Tema 5 | Análisis Sintáctico Ascendente

Contenidos

Bibliografía básica

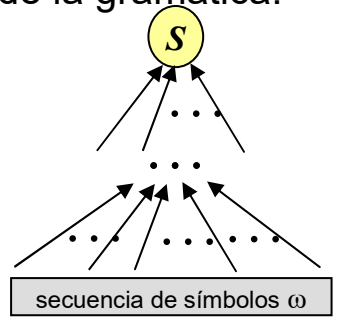
14-Feb-2009

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

[Trem85] J Tremblay, P.G. Sorenson
The theory and practice of compiler writing. McGraw-Hill, 1985.

5.1 Estrategia ascendente y métodos de análisis ascendente

Objetivo: Comprobar si hay una secuencia de símbolos pertenece a un lenguaje $L(G)$ tal que aplicando reducciones sobre la cadena de entrada en base a las producciones de la gramática nos permita alcanzar el símbolo inicial de la gramática.



Métodos de análisis ascendente:

- Método de **reducción-desplazamiento** (no predictivo).
- Métodos predictivos:
  - Análisis de **precedencia de operador**.
  - Análisis de **precedencia simple**.
  - Análisis **LR**.

5.2 Método de reducción y desplazamiento

Se basa en una máquina de pila donde:

- El **alfabeto de la pila** está formado por los símbolos terminales y no terminales.
- El **alfabeto de entrada** está formado por los símbolos terminales.
- La **función de transición** se define en base a las siguientes acciones sobre la pila:

{ *Desplazar*, *Reducir*, *Aceptar*, *Error* }

Parte de la secuencia de símbolos de la entrada.
El proceso de análisis consiste en ir explorando el tope de la pila y los elementos siguientes en la pila hasta encontrar una subcadena τ que coincida con la parte derecha de una producción dada $A \rightarrow \tau$.

Si se encuentra τ en la pila, entonces se sustituye τ por A en el tope de la pila. A este proceso se le denomina **reducción.** Y el proceso de pasar símbolos de entrada al tope de la pila se llama **desplazamiento**.

Problemas:

- Que existe más de una producción para aplicar reducción (conflicto reduce/reduce).
- Incertidumbre si es posible aplicar desplazamiento en vez de reducción (conflicto desplaza/reduce). Siempre es posible aplicar desplazamiento, ampliando las posibles subcadenas en el tope de la pila suceptibles de ser reducidas en base a un mayor número de producciones. ¿Qué acción es la más apropiada?.

Descripción funcional del método de reducción y desplazamiento

Sea la gramática $G = (V_N, V_T, P, S)$ y sea $V = V_N \cup V_T \cup \{\varepsilon\}$. Se define la función f y g de la forma siguiente:

$$f : V^* \times (V_T \cup \{\varepsilon\})^* \longrightarrow \{desplaza, reduce, error, acepta\}$$

$$g : V^* \times (V_T \cup \{\varepsilon\})^* \longrightarrow \{1, 2, 3, \ldots, n\}$$

donde $\{1, 2, 3, \ldots, n\}$ representa la enumeración de las producciones de la gramática de tal forma que si $\exists i \in [1..n]$, $\exists p_i \in P$.

Descripción funcional: Sea D la descripción instantánea de la máquina de pila representada por la terna que forman el estado de la pila, la secuencia de entrada y las producciones aplicadas.

$$D = (\underbrace{\$X_1 \cdots\cdots X_n}_{\text{Estado de la pila}} , \underbrace{a_1 \cdots\cdots a_m}_{\text{Cadena restante}} , \underbrace{P_1 \cdots\cdots P_s}_{\text{Producciones usadas}})$$

Descripción funcional del método de reducción y desplazamiento

Si representamos las acciones de la siguiente forma, tenemos las siguientes situaciones:

Desplazar:	$\Rightarrow^d$
Reducir:	$\Rightarrow^r$

- Si $f(\alpha, a\tau) = desplaza$, entonces añadir un símbolo:

$$(\alpha, a\tau, \Lambda) \Rightarrow^d (\alpha a, \tau, \Lambda)$$

- Si $f(\alpha\beta, \tau) = reducir$ y $g(\alpha\beta, \tau) = i$, donde i es la producción $A \rightarrow \beta$, entonces

$$(\alpha\beta, \tau, \Lambda) \Rightarrow^r (\alpha A, \tau, \Lambda_i)$$

- Si $f(\alpha, \tau) = aceptar$, entonces:

$$(\alpha, \tau, \Lambda) \Rightarrow^d aceptar$$

En cualquier otro caso:

$$(\alpha, \tau, \Lambda) \Rightarrow^d Error$$

Ejemplo 5.1: Sea G una gramática definida por las producciones siguientes:

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

Y sea la cadena de entrada **`id+id*id`**. El análisis sintáctico ascendente según el método general de reducción y desplazamiento es el siguiente:

Tope de la Pila	Entrada	Acción
$\$$	$id + id * id\$$	*Desplaza*
$\$id$	$+id * id\$$	*Reduce* $(E \rightarrow id)$
$\$E$	$+id * id\$$	*Desplaza*
$\$E+$	$id * id\$$	*Desplaza*
$\$E + id$	$*id\$$	*Reduce* $(E \rightarrow id)$
$\$E + E$	$*id\$$	*Desplaza o Reduce* $(E \rightarrow E + E)$
$\$E + E*$	$id\$$	*Desplaza*
$\$E + E * id$	$\$$	*Reduce* $(E \rightarrow id)$
$\$E + E * E$	$\$$	*Reduce* $(E \rightarrow E * E)$
$\$E + E$	$\$$	*Reduce* $(E \rightarrow E + E)$
$\$E$	$\$$	*Aceptar*

PROBLEMA: CUANDO PODEMOS APLICAR MÁS DE UNA ACCIÓN ANTE UN SÍMBOLO DE ENTRADA Y NO SE ALCANZA EL SÍMBOLO INICIAL DE LA GRAMÁTICA, ENTONCES PUEDE SER NECESARIO APLICAR VUELTA ATRÁS (RETROCESO).

5.3 Análisis ascendente predictivo de precedencia de operador

Resuelve la acción alternativa (ante posibles conflictos desplaza/reduce) en base a criterios de precedencia entre operadores.

El análisis de precedencia de operador es aplicable a lenguajes definidos por gramáticas con las siguientes restricciones:

1. No podrán aparecer dos símbolos no terminales consecutivos en las producciones.
2. No podrán existir producciones a la cadena vacía.
3. Las relaciones de precedencia deben ser disjuntas.

La tabla de análisis de precedencia de operador está formada únicamente por símbolos terminales puestos de la siguiente forma:

	$\cdots V_T \cdots$	a_j
V_T	$\ddots$	
a_i	$\cdots\cdots\cdots$	D_{ij}

donde $D_{ij} = \begin{cases} \text{Vacío} \\ a_i \gtrdot a_j \\ a_i \lessdot a_j \\ a_i \doteq a_j \end{cases}$

5.3.3 Método para obtener las relaciones de precedencia de operador

Las *relaciones de precedencia de operador* se calculan de la siguiente forma, para una gramática $G = (V_N, V_T, P, S)$:

1. $a \doteq b$: Si $\exists A \rightarrow \alpha a \beta b \gamma \ \wedge \ (\beta \in V_N \vee \beta = \{\varepsilon\}$.
2. $a \lessdot b$: Si $\exists A \rightarrow \alpha a B \beta \ \wedge \ (B \Rightarrow^+ b\ldots \vee B \Rightarrow^+ Cb\ldots, C \in V_N)$.
3. $a \gtrdot b$: Si $\exists A \rightarrow \alpha B b \beta \ \wedge \ (B \Rightarrow^+ \ldots a \vee B \Rightarrow^+ \ldots aY, Y \in V_N)$.

El *método para obtener la relación de precedencia de operador* se realiza definiendo las funciones $PRIMERO$ y $ULTIMO$ de la siguiente forma:

$$\textbf{PRIMERO}(\text{A}) = \{a \,/\, A \Rightarrow^+ a\cdots \vee A \Rightarrow^+ Xa\cdots\ A, X \in V_N, a \in V_T\}$$

$$\textbf{ULTIMO}(\text{A}) = \{a \,/\, A \Rightarrow^+ \cdots a \vee A \Rightarrow^+ \cdots aX\ A, X \in V_N, a \in V_T\}$$

