PROCESADORES DE LENGUAJE

Ingeniería Informática
Primer curso de segundo ciclo

Departamento de Informática y Análisis Numérico
Escuela Politécnica Superior
Universidad de Córdoba
Curso académico 2013 – 2014

Relación de ejercicios nº 4: **ANÁLISIS SINTÁCTICO ASCENDENTE**

1. La siguiente gramática genera la tabla SLR que se indica

   P = {
   1) E → T
   2) E → E **+** T
   3) T → P
   4) T → T ***** P
   5) P → F
   6) P → F **^** P
   7) F → (E)
   8) F → **identificador**
   9) F → **número**
   }

	Acción									Ir a			
Estado	**+**	*****	**^**	**(**	**)**	**identificador**	**número**	**$**		E	T	P	F
0				d5		d 6	d 7			1	2	3	4
1	d8							**ACEPTAR**					
2	r1	d9	r1	r1	r1	r1	r1	r1					
3	r3	r3	r3	r3	r3	r3	r3	r3					
4	r5	r5	d10	r5	r5	r5	r5	r5					
5				d5		d6	d7			11	2	3	4
6	r8	r8	r8	r8	r8	r8	r8	r8					
7	r9	r9	r9	r9	r9	r9	r9	r9					
8				d5		d6	d7				12	3	4
9				d5		d6	d7					13	4
10				d5		d6	d7					14	4
11	d8				d15								
12	r2	d9	r2	r2	r2	r2	r2	r2					
13	r4	r4	r4	r4	r4	r4	r4	r4					
14	r6	r6	r6	r6	r6	r6	r6	r6					
15	r7	r7	r7	r7	r7	r7	r7	r7					

   a. Analiza las expresiones:
      - a ^2 + b

- a ^(2 + b)

b. Calcula las clausuras de la colección canónica de LR(0) – elementos:

I_0 = clausura { E'→ • E })
I_1 = clausura {E ' → E • , E → E • + T })
I_2 = clausura { E → T • , T → T • * P })
I_3 = clausura { T → P • })
I_4 = clausura { P → F • , P → F • ^ P })
I_5 = clausura { F → (• E) })
I_6 = clausura ({F → **identificador** • })
I_7 = clausura ({F → **número** • })
I_8 = clausura ({ E → E +• T })
I_9 = clausura ({ T → T *• P })
I_{10} = clausura ({P → F ^• P })
I_{11} = clausura ({E → E • + T , F → (E •) })
I_{12} = clausura ({ E → E + T • , T → T • * P })
I_{13} = clausura ({ T → T * P • })
I_{14} = clausura ({ P → F ^ P • })
I_{15} = clausura ({ F → (E) • })

c. Utiliza la colección canónica de LR(0) - elementos para completar la tabla de análisis SLR con funciones de recuperación de errores para aplicar el método de "nivel de frase".
d. Analiza la siguiente expresión errónea: **((a ++ b * * 2)**

2. La siguiente gramática permite generar **asignaciones múltiples** en el lenguaje C:

P = {

1) S → S A
2) S → A
3) A → **identificador =** L ;
4) L → **número**
5) L → **identificador =** L

}

Por ejemplo:

a = b = 1;
c = 2;

a. Construye la colección canónica de **LR(0)-elementos**
b. Dibuja el **autómata** que reconoce los prefijos viables.
c. Construye la tabla de análisis sintáctico **SLR**
d. Utiliza el método recuperación de errores de **"nivel de frase"** para completar la tabla SLR.
e. Utiliza la tabla SLR para analizar la siguiente declaración errónea:

= a = = b 1 ;

3. Considera la siguiente gramática de contexto libre

P = {

1) S → S D
2) S → ε

3) D → ***enum identificador*** { L } ;
4) L → ***identificador***
5) L → L , ***identificador***
}

Esta gramática puede generar algunas "enumeraciones" del lenguaje de programación C:

enum color {blanco, negro, amarillo};

a. Construye la colección canónica de **LR(0)-elementos**
b. Dibuja el **autómata** que reconoce los prefijos viables.
c. Construye la tabla de análisis sintáctico **SLR**
d. Utiliza el método recuperación de errores de **"nivel de frase"** para completar la tabla SLR.
e. Utiliza la tabla SLR para analizar la siguiente declaración errónea:

color { blanco amarillo ;

4. Dada la gramática

P = {
S → L **punto** L | L
L → L **dígito** | **dígito**
}

donde **punto** representa el punto decimal (".")

a. Construye la tabla de análisis sintáctico LR(1) - canónico.
b. Construye la tabla de análisis sintáctico LALR(1).
c. Añade a la tabla de análisis LALR(1) funciones de recuperación de errores para poder aplicar el método de nivel de frase.
d. Analiza la cadena errónea: **3..12.1.**

5. Dada la gramática

P = {
S → **función identificador (L) : T**
L → T **identificador ,** L
L → T **identificador**
L → ε
T → **real**
T → **carácter**
}

a. Construye la tabla de análisis sintáctico LALR(1).
b. Analiza la sentencia:
función media (real x, real y) : real

c. Dada la sentencia errónea:
carácter función error real valor ,) : real

- Utiliza el método de “modo de pánico” para analizar esta sentencia errónea.
- Utiliza el método de “nivel de fase” para analizar esta sentencia errónea.

6. Análisis sintáctico **ascendente** de prototipos de funciones en C:

   a. Diseña una gramática que permita generar los prototipos de las funciones de C que sólo utilizan los tipos **int**, **char** y punteros a **int** o **char**.

   b. Construye la tabla de análisis sintáctico LALR y analiza las siguientes sentencias:
      - **int tasa ( );**
      - **char * letras (int , char **, char);**

   c. Utiliza el método de nivel de fase para analizar la siguiente sentencia errónea:
      - **int char consultar ((int , , ;**

7. Demuestra que una gramática LR no puede ser ambigua.

8. Demuestra que nunca se van a producir errores al consultar la tabla “**ir_a**” durante un análisis sintáctico LR.