ANALIZADOR SINTACTICO

Por Andrés Morales Martos y Miguel Angel Galán Montalvo

FACULTAD DE INFORMÁTICA
BIBLIOTECA

Introducción

Dentro de las diversas técnicas habidas para la realización de un analizador sintáctico usaremos la denominada de PREFERENCIA DE OPERADORES, dado que la gramática a emplear se encuentra en forma ideal para realizar con ella dicho proceso.

Teóricamente no explicaremos el método, puesto que -- prácticamente a medida que vayamos desarrollándolo, dicho estudio refleja todo el contenido del Analizador.

El método empleado en comparación con el que se utiliza usando una gramática de Preferencia Simple difiere del mismo - en que aquí sólo usaremos relaciones entre operadores, entendiendo por tales los terminales de la gramática.

Entenderemos por operandos las clases no terminales.

1. Gramática

1.1. Gramática dada

Esta gramática cumple las condiciones de una gramática de operadores, al no existir ninguna regla de la forma

$$U ::= \ldots V W \ldots \qquad V,W \in V_N$$

Concluiremos en su momento, que es una gramática de - preferencia de operadores cuando comprobemos que entre cada dos de sus terminales existe a lo sumo una relación de

las que posteriormente definiremos.

GRAMATICA

< PROGRA > → < INSTRU >

< PROGRA > ; < INSTRU >

< INSTRU > → < DECLARA >

< NO DECIARA >

< CONTROL >

< DECLARA > → REAL < LETRA >

INTEGER < LETRA >

LOGICAL < LETRA >

< NO DECLARA > → < ASIGNA >

< CONDIC >

< ASIGNA > → < LETRA > = < EXPRE >

< EXPRE > → < EXPRE > .OR. < EXPRE 1 >

< EXPRE 1 >

< EXPRE 1 > → < EXPRE 1 > .AND. < EXPRE 2 >

< EXPRE 2 >

< EXPRE 2 > → < EXPRE 2 > .RO. < EXPRE 3 >

< EXPRE 3 >

< EXPRE 3 > → < EXPRE 3 > + < TERM >

< EXPRE 3 > - < TERM >

< TERM >

< TERM > → < TERM > * < FACTOR >

< TERM > / < FACTOR >

< FACTOR >

< FACTOR > → (< EXPRE >)

< LETRA >

.NOT. < FACTOR >

< NUMERO >

< CONDIC > → < IFCLAR > THEN < ASIGNA > ELSE < NO DECLARA >

< IFCLAR > → IF < EXPRE >

< CONTROL > → END

< LETRA > → A

B

.
.
.

Z

< NUMERO > → & CUALQUIER NUMERO &

1.2. Codificación de la Gramática

Para el mejor manejo de la gramática en lo sucesivo,- asignaremos los siguientes códigos numéricos a sus clases terminales y no terminales.

Operandos (no terminales).

< PROGRA >..... 1

< INSTRU >..... 2

< DECLARA >..... 3

< NO DECLARA >..... 4

< CONTROL >..... 5

< LETRA >..... 6

< ASIGNA >..... 7

< CONDIC >..... 8

< EXPRE >..... 9

< EXPRE 1 >..... 10

< EXPRE 2 >..... 11

< EXPRE 3 >..... 12

< TERM >..... 13

< FACTOR >..... 14

< IFCLAR >..... 15

< NUMERO >..... 16

Operadores (terminales).

; 17

REAL 18

INTEGER 19

LOGICAL 20

= 21

.OR. 22

. AND. 23

.RO. 24

+ 25

- 26

* 27

/ 28

(..... 29

) 30

.NOT. 31

THEN 32

ELSE 33

IF 34

END 35

A,B,...,Z 36

CUALQUIER NUMERO 37

2. Matrices de la Gramática

2.1. Matriz relación primero.

Como primera matriz necesaria para la obtención de -- las relaciones entre operadores, aparece la "matriz relación primero".

Construiremos la matriz con arreglo al siguiente es--

quema.

En la fila I columna J, aparecerá un 1 si existe -- una regla

$I \longrightarrow J \ldots$ $\quad I \in V_N \quad J \in V$

no es necesario ampliar nuestra definición ya que en esta gramática no hay clases que conduzcan al vacio.

Aparecerá un 0 si no existe una regla de este tipo.

Daremos más tarde el listado de dicha matriz.

2.2. Matriz relación última.

Muy similar a la anterior.

El esquema que usaremos para la construcción de esta matriz se apoya en la relación último, al igual que la anterior se apoyaba en la relación primero.

Dado el carácter práctico del trabajo, preferimos en lo anterior y sucesivo, definir directamente las matrices de trabajo, sin dar el paso intermedio de la formaliza--- ción en la definición de las relaciones.

