CI3725 - Traductores e Interpretadores
Abril-Julio 2006
Prof. Ernesto Hernández-Novich

Laboratorio Semana XI

Parser Predictivo no Recursivo

Es una máquina de estados que consta de:

- Una pila que contiene símbolos de la gramática, más un símbolo especial ($) para indicar el fondo de la pila.
- Un apuntador al siguiente elemento de la entrada. Al comenzar la ejecución de la máquina apunta al primer caracter de la entrada.
- Un mecanismo de emisión para presentar por salida los números asociados a las producciones de la gramática que han sido utilizadas. Al terminar la ejecución del parser, la salida tendrá las producciones necesarias para conseguir una derivación más izquierda de la cadena suministrada en la entrada, a partir del símbolo inicial de la gramática.
- Una tabla de parsing predictivo que controla el funcionamiento de la máquina de estados. La tabla de *reglas* ($N \times \Sigma$) indica la producción que deben utilizarse en cada caso.

¿Cómo funciona? La ejecución de una máquina de parsing predictivo no recursivo es como sigue:

1. Al comenzar el procesamiento de una cadena la pila contiene el símbolo inicial de la gramática y se apunta al primer caracter de la entrada.
2. Se observa el símbolo X en el tope de la pila y el símbolo terminal **a** que está en la entrada:
   a) Si $X = \$ \wedge \mathbf{a} = \$$, quiere decir que la pila ha sido vaciada y que se ha consumido toda la entrada, por tanto la máquina se detiene anunciando el éxito del reconocimiento.
   b) Si $X = \mathbf{a} \wedge X \neq \$$, quiere decir que el símbolo al tope de la pila coincide con el siguiente de la entrada, por tanto se consume la entrada y se retira el tope de la pila.
   c) Si $X \in N$, la máquina consulta $reglas[X, \mathbf{a}]$:
      1) Si la tabla contiene una producción $X \rightarrow Y_1 Y_2 \ldots Y_k$, se emite por la salida la producción, se retira X del tope de la pila y se empilan $Y_k \ldots Y_2 Y_1$ de modo que Y_1 quede en el **tope** de la pila.
      2) Si la tabla no contiene una producción, se termina la ejecución indicando el error.

Construcción de la Tabla de Parsing Predictivo No Recursivo

El algoritmo de construcción consiste en para toda producción $A \rightarrow \alpha$ de la gramática:

1. $\forall \mathbf{a} \in \Sigma \wedge \mathbf{a} \in FIRST(\alpha) \Rightarrow reglas[A, \mathbf{a}] = A \rightarrow \alpha$
2. Si $\lambda \in FIRST(\alpha)$ entonces $\forall b \in FOLLOW(A) \Rightarrow reglas[A, \mathbf{b}] = A \rightarrow \alpha$
3. Si $\lambda \in FIRST(\alpha) \wedge \$ \in FOLLOW(A) \Rightarrow reglas[A, \$] = A \rightarrow \alpha$

Aquellas posiciones de la tabla que queden vacías después de considerar todas las producciones, serán completadas con *error*.

Ejemplo 1

Construyamos la tabla de parsing predictivo no recursivo para la gramática

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

Observemos que

$$\begin{aligned} FIRST(E) = FIRST(T) = FIRST(F) &= \{(, \mathbf{id}\} \\ FIRST(E') &= \{+, \lambda\} \\ FIRST(T') &= \{*, \lambda\} \\ FOLLOW(E) = FOLLOW(E') &= \{), \$\} \\ FOLLOW(T) = FOLLOW(T') &= \{+,), \$\} \\ FOLLOW(F) &= \{+, *,), \$\} \end{aligned}$$

Ahora aplicamos el algoritmo de construcción. Observamos las producciones:

