \\ SEGUIDORES(T) &= \{+, SEGUIDORES(E), SEGUIDORES(E')\} = \{+, \$,)\} \\ SEGUIDORES(E') &= \{SEGUIDORES(E)\} = \{\$,)\} \\ SEGUIDORES(F) &= \{*, SEGUIDORES(T), SEGUIDORES(T')\} \\ SEGUIDORES(T') &= \{SEGUIDORES(T)\} = \{+, \$,)\} \end{aligned}$$

4.5.3 Procedimiento y algoritmo de construcción de la tabla de análisis

El algoritmo para construir la tabla de análisis de un método descendente predictivo no recursivo es el siguiente:

Algoritmo para construir la tabla de análisis

Entrada: Gramática G.
Salida: Tabla de análisis M.

```
for all (producción "A→α"∈G) do
   for all (símbolo terminal a∈INICIALES(α)) do
       M[A,a]= "A→α";
       if (ε∈INICIALES(α)) then
          for all (b∈SEGUIDORES(A)) do
             M[A,b]= "A→α" ;
          end for
       end if
   end for
end for
for all (entrada (i,j) indefinida en M) do
   M[i,j]= ERROR;
end for
```


4.5.3 Procedimiento y algoritmo de construcción de la tabla de análisis

$$\begin{array}{lcl} INICIALES(E) & = & \{(,id\} \\ INICIALES(T) & = & \{(,id\} \\ INICIALES(F) & = & \{(,id\} \\ INICIALES(E') & = & \{+,\varepsilon\} \\ INICIALES(T') & = & \{*,\varepsilon\} \end{array}$$

$$\begin{array}{lcl} SEGUIDORES(E) & = & \{\$,)\} \\ SEGUIDORES(T) & = & \{+,SEGUIDORES(E),SEGUIDORES(E')\} = \{+,\$,)\} \\ SEGUIDORES(E') & = & \{SEGUIDORES(E)\} = \{\$,)\} \\ SEGUIDORES(F) & = & \{*, SEGUIDORES(T), SEGUIDORES(T')\} \\ SEGUIDORES(T') & = & \{SEGUIDORES(T)\} = \{+,\$,)\} \end{array}$$

$$\begin{array}{l} E \rightarrow TE' \\ E' \rightarrow +TE' \mid \varepsilon \\ T \rightarrow FT' \\ T' \rightarrow *FT' \mid \varepsilon \\ F \rightarrow (E) \mid id \end{array}$$

TABLA DE ANÁLISIS

	id	+	*	(	)	$
E	$E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \varepsilon$	$E' \rightarrow \varepsilon$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \varepsilon$	$T' \rightarrow *FT$		$T' \rightarrow \varepsilon$	$T' \rightarrow \varepsilon$
F	$F \rightarrow id$			$F \rightarrow (E)$		

4.6 Gramáticas LL(1)

Gramática LL(1): Una gramática es LL(1) cuando no hay más de una producción en cada celda de la tabla de análisis. Cuando aparecen más de una producción en una o más celdas de la tabla de análisis, se dice que la gramática es ambigua LL(1).

Gramática LL(1) Simple: Es una gramática LL(1) tal que la parte derecha de las producciones que posean alternativas comienzan por símbolos terminales distintos y además no aparecen producciones a la cadena vacía.

La gramática LL(1) simple se le denomina S-Gramática, ya que todas las producciones con alternativas son de la forma:

$$A \rightarrow a_1\alpha_1 \mid a_2\alpha_2 \mid \cdots \mid a_n\alpha_n, \text{ donde } a_i \neq a_j, i \neq j \wedge a_i \in V_T$$

4.7 Condiciones para que una gramática sea LL(1)

Para toda regla $A \rightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$:

1. $INICIALES(\alpha_i) \cap INICIALES(\alpha_j) = \phi, \forall i \neq j.$
2. En el caso de que $\alpha_i \Rightarrow^* \varepsilon$ además: $INICIALES(\alpha_j) \cap SEGUIDORES(A) = \phi, \forall j \neq i$

Ejemplo 4.5: Dada la gramática descrita por las siguientes producciones.

$$S \rightarrow \mathrm{w}A\mathrm{z} \mid \mathrm{x}B\mathrm{z} \mid \mathrm{w}B\mathrm{y} \mid \mathrm{w}B\mathrm{z} \mid \mathrm{x}A\mathrm{y}$$
$$A \rightarrow \mathrm{v}$$
$$B \rightarrow \mathrm{v}$$