Así pues en este caso, aparecerá en la fila I y columna J, un 1 si existe una regla

$I \longrightarrow \ldots J$ $\quad I \in V_N \quad J \in V$

y aparecerá un 0 en caso contrario.

Daremos más tarde el listado de dicha matriz.

2.3. Matriz primeros.

Aplicando el teorema de WARSHALL, según programa -- que daremos, calculamos el cierre transitivo de la matriz relación primera.

Daremos un listado oportunamente.

2.4. Matriz últimos.

Análogo al caso anterior y aplicando el teorema de - WARSHALL calculamos el cierre transitivo de la ma-- triz relación última.

Publicaremos oportunamente su listado.

2.5. Matriz igual.

Construiremos esta matriz con arreglo al siguiente - esquema.

En la fila I, columna J aparecerá un 1, si existe - una regla

$$U \longrightarrow \ldots IJ \ldots$$

donde $U \in V_N$ $I, J \in V$

aparecerá un 0 en caso contrario.

Oportunamente daremos su listado.

2.6. Matriz primer término.

En la fila I, columna J, aparecerá un 1 si existe una regla

$I \longrightarrow J\ldots$ $I \in V_N$ $J \in V_T$

o bien una regla

$$I \longrightarrow NJ\ldots$$

donde $N \in V_N$.

Aparecerá un 0 en caso contrario.

Publicaremos su listado oportunamente.

2.7. Matriz último término.

En la fila I columna J, aparecerá un 1 si existe -- una regla

$I \longrightarrow \ldots J \qquad I \in V_N \qquad J \in V_T$

o bien una regla

$I \longrightarrow \ldots JN \qquad N \in V_N$

aparecerá un 0 en caso contrario.

Oportunamente daremos su listado.

3. Matriz de relaciones entre operadores.

3.1. Relación $\lessdot$

Diremos que R es $\lessdot$ que S si y sólo si existe una regla

U::=...RW... donde $W \Rightarrow^{+} S\ldots$ ó $W \Rightarrow^{+} VS\ldots$

siendo RS terminales $V,W \in V_N$

A nivel práctico la matriz de la relación menor la -- obtenemos como el producto booleano de las matrices $\doteq$,- primeros y primer término, es decir:

$\lessdot = (\doteq)\,(\text{PRIMEROS})\,(\text{PRIMER TERMINO})$

Aparecerá en la fila I, columna J de la matriz un 1 - cuando entre dos terminales se dé esta relación, y un 0 - en caso contrario.

3.2. Matriz $\gtrdot$

Diremos que R es $\gtrdot$ que S si y sólo si existe una re gla

U::=...WS... donde

$W \Rightarrow + \ldots R$ ó $W \Rightarrow + \ldots RV$ R, S terminales

$W, V \in V_N$

A nivel práctico obtendremos esta matriz como el producto matricial booleano de la transpuesta del producto - de la matriz de último, por la de último término. Y esta transpuesta por la matriz igual. Es decir:

$\gtrdot$ = TRANSP ((ULTIMOS)(ULTIMO TERMINO)) ($\doteq$)

Aparecerá en la fila I, columna J de la matriz un 1, cuando entre dos terminales se dé esta relación y un 0 en caso contrario.

3.3. Relación $\doteq$

Diremos que $R \doteq S$, R, S terminales si y sólo si existe una regla

$U ::= \ldots RS \ldots$ ó $U ::= \ldots RVS \ldots$ $V \in V_N$

A nivel práctico obtendremos esta matriz por observación directa de las reglas de la gramática.

Aparecerá en la fila I, columna J de la matriz un 1,- cuando entre dos terminales se dé esta relación y un 0 en caso contrario.

4. Algoritmo de reconocimiento.

Usaremos como principio y final de cada sentencia un delimitador # . El delimitador será menor que cualquier otro terminal y cualquier otro terminal será mayor que él.

Dado que este método se encuadra dentro del tipo de - los analizadores bottom-up, dada una frase sentencial habremos

de irla reduciendo hasta llegar al axioma.

Siguiendo el estudio teórico del libro del Profesor - Gries, damos el siguiente algoritmo análogo al que se da en la figura 6.4. del citado libro.

Hacemos uso de una pila S, con un contador i, una cadena de símbolos de entrada $T_1 \ldots T_n$ equipados con un contador k, una variable entera j y dos variables R y Q.

(Ver página 10).

El principal objeto de este algoritmo es el de ir localizando partes derechas de las diferentes reglas gramaticales, e ir reduciéndolas a sus respectivas partes izquierdad.

Es fundamental para ello, la localización de lo que - denominaremos primera frase.