- Para $E \rightarrow TE'$, como $FIRST(TE') = FIRST(T)$ tenemos $reglas[E, (] \leftarrow E \rightarrow TE'$ y $reglas[E, \mathbf{id}] \leftarrow E \rightarrow TE'$.
- Para $E' \rightarrow +TE'$, como $FIRST(+TE') = \{+\}$ tenemos $reglas[E', +] \leftarrow +TE'$.
- Para $E' \rightarrow \lambda$, usamos $FOLLOW(E')$ y tenemos $reglas[E',)] \leftarrow E' \rightarrow \lambda$ y $reglas[E', \$] \leftarrow E' \rightarrow \lambda$.
- Para $T \rightarrow FT'$, como $FIRST(FT') = FIRST(F)$ tenemos $reglas[T, (] \leftarrow T \rightarrow FT'$ y $reglas[T, \mathbf{id}] \leftarrow T \rightarrow FT'$.
- Para $T' \rightarrow *FT'$, como $FIRST(*FT') = \{*\}$ tenemos $reglas[T', *] \leftarrow T' \rightarrow *FT'$.
- Para $T' \rightarrow \lambda$, usamos $FOLLOW(T')$ y tenemos $reglas[T', +] \leftarrow T' \rightarrow \lambda$, $reglas[T',)] \leftarrow T' \rightarrow \lambda$ y $reglas[T', \$] \leftarrow T' \rightarrow \lambda$.
- Para $F \rightarrow (E)$, como $FIRST((E)) = \{(\}$ tenemos $reglas[F, (] \leftarrow F \rightarrow (E)$.
- Para $F \rightarrow \mathbf{id}$, como $FIRST(\mathbf{id}) = \{\mathbf{id}\}$ tenemos $reglas[F, \mathbf{id}] \leftarrow F \rightarrow \mathbf{id}$.

Y la tabla nos queda como sigue (los espacios vacíos llevan *error*)

	id	+	*	(	)	$
E	$E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow \mathbf{id}$			$F \rightarrow (E)$		

Cuadro 1: Tabla de Parsing Predictivo No-Recursivo para el Ejemplo 1

Ilustraremos el funcionamiento del parser predictivo no recursivo utilizando la tabla para hacer el análisis de la cadena $\mathbf{id} + \mathbf{id} * \mathbf{id}\$$ que pertenece al lenguaje. En la tabla, la pila crece hacia la derecha (nótese el uso de \$ para indicar el fondo de la pila).

Pila	Entrada	Salida	Detalles
$\$E$	$\mathbf{id} + \mathbf{id} * \mathbf{id}\$$		Estado inicial. Aplicar $reglas[E, \mathbf{id}]$
$\$E'T$	$\mathbf{id} + \mathbf{id} * \mathbf{id}\$$	$E \to TE'$	Aplicar $reglas[T, \mathbf{id}]$
$\$E'T'F$	$\mathbf{id} + \mathbf{id} * \mathbf{id}\$$	$T \to FT'$	Aplicar $reglas[F, \mathbf{id}]$
$\$E'T'\mathbf{id}$	$\mathbf{id} + \mathbf{id} * \mathbf{id}\$$	$F \to \mathbf{id}$	Consumir **id** y desempilar.
$\$E'T'$	$+\mathbf{id} * \mathbf{id}\$$		Aplicar $reglas[T', +]$
$\$E'$	$+\mathbf{id} * \mathbf{id}\$$	$T' \to \lambda$	Aplicar $reglas[E', +]$
$\$E'T+$	$+\mathbf{id} * \mathbf{id}\$$	$E' \to +TE'$	Consumir + y desempilar.
$\$E'T$	$\mathbf{id} * \mathbf{id}\$$		Aplicar $reglas[T, \mathbf{id}]$
$\$E'T'F$	$\mathbf{id} * \mathbf{id}\$$	$T \to FT'$	Aplicar $reglas[F, \mathbf{id}]$
$\$E'T'\mathbf{id}$	$\mathbf{id} * \mathbf{id}\$$	$F \to \mathbf{id}$	Consumir **id** y desempilar.
$\$E'T'$	$*\mathbf{id}\$$		Aplicar $reglas[T', *]$
$\$E'T'F*$	$*\mathbf{id}\$$	$T' \to *FT'$	Consumir $*$ y desempilar.
$\$E'T'F$	$\mathbf{id}\$$		Aplicar $reglas[F, \mathbf{id}]$
$\$E'T'\mathbf{id}$	$\mathbf{id}\$$	$F \to \mathbf{id}$	Consumir **id** y desempilar.
$\$E'T'$	\$		Aplicar $reglas[T', \$]$
$\$E'$	\$	$T' \to \lambda$	Aplicar $reglas[E', \$]$
\$	\$	$E' \to \lambda$	Entrada aceptada.