5.3.3 Método para obtener las relaciones de precedencia de operador

Algoritmo para Obtener las Relaciones de Precedencia de Operador

for all $A \rightarrow X_1 X_2 \cdots X_n$ **do**
  for $(I = 1$ **to** $(n-1))$ **do**
    if $(X_I \in V_T) \wedge (X_{I+1} \in V_T)$ **then**
      $X_I \doteq X_{I+1}$;
    end if
    if $(I < (n-2)) \wedge (X_I \in V_T) \wedge (X_{I+2} \in V_T) \wedge (X_{I+1} \in V_N)$ **then**
      $X_I \doteq X_{I+2}$;
    end if
    if $(X_I \in V_T) \wedge (X_{I+1} \in V_N)$ **then**
      for all $(a \in PRIMERO(X_{I+1}))$ **do**
        $X_I \lessdot a$;
      end for
    end if
    if $(X_I \in V_N) \wedge (X_{I+1} \in V_T)$ **then**
      for all $(a \in ULTIMO(X_I))$ **do**
        $a \gtrdot X_{I+1}$;
      end for
    end if
  end for
end for

Para todo $a \in PRIMERO(S)$ hacer $\$ \lessdot a$.

Para todo $b \in ULTIMO(S)$ hacer $b \gtrdot \$$.

Algoritmo de análisis de precedencia de operador

Algoritmo de Análisis de Precedencia de Operador

```
while cadena de entrada no vacía do
  if ($ = Top()) ∧ ($ = SimboloEntrada()) then
    return("Cadena aceptada");
  else
    a=Top();
    b=SimboloEntrada();
    if (a ⋖ b) ∨ (a ≐ b) then
      Push(b);
    else
      if a ⋗ b then
        repeat
          α=Pop();
        until alpha ⋖ a
      else
        return("Error");
      end if
    end if
  end if
end while
```

Ejemplo 5.2: Sea G una gramática definida por las producciones siguientes:

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

Basándonos en relaciones de precedencia obtenidas bajo premisas establecidas (método intuitivo), la tabla de precedencia de operador y el análisis de la secuencia de entrada `id+id*id`, son respectivamente:

TABLA DE PRECEDENCIA

	id	+	*	(	)	$
id		$\gtrdot$	$\gtrdot$		$\gtrdot$	$\gtrdot$
+	$\lessdot$	$\gtrdot$	$\lessdot$	$\lessdot$	$\gtrdot$	$\gtrdot$
*	$\lessdot$	$\gtrdot$	$\lessdot$	$\lessdot$	$\gtrdot$	$\gtrdot$
(	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$	
)		$\gtrdot$	$\gtrdot$		$\gtrdot$	$\gtrdot$
$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$		

PILA	PRECEDENCIA	ENTRADA
$	$\lessdot$	$id + id * id\$$
$id	$\gtrdot$	$+id * id\$$
$	$\lessdot$	
$+	$\lessdot$	$id * id\$$
$ + id	$\gtrdot$	$*id\$$
$+	$\lessdot$	
$ + *	$\lessdot$	$id\$$
$ + *id	$\gtrdot$	$\$$
$ + *	$\gtrdot$	
$+	$\gtrdot$	
$		

Problemas del análisis de precedencia de operador

- **No contempla** la posibilidad de que haya tokens con **doble precedencia**.

  El operador "menos" tiene más precedencia cuando actúa como unario y menos precedencia cuando actúa como binario.

- El algoritmo puede aceptar entradas que no se corresponden con la sintaxis

  Ejemplo: `id++id`. Se debe a que no comprueba la parte derecha de las producciones antes de realizar la eliminación de símbolos en la pila (reducción).

Métodos para obtener las relaciones de precedencia de operador

1. **Método intuitivo**: Basado en el sentido común y asociado con las reglas del cálculo en donde se aplica una precedencia de cálculo según una precedencia de operadores. Se conoce el lenguaje en virtud de la gramática y es sencillo extraer las relaciones de precedencias.

2. **Método deductivo**: Basado en el análisis de las gramáticas mediante el desarrollo de los árboles de análisis. Como requisito, las gramáticas deben ser no ambiguas. Útil cuando la gramática no permite determinar las precedencias.

3. **Método asociativo**: Basado en la asociatividad de los operadores y en las reglas de precedencia de operadores para el cálculo **[Aho90]**. Método general de resolución de conflictos en base a precedencia y asociatividad.

Precedencia y asociatividad de operadores

1. Si un operador O_1 tiene mayor precedencia que O_2, entonces $O_1 \gtrdot O_2$ y $O_2 \lessdot O_1$.
2. Si los operadores O_1 y O_2 son operadores de igual precedencia entonces:
   (a) Si los operadores son asociativos por la izquierda entonces hacer $O_1 \gtrdot O_2$ y $O_2 \gtrdot O_1$.
   (b) Si los operadores son asociativos por la derecha entonces hacer $O_1 \lessdot O_2$ y $O_2 \lessdot O_1$.