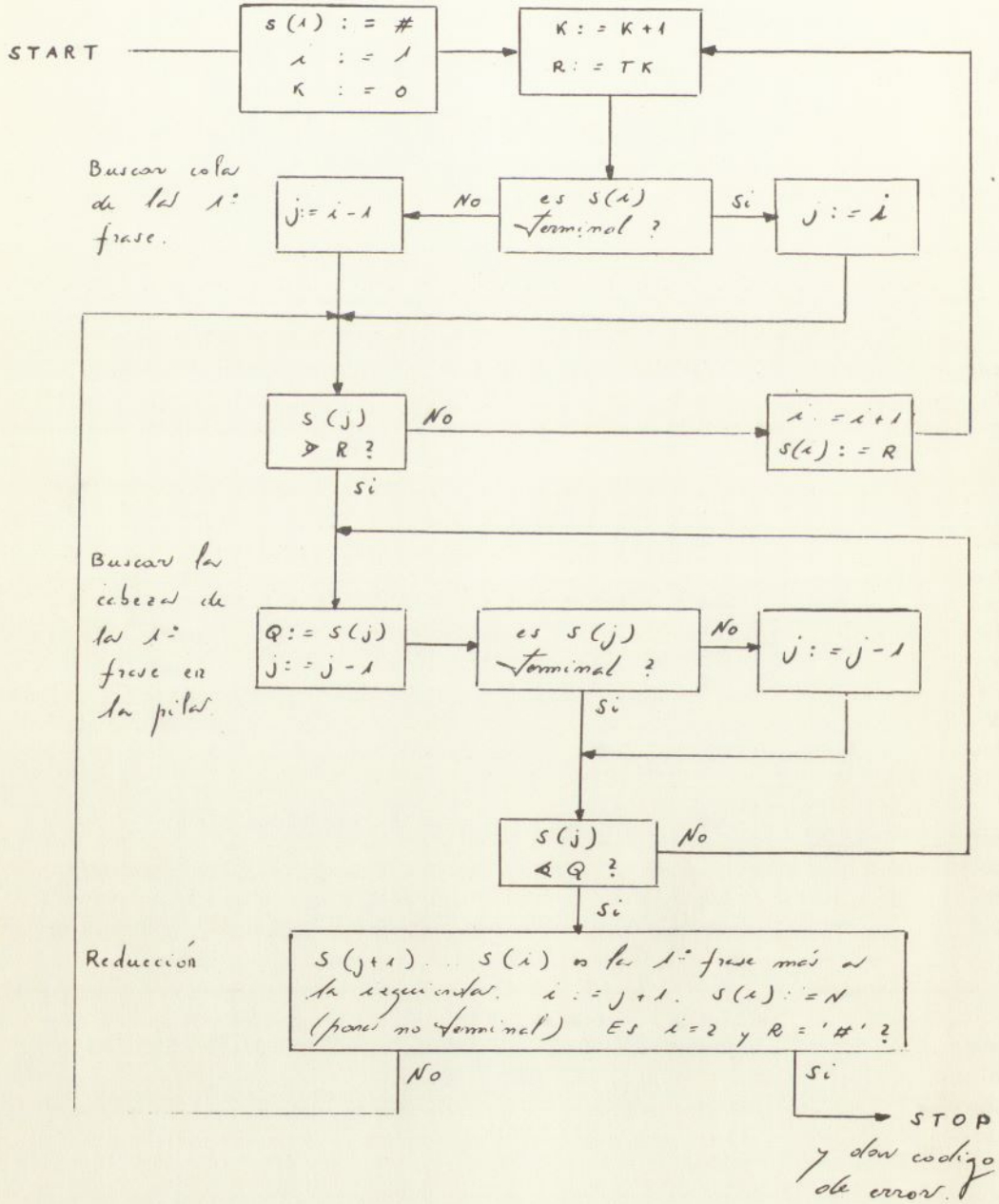
Evidentemente para el uso de este algoritmo situaremos en la memoria, la gramática en estudio, de otro modo no sería posible la localización referida de la correspondiente parte izquierda.

Daremos listados de todas las matrices usadas, como - los programas de cálculo de cierres transitivos, de producto - de las mismas, así como el programa que lleva a cabo el algoritmo descrito.

Las entradas del programa estarán constituidas por -- los códigos numéricos asociados a las terminales de la gramática.

La salida del mismo será un mensaje de aprobación, caso de que la entrada se ajuste a las reglas de la gramática, - o bien un mensaje de error y una parada en caso contrario.

Figura 6.4


START
s(1) := #
i := 1
K := 0
K := K+1
R := TK
Buscar cola de la 1ª frase.
j := i-1
No
es s(i) terminal ?
Si
j := i
s(j) > R ?
No
i := i+1
s(i) := R
Si
Buscar la cabeza de la 1ª frase en la pila.
Q := s(j)
j := j-1
es s(j) terminal ?
No
j := j-1
Si
s(j) < Q ?
No
Si
Reducción
s(j+1) ... s(i) es la 1ª frase más a la izquierda. i := j+1. s(i) := N (para no terminal) Es i=2 y R = '#' ?
No
Si
STOP
y dar codigo de error.