Cuadro 2: Ejecución del Parser Predictivo No-Recursivo

Nótese que lo emitido por la salida representa las producciones que deben utilizarse para obtener una derivación más izquierda en la gramática para la cadena suministrada en la entrada. Concretamente:

$$
\begin{aligned}
E &\Rightarrow TE' \\
&\Rightarrow FT'E' \\
&\Rightarrow \mathbf{id}T'E' \\
&\Rightarrow \mathbf{id}E' \\
&\Rightarrow \mathbf{id}{+}TE' \\
&\Rightarrow \mathbf{id}{+}FTE' \\
&\Rightarrow \mathbf{id} + \mathbf{id}TE' \\
&\Rightarrow \mathbf{id} + \mathbf{id}{*}FT'E' \\
&\Rightarrow \mathbf{id} + \mathbf{id} * \mathbf{id}T'E' \\
&\Rightarrow \mathbf{id} + \mathbf{id} * \mathbf{id}E' \\
&\Rightarrow \mathbf{id} + \mathbf{id} * \mathbf{id}
\end{aligned}
$$

Ejemplo 2

Consideremos la gramática

$$
\begin{aligned}
S &\to \mathbf{i}E\mathbf{t}SS' \\
S &\to \mathbf{a} \\
S' &\to \mathbf{e}S \\
S' &\to \lambda \\
E &\to \mathbf{b}
\end{aligned}
$$

Observemos

$$\begin{aligned} FIRST(S) &= \{\mathbf{i}, \mathbf{a}\} \\ FIRST(S') &= \{\mathbf{e}\} \\ FIRST(E) &= \{\mathbf{b}\} \\ FOLLOW(S) = FOLLOW(S') &= \{\mathbf{e}, \$\} \\ FOLLOW(E) &= \{\mathbf{t}\} \end{aligned}$$

y construyamos la tabla de parsing predictivo no recursivo

	a	**b**	**e**	**i**	**t**	$
S	$S \to \mathbf{a}$			$S \to \mathbf{i}E\mathbf{t}SS'$		
S'			$S' \to \mathbf{e}S \mid \lambda$			$S' \to \lambda$
E		$E \to \mathbf{b}$				

Cuadro 3: Tabla de Pasing Predictivo No Recursivo para el Ejemplo 2

La entrada para $reglas[S', \mathbf{e}]$ contiene tanto a $S' \to \mathbf{e}S$ como a $S' \to \lambda$. La gramática es ambigua y esto se manifiesta al existir más de una producción susceptible de ser seleccionada cuando se recibe un **e** en la entrada. Podemos resolver la ambiguedad decidiendo por $S' \to \mathbf{e}S$ puesto que si escogemos la otra producción jamás se consumirían los **e** de la entrada.

Una gramática cuya tabla de parsing predictivo no tiene ninguna entrada con definiciones múltiples se denomina $LL(1)$. Ninguna gramática ambigua puede ser $LL(1)$ y tampoco puede serlo una gramática recursiva por izquierda. De hecho, puede mostrarse que si se tienen dos producciones $A \to \alpha$ y $A \to \beta$, entonces:

- Si α deriva una cadena que comienza con un terminal **a** entonces β no deriva ninguna cadena que comience con **a** y viceversa.
- Alguna de las dos producciones nunca deriva en λ.
- Si $\beta \Rightarrow^* \lambda$ entonces α no deriva ninguna cadena que comience con terminales en $FOLLOW(A)$.