Tenemos que estudiar la primera producción que es la que posee alternativas:

$$S \rightarrow \overbrace{\mathrm{w}A\mathrm{z}}^{\alpha_1} \mid \overbrace{\mathrm{x}B\mathrm{z}}^{\alpha_2} \mid \overbrace{\mathrm{w}B\mathrm{y}}^{\alpha_3} \mid \overbrace{\mathrm{w}B\mathrm{z}}^{\alpha_4} \mid \overbrace{\mathrm{x}A\mathrm{y}}^{\alpha_5}$$

$$\left.\begin{array}{l} INICIALES(\alpha_1) = \{\mathrm{w}\} \\ INICIALES(\alpha_2) = \{\mathrm{x}\} \\ INICIALES(\alpha_3) = \{\mathrm{w}\} \\ INICIALES(\alpha_4) = \{\mathrm{w}\} \\ INICIALES(\alpha_5) = \{\mathrm{x}\} \end{array}\right\} \quad INICIALES(\alpha_1) \cap INICIALES(\alpha_3) \neq \phi$$

→→→ No es LL(1) ←←←

4.8 Tratamiento de errores

Cuando se alcanza una celda vacía en la tabla de análisis entonces se puede afirmar que la secuencia de entrada es errónea sintácticamente.

Desde la óptica de los traductores de lenguajes de programación no basta con saber si la secuencia de entrada es o no correcta sintácticamente. En ese sentido se debe:

- **Anunciar el mensaje** de error de forma que pueda localizarse con facilidad.
- **Recuperarse para poder seguir analizando** el resto de los símbolos de entrada.

Estrategias para el tratamiento de los errores:

- A nivel de frase (ad hoc, **[Trem88]**).
- Modo pánico (*Pánic*) (sincronización).

Tratamiento a nivel de frase

Consiste en rellenar las celdas vacías de la tabla de análisis con marcas que nos permitan seleccionar un procedimiento concreto para el tratamiento del error según sea la celda vacía alcanzada.

Para cada celda vacía en la tabla de análisis, será necesario estudiar cuáles pueden ser las causas para alcanzarla, así como establecer los medios más adecuados para recuperarse ante el error.

Tratamiento en modo pánico (*Panic*)

Detectado un error, el analizador se recupera del mismo saltando símbolos de la entrada hasta que aparezca un símbolo tal que con el símbolo situado en el tope de la pila nos haga seleccionar una celda no vacía en la tabla de análisis.

Para optimizar el proceso de recuperación en modo pánico será necesario delimitar los conjuntos de símbolos de sincronización.

Selección del conjunto de símbolos de sincronización

Debemos concretar los tipos de errores que se desean tratar y, para cada caso concreto, adoptar los símbolos de sincronización adecuados.

Casos más conocidos:

1. Para cada símbolo no terminal ***A***, incluir como símbolos de sincronización los **SEGUIDORES(*A*)**.
2. Cuando se usan delimitadores de bloque o sentencia y se omite el delimitador final, resulta más óptimo sincronizar por **INICIALES** del bloque o sentencia.
3. Si un símbolo no terminal ***A*** deriva en la cadena vacía, entonces no se podrá detectar el error.
4. Si aparece un símbolo terminal ***a*** en el tope de la pila y no coincide con el símbolo de entrada, entonces se podrá optar por sacar el símbolo a de la pila y emitir un mensaje que nos indique que el símbolo ***a*** ha sido omitido.

Ejemplo 4.6: Dada la siguiente gramática junto con su tabla de análisis para un analizador descendente predictivo, se realiza el análisis de la secuencia de entrada `id+*id`

$$
\begin{aligned}
&E \to TE' \\
&E' \to +TE' \mid \varepsilon \\
&T \to FT' \\
&T' \to *FT' \mid \varepsilon \\
&F \to (E) \mid id
\end{aligned}
$$

TABLA DE ANÁLISIS

	id	+	*	(	)	$
E	$E \to TE'$			$E \to TE'$		
E′		$E' \to +TE'$			$E' \to \varepsilon$	$E' \to \varepsilon$
T	$T \to FT'$			$T \to FT'$		
T′		$T' \to \varepsilon$	$T' \to *FT$		$T' \to \varepsilon$	$T' \to \varepsilon$
F	$F \to id$			$F \to (E)$		

PILA	ENTRADA	SALIDA
$E	id+*id$	
$E'T	id+*id$	$E \to TE'$
$E'T'F	id+*id$	$T \to FT'$
$E'T'id	id+*id$	$F \to id$
$E'T'	+*id$	
$E'	+*id$	$T' \to \varepsilon$
$E'T+	+*id$	$E' \to +TE'$
$E'T	*id$	***ERROR***
¿qué hacer a partir de aquí?		