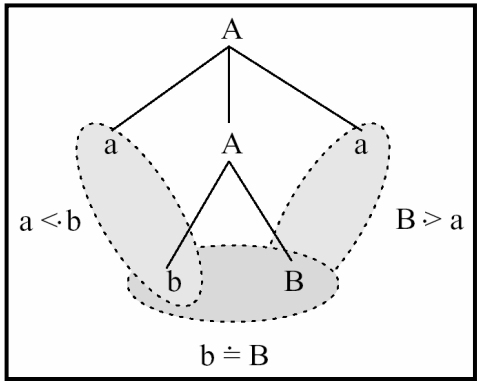
5.4 Análisis ascendente predictivo de precedencia simple

Propuesta por Wirth y Weber en 1966 como generalización de la precedencia de operador. Observaron que determinados símbolos terminales aparecen siempre adyacentes dependiendo de las producciones de la gramática, de tal forma que se puede establecer relaciones de precedencia entre símbolos terminales y no terminales.

Esta relación nos puede servir para determinar la subcadena de entrada que debe ser reducida o desplazada.

Se definen tres relaciones de precedencia entre símbolos terminales y no terminales en base a su disposición en el árbol de análisis:

1. **Menor precedencia** $a \lessdot b$: Si **a** cuelga de algún nodo superior a la izquierda del nodo que cuelga **b**.
2. **Mayor precedencia** $a \gtrdot b$: Si **a** cuelga de algún nodo superior a la derecha del nodo que cuelga **b**.
3. **Igual precedencia** $a \doteq b$: Si **a** y **b** cuelgan del mismo nodo.



5.4 Análisis ascendente predictivo de precedencia simple

Dado un árbol de análisis, para definir las relaciones de precedencia se aplican las reglas siguientes:

1. Todos los símbolos que cuelgan de nodos superiores y están a la izquierda inmediatamente, son de menor precedencia, es decir:

$$X \lessdot Y \text{ si } A \rightarrow \alpha X B \beta \text{ y } B \Rightarrow^{+} Y\gamma$$

2. Todos los símbolos que cuelgan de un mismo nodo son entre sí de igual precedencia, es decir:

$$X \doteq Y \text{ si } A \rightarrow \alpha X Y \beta$$

3. Todos los símbolos que cuelgan de nodos superiores y están a la derecha, son de mayor precedencia, es decir:

$$X \gtrdot a \text{ si } a \in V_T \text{ y } B \rightarrow \alpha C a \beta \text{ y } C \Rightarrow^{+} \gamma X$$

5.4.1 Concepto de pivote y relación de precedencia

Pivote: Subcadena susceptible de ser reducida.

El análisis de precedencia simple es aplicable a lenguajes definidos por gramáticas con las siguientes restricciones:

1. No podrán existir producciones cuya parte derecha sea la misma.
2. No podrán existir producciones a la cadena vacía.
3. Las relaciones de precedencia deben ser disjuntas.

Proceso para determinar el pivote: Dada una secuencia de entrada, para determinar el pivote se obtiene la relación de precedencia entre los símbolos adyacentes y, cuando detectemos un cambio de precedencia de menor/igual a mayor, entonces los símbolos comprendidos entre las relaciones de menor y mayor precedencia formarán el pivote por el que se debe reducir.

Un ejemplo simple sería el siguiente:

$$a \lessdot b \lessdot \Lambda \lessdot \underbrace{d \doteq e \doteq B \doteq \Lambda}_{\text{pivote}} \gtrdot i \gtrdot k$$

5.4.2 Tabla de precedencia simple

La tabla de análisis de precedencia simple está formada por símbolos terminales y no terminales puestos de la siguiente forma:

	$\cdots (V_T \cup V_N) \cdots$	a_j
$\vdots$ $(V_T \cup V_N)$ $\vdots$	$\ddots$	$\vdots$
a_i	$\cdots\cdots\cdots$	D_{ij}
$\vdots$		

donde $D_{ij} = \begin{cases} \text{Vacío} \\ a_i \gtrdot a_j \\ a_i \lessdot a_j \\ a_i \doteq a_j \end{cases}$

Método de cálculo de las relaciones de precedencia simple

De forma parecida a precedencia de operador, se definen los conceptos *PRIMERO*$^+$ y *ULTIMO*$^+$ en los siguientes términos:

$$PRIMERO^+(A) = \{\alpha \,/\, A \Rightarrow^+ \alpha\cdots \wedge \; si\ \alpha \in V_N,\ entonces\ PRIMERO^+(\alpha) \subseteq PRIMERO^+(A)\}$$

$$ULTIMO^+(A) = \{\alpha \,/\, A \Rightarrow^+ \cdots\alpha \wedge \; si\ \alpha \in V_N,\ entonces\ ULTIMO^+(\alpha) \subseteq ULTIMO^+(A)\}$$

Análogamente a precedencia de operador:

Para todo $a \in PRIMERO^+(S)$ hacer $\$ \lessdot a$.