Ejemplo 3

Construyamos una tabla de parsing predictivo no recursivo para la gramática

$$\begin{aligned} S &\to (L) \\ S &\to \mathbf{a} \\ L &\to L,S \\ L &\to S \end{aligned}$$

Primero debemos eliminar la recursión izquierda en la tercera producción, para convertir la gramática en

$$\begin{aligned} S &\to (L) \\ S &\to \mathbf{a} \\ L &\to SR \\ R &\to ,S \\ R &\to \lambda \end{aligned}$$

En seguida calculamos

$$\begin{aligned} FIRST(S) = FIRST(L) &= \{(, \mathbf{a}\} \\ FIRST(R) &= \{,, \lambda\} \\ FOLLOW(S) &= \{\$,), ,\} \\ FOLLOW(L) = FOLLOW(R) &= \{)\} \end{aligned}$$

lo cual nos permite calcular la tabla

	(	)	**a**	,	$
S	$S \to (L)$		$S \to \mathbf{a}$		
L	$L \to SR$		$L \to SR$		
R		$R \to \lambda$		$R \to ,S$	

Cuadro 4: Tabla de Pasing Predictivo No Recursivo para el Ejemplo 3

Comprobaremos su funcionamiento procesando la cadena $(\mathbf{a}, (\mathbf{a}, \mathbf{a}))$ que pertenece al lenguaje

Pila	Entrada	Salida	Detalles
$\$S$	$(\mathbf{a}, (\mathbf{a}, \mathbf{a}))\$$		Estado inicial. Aplicar $reglas[S, (]$
$\$)L($	$(\mathbf{a}, (\mathbf{a}, \mathbf{a}))\$$	$S \to (L)$	Consumir (y desempilar.
$\$)L$	$\mathbf{a}, (\mathbf{a}, \mathbf{a}))\$$		Aplicar $reglas[L, \mathbf{a}]$
$\$)RS$	$\mathbf{a}, (\mathbf{a}, \mathbf{a}))\$$	$L \to SR$	Aplicar $reglas[S, \mathbf{a}]$
$\$)R\mathbf{a}$	$\mathbf{a}, (\mathbf{a}, \mathbf{a}))\$$	$S \to \mathbf{a}$	Consumir **a** y desempilar.
$\$)R$	$, (\mathbf{a}, \mathbf{a}))\$$		Aplicar $reglas[R, ,]$
$\$)S,$	$, (\mathbf{a}, \mathbf{a}))\$$	$R \to ,S$	Consumir , y desempilar.
$\$)S$	$(\mathbf{a}, \mathbf{a}))\$$		Aplicar $reglas[S, (]$
$\$))L($	$(\mathbf{a}, \mathbf{a}))\$$	$S \to (L)$	Consumir (y desempilar.
$\$))L$	$\mathbf{a}, \mathbf{a}))\$$		Aplicar $reglas[L, \mathbf{a}]$
$\$))RS$	$\mathbf{a}, \mathbf{a}))\$$	$L \to SR$	Aplicar $reglas[S, \mathbf{a}]$
$\$))R\mathbf{a}$	$\mathbf{a}, \mathbf{a}))\$$	$S \to \mathbf{a}$	Consumir **a** y desempilar.
$\$))R$	$, \mathbf{a}))\$$		Aplicar $reglas[R, ,]$
$\$))S,$	$, \mathbf{a}))\$$	$R \to ,S$	Consumir , y desempilar.
$\$))S$	$\mathbf{a}))\$$		Aplicar $reglas[S, \mathbf{a}]$
$\$))\mathbf{a}$	$\mathbf{a}))\$$	$S \to \mathbf{a}$	Consumir **a** y desempilar.
$\$))$	$))\$$		Consumir) y desempilar.
$\$)$	$)\$$		Consumir) y desempilar.
$\$$	$\$$		Entrada aceptada.

Cuadro 5: Ejecución del Parser Predictivo No-Recursivo para el Ejemplo 3

Nótese que lo emitido por la salida representa las producciones que deben utilizarse para obtener una derivación más izquierda en la gramática para la cadena suministrada en la entrada. Concretamente:

$$\begin{aligned} S &\Rightarrow (L) \\ &\Rightarrow (SR) \\ &\Rightarrow (\mathbf{a}R) \\ &\Rightarrow (\mathbf{a},S) \\ &\Rightarrow (\mathbf{a}, (L)) \\ &\Rightarrow (\mathbf{a}, (SR)) \\ &\Rightarrow (\mathbf{a}, (\mathbf{a}R)) \\ &\Rightarrow (\mathbf{a}, (\mathbf{a},S)) \\ &\Rightarrow (\mathbf{a}, (\mathbf{a}, \mathbf{a})) \end{aligned}$$

Ejemplo 4

Construyamos una tabla de parsing predictivo no recursivo para la gramática de expresiones regulares sobre $\{\mathbf{a}, \mathbf{b}\}$

$$R \to R{+}R$$

$$
\begin{aligned}
R &\rightarrow RR \\
R &\rightarrow R* \\
R &\rightarrow (R) \\
R &\rightarrow \mathbf{a} \\
R &\rightarrow \mathbf{b}
\end{aligned}
$$

preservando la precedencia y asociatividad habitual de los operadores. Comenzamos por eliminar la ambigüedad pero preservando las precedencias, de la misma manera que hicimos en el mismo ejercicio (Semana VI), quedando

$$
\begin{aligned}
S &\rightarrow S+C \\
S &\rightarrow C \\
C &\rightarrow CE \\
C &\rightarrow E \\
E &\rightarrow T* \\
E &\rightarrow T \\
T &\rightarrow (S) \\
T &\rightarrow \mathbf{a} \\
T &\rightarrow \mathbf{b}
\end{aligned}
$$

Ahora, aún cuando no hay ambigüedad, tenemos recursion por izquierda en los símbolos no terminales S y C, además de tener lados derechos con el mismo prefijo para el no terminal E. Eliminamos el primer problema y factorizamos a la izquierda para el segundo problema, quedando

$$
\begin{aligned}
S &\rightarrow CR \\
R &\rightarrow +CR \\
R &\rightarrow \lambda \\
C &\rightarrow EL \\
L &\rightarrow \lambda \\
L &\rightarrow EL \\
E &\rightarrow TU \\
U &\rightarrow * \\
U &\rightarrow \lambda \\
T &\rightarrow (S) \\
T &\rightarrow \mathbf{a} \\
T &\rightarrow \mathbf{b}
\end{aligned}
$$

y ahora calculamos

$$
\begin{aligned}
FIRST(S) = FIRST(C) = FIRST(E) = FIRST(T) &= \{\mathbf{a}, \mathbf{b}, (\} \\
FIRST(R) &= \{+, \lambda\} \\
FIRST(L) &= \{\mathbf{a}, \mathbf{b}, (, \lambda\} \\
FIRST(U) &= \{*, \lambda\} \\
FOLLOW(S) = FOLLOW(R) &= \{\$,)\} \\
FOLLOW(C) = FOLLOW(L) &= \{\$, +,)\} \\
FOLLOW(E) = FOLLOW(U) &= \{\$, \mathbf{a}, \mathbf{b}, (,), +\} \\
FOLLOW(T) &= \{\$, \mathbf{a}, \mathbf{b}, (,), +, *\}
\end{aligned}
$$

lo cual nos permite calcular la tabla

	a	**b**	(	)	+	$*$	\$
S	$S \to CR$	$S \to CR$	$S \to CR$				
R				$R \to \lambda$	$R \to +CR$		$R \to \lambda$
C	$C \to EL$	$C \to EL$	$C \to EL$				
L	$L \to EL$	$L \to EL$	$L \to EL$	$L \to \lambda$	$L \to \lambda$		$L \to \lambda$
E	$E \to TU$	$E \to TU$	$E \to TU$				
U	$U \to \lambda$	$U \to \lambda$	$U \to \lambda$	$U \to \lambda$	$U \to \lambda$	$U \to *$	$U \to \lambda$
T	$T \to \mathbf{a}$	$T \to \mathbf{b}$	$T \to (S)$				

Cuadro 6: Tabla de Pasing Predictivo No Recursivo para el Ejemplo 4

Comprobaremos su funcionamiento procesando la cadena $(\mathbf{a} + \mathbf{b}) * (\mathbf{ab} + \mathbf{b}*)*$ que pertenece al lenguaje