Para todo $b \in ULTIMO^+(S)$ hacer $b \gtrdot \$$.

Ejemplo 5.3: Dada la gramática definida por las producciones:

$$S \to aAa$$
$$A \to bB$$
$$A \to c$$
$$B \to Acd$$

las relaciones de precedencia simple estudiadas son las siguientes:

$$\begin{array}{c} a \doteq A,\ A \doteq a \\ b \doteq B \\ A \doteq c,\ c \doteq d \end{array} \left| \begin{array}{lll} a \lessdot b & a \lessdot c & \\ B \gtrdot a & c \gtrdot a & d \gtrdot a \\ b \lessdot A & b \lessdot b & b \lessdot c \\ B \gtrdot c & c \gtrdot c & d \gtrdot c \end{array} \right.$$

La tabla de precedencia simple es la siguiente:

	S	a	A	B	c	b	d	\$
S								
a			$\doteq$		$\lessdot$	$\lessdot$		$\gtrdot$
A		$\doteq$			$\doteq$			
B		$\gtrdot$			$\gtrdot$			
c		$\gtrdot$			$\gtrdot$		$\doteq$	
b			$\lessdot$	$\doteq$	$\lessdot$	$\lessdot$		
d		$\gtrdot$			$\gtrdot$			
\$		$\lessdot$						

Supongamos la secuencia de entrada ***abccda***, el proceso de análisis sería el siguiente:

$$a \lessdot b \lessdot \underbrace{c}_{} \gtrdot c \doteq d \gtrdot a$$
$$\boxed{A \to c}$$
$$a \lessdot b \lessdot \underbrace{A \doteq c \doteq d} \gtrdot a$$
$$\boxed{B \to Acd}$$
$$a \lessdot \underbrace{b \doteq B} \gtrdot a$$
$$\boxed{A \to bB}$$
$$\underbrace{a \doteq A \doteq a}$$
$$\boxed{S \to aAa}$$

5.5 Tratamiento de errores en análisis de precedencia

Detección de errores

1. Cuando no esté definida la relación de precedencia entre dos símbolos adyacentes de entrada: Error de precedencia.
2. Cuando se ha obtenido un pivote para aplicar reducción y no existe ninguna producción con idéntica parte derecha al pivote: Error de reducción.

Manejo de errores durante la reducción

Ante un error se actuará sacando el símbolo de la pila y notificar el error. Se distinguen dos casos:

- Cuando el pivote no casa pero alguna subsecuencia si.
- Cuando falte algún símbolo para poder aplicar con éxito la reducción.

No es trivial obtener un mensaje certero en análisis de precedencia. Cuando se detecta este tipo de error hay que decidir qué símbolos de la pila hay que sacar.

[Aho90] propone un sistema simple consistente en sacar todos los símbolos con igual precedencia en el orden en el que fueron introducidos hasta que se alcance un <· o ·> con el símbolo inicial.

5.5 Tratamiento de errores en análisis de precedencia

También, Aho [Aho90] propone otro método basado en el sentido común.

Ejemplo 6.4: Dada la gramática definida por las producciones: $E \rightarrow E+E \mid E*E \mid (E) \mid id$

Se emite los mensajes:

- Si se reduce "+" o "*" y no hay símbolo no terminal a izquierda y derecha, "`Falta operando`".
- Si se reduce " (" y ") " y no hay símbolo no terminal, "`No hay expresion entre los paréntesis`".

Manejo de errores cuando no hay relación de precedencia (error de precedencia)

Se estudian los símbolos cercanos al tope de la pila, pudiéndose dar los casos siguientes:

a) Saltar el símbolo de la entrada.
b) Saltar el símbolo en el tope de la pila o ambos (entrada y pila).
c) Insertar un símbolo en la pila.
d) Insertar un símbolo en la entrada.

Problema: Se puede alcanzar un **lazo infinito** en el algoritmo de análisis ya que podemos estar realizando alguna de estas acciones de forma indefinida sin detectar alguna precedencia correcta.