Pila	Entrada	Salida	Detalles
$\$S$	$(\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$		Estado inicial. Aplicar $reglas[S,($
$\$RC$	$(\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$S \to CR$	Aplicar $reglas[C,($
$\$RLE$	$(\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$C \to EL$	Aplicar $reglas[E,($
$\$RLUT$	$(\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$E \to TU$	Aplicar $reglas[T,($
$\$RLU)S($	$(\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$T \to (S)$	Consumir (y desempilar.
$\$RLU)S$	$\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$		Aplicar $reglas[S,\mathbf{a}]$
$\$RLU)RC$	$\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$S \to CR$	Aplicar $reglas[C,\mathbf{a}]$
$\$RLU)RLE$	$\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$R \to EL$	Aplicar $reglas[E,\mathbf{a}]$
$\$RLU)RLUT$	$\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$E \to TU$	Aplicar $reglas[T,\mathbf{a}]$
$\$RLU)RLU\mathbf{a}$	$\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$T \to \mathbf{a}$	Consumir **a** y desempilar.
$\$RLU)RLU$	$+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$		Aplicar $reglas[U,+]$
$\$RLU)RL$	$+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$U \to \lambda$	Aplicar $reglas[L,+]$
$\$RLU)R$	$+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$L \to \lambda$	Aplicar $reglas[R,+]$
$\$RLU)RC+$	$+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$R \to +CR$	Consumir + y desempilar.
$\$RLU)RC$	$\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$		Aplicar $reglas[C,\mathbf{b}]$
$\$RLU)RLE$	$\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$C \to EL$	Aplicar $reglas[E,\mathbf{b}]$
$\$RLU)RLUT$	$\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$E \to TU$	Aplicar $reglas[T,\mathbf{b}]$
$\$RLU)RLU\mathbf{b}$	$\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*\$$	$T \to \mathbf{b}$	Consumir **b** y desempilar.
$\$RLU)RLU$	$))*(\mathbf{ab}+\mathbf{b}*)*\$$		Aplicar $reglas[U,)]$
$\$RLU)RL$	$))*(\mathbf{ab}+\mathbf{b}*)*\$$	$U \to \lambda$	Aplicar $reglas[L,)]$
$\$RLU)R$	$))*(\mathbf{ab}+\mathbf{b}*)*\$$	$L \to \lambda$	Aplicar $reglas[R,)]$
$\$RLU)$	$))*(\mathbf{ab}+\mathbf{b}*)*\$$	$R \to \lambda$	Consumir) y desempilar.
$\$RLU$	$*(\mathbf{ab}+\mathbf{b}*)*\$$		Aplicar $reglas[U,*]$
$\$RL*$	$*(\mathbf{ab}+\mathbf{b}*)*\$$	$U \to *$	Consumir $*$ y desempilar.
$\$RL$	$(\mathbf{ab}+\mathbf{b}*)*\$$		Aplicar $reglas[L,($
$\$RLE$	$(\mathbf{ab}+\mathbf{b}*)*\$$	$L \to EL$	Aplicar $reglas[E,($
$\$RLUT$	$(\mathbf{ab}+\mathbf{b}*)*\$$	$E \to TU$	Aplicar $reglas[T,($
$\$RLU)S($	$(\mathbf{ab}+\mathbf{b}*)*\$$	$T \to (S)$	Consumir (y desempilar.
$\$RLU)S$	$\mathbf{ab}+\mathbf{b}*)*\$$		Aplicar $reglas[S,\mathbf{a}]$
$\$RLU)RC$	$\mathbf{ab}+\mathbf{b}*)*\$$	$S \to CR$	Aplicar $reglas[C,\mathbf{a}]$
$\$RLU)RLE$	$\mathbf{ab}+\mathbf{b}*)*\$$	$C \to EL$	Aplicar $reglas[E,\mathbf{a}]$
$\$RLU)RLUT$	$\mathbf{ab}+\mathbf{b}*)*\$$	$R \to TU$	Aplicar $reglas[T,\mathbf{a}]$
$\$RLU)RLU\mathbf{a}$	$\mathbf{ab}+\mathbf{b}*)*\$$	$T \to \mathbf{a}$	Consumir **a** y desempilar.
$\$RLU)RLU$	$\mathbf{b}+\mathbf{b}*)*\$$		Aplicar $reglas[U,\mathbf{b}]$
$\$RLU)RL$	$\mathbf{b}+\mathbf{b}*)*\$$	$U \to \lambda$	Aplicar $reglas[L,\mathbf{b}]$
$\$RLU)RLE$	$\mathbf{b}+\mathbf{b}*)*\$$	$L \to EL$	Aplicar $reglas[E,\mathbf{b}]$
$\$RLU)RLUT$	$\mathbf{b}+\mathbf{b}*)*\$$	$E \to TU$	Aplicar $reglas[T,\mathbf{b}]$
$\$RLU)RLU\mathbf{b}$	$\mathbf{b}+\mathbf{b}*)*\$$	$T \to \mathbf{b}$	Consumir **b** y desempilar.
$\$RLU)RLU$	$+\mathbf{b}*)*\$$		Aplicar $reglas[U,+]$
$\$RLU)RL$	$+\mathbf{b}*)*\$$	$U \to \lambda$	Aplicar $reglas[L,+]$
$\$RLU)R$	$+\mathbf{b}*)*\$$	$L \to \lambda$	Aplicar $reglas[R,+]$
$\$RLU)RC+$	$+\mathbf{b}*)*\$$	$R \to +CR$	Consumir + y desempilar.
$\$RLU)RC$	$\mathbf{b}*)*\$$		Aplicar $reglas[C,\mathbf{b}]$
$\$RLU)RLE$	$\mathbf{b}*)*\$$	$C \to EL$	Aplicar $reglas[E,\mathbf{b}]$
$\$RLU)RLUT$	$\mathbf{b}*)*\$$	$E \to TU$	Aplicar $reglas[T,\mathbf{b}]$
$\$RLU)RLU\mathbf{b}$	$\mathbf{b}*)*\$$	$T \to \mathbf{b}$	Consumir **b** y desempilar.
$\$RLU)RLU$	$*)*\$$		Aplicar $reglas[U,*]$
$\$RLU)RL*$	$*)*\$$	$U \to *$	Consumir $*$ y desempilar.
$\$RLU)RL$	$))*\$$		Aplicar $reglas[L,)]$
$\$RLU)R$	$))*\$$	$L \to \lambda$	Aplicar $reglas[R,)]$
$\$RLU)$	$))*\$$	$R \to \lambda$	Consumir) y desempilar.
$\$RLU$	$*\$$		Aplicar $reglas[U,*]$
$\$RL*$	$*\$$	$U \to *$	Consumir $*$ y desempilar.
$\$RL$	$\$$	8	Aplicar $reglas[L,\$]$
$\$R$	$\$$	$L \to \lambda$	Aplicar $reglas[R,\$]$
$\$$	$\$$	$R \to \lambda$	Aceptar.

Cuadro 7: Ejecución del Parser Predictivo No Recursivo para el Ejemplo 4

Nótese que lo emitido por la salida representa las producciones que deben utilizarse para obtener una derivación más izquierda en la gramática para la cadena suministrada en la entrada. Concretamente:

$$
\begin{aligned}
S &\Rightarrow CR \\
&\Rightarrow ELR \\
&\Rightarrow TULR \\
&\Rightarrow (S)ULR \\
&\Rightarrow (CR)ULR \\
&\Rightarrow (ELR)ULR \\
&\Rightarrow (TULR)ULR \\
&\Rightarrow (\mathbf{a}ULR)ULR \\
&\Rightarrow (\mathbf{a}LR)ULR \\
&\Rightarrow (\mathbf{a}R)ULR \\
&\Rightarrow (\mathbf{a}+CR)ULR \\
&\Rightarrow (\mathbf{a}+ELR)ULR \\
&\Rightarrow (\mathbf{a}+TULR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b}ULR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b}LR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b}R)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*LR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*ELR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*TULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(S)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(CR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(ELR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(TULR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{a}ULR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{a}LR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{a}ELR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{a}TULR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}ULR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}LR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}R)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+CR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+ELR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+TULR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}ULR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*LR)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*R)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)ULR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*LR \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*R \\
&\Rightarrow (\mathbf{a}+\mathbf{b})*(\mathbf{ab}+\mathbf{b}*)*
\end{aligned}
$$