Compiladores
Práctica 3
Análisis Sintáctico de Descenso Recursivo
Nombre_____
Grupo _____

José Sánchez Juárez

11 de abril de 2016

1. Objetivo

Construir un analizador sintáctico descendente del tipo descenso recursivo.

2. Analizador Sintáctico de Descenso Recursivo

El analizador sintáctico de descenso recursivo, se construye de la siguiente manera: es necesario que la gramática sea del tipo LL(1), por lo que se deben cumplir dos reglas:

1. Si la producción es de la forma $A \to \sigma_1 \mid \sigma_2 \mid \cdots \mid \sigma_n$, se debe cumplir con la siguiente condición: $PRIMERO(\sigma_1) \cap PRIMERO(\sigma_2) \cap \cdots \cap PRIMERO(\sigma_n) = \Phi$.

2. Si la producción es de la siguiente forma $X \to \sigma \mid \lambda$, se aplica la siguiente condición $PRIMERO(X) \cap SIGUIENTE(X) = \Phi$

Cuando una gramática no es LL(1), se debe seguir el siguiente procedimiento para programarla:

1. Se aplican las reglas UNO y DOS, para determinar si es LL(1).

2. Se debe determinar si las producciones de la gramática tienen recursividad izquierda. Esto se comprueba si las producciones tienen la siguiente forma $A \to A\alpha \mid \beta$.

3. Se debe eliminar la recursividad izquierda, haciendo la siguiente transformación: $A \to \beta A'$, $A' \to \alpha A' \mid \lambda$.

4. Si las alternativas son de la forma $A \to \sigma_1 \mid \sigma_2 \mid \cdots \mid \sigma_n$, pero se intersectan entonces para corregir se debe factorizar.

5. Una vez que la gramática es LL(1) se procede a construir el analizador sintáctico descendente LL(1).

Después de que la gramática ya es LL(1), la gramática se debe aumentar. Esto es agregando una producción al principio de la gramática con la finalidad de asegurar la aceptación. Se calculan los primeros y siguientes de los símbolos no terminales. Y por último se construyen las funciones para cada uno de los no terminales que existan.

Algorithm .1: Analizador sintáctico de descenso recursivo

```
1  PROGRAMA
2  Fun Error();
3  Fun SigPal();
4  Fun Analizar N_0();
5  Fun Analizar N_1();
   ⋮
6  Fun Analizar N_m();
7  Iniciar PROGRAMA
8  SigPal();
9  Analizar N_0();
10 Fin PROGRAMA
```

Donde N_0, N_1, ..., N_m son los símbolos no terminales. Y N_0 es el símbolo inicial, el de la producción aumentada.

Se tiene la siguiente gramática:

$$S \to aAb$$
$$A \to cB$$
$$B \to d|\lambda$$

La gramática aumentada es:

$$S' \to S$$
$$S \to aAb$$
$$A \to cB$$
$$B \to d|\lambda$$

Las funciones de la gramática 1, se obtienen de la siguiente manera. Para el símbolo inicial S' se tiene la función:

Algorithm .2: Función Analizar $S'()$, del símbolo inicial

```
1 Fun Analizar S'();
2 Analizar S();
3 if pal == $ then
4 |   EsEnPan(Sintaxis Correcta);
5 else
6 |   EsEnPan(Sintaxis Incorrecta);
7 Fin Fun;
```

Para el símbolo S se tiene la función:

Algorithm .3: Función Analizar $S()$, del símbolo S

```
1 Fun Analizar S();
2 if pal == a then
3 |   Palabra ← SigPal();
4 |   Analizar A();
5 |   Palabra ← SigPal();
6 else
7 |   Error();
8 Fin Fun;
```

Para el símbolo A se tiene la función:

Algorithm .4: Función Analizar $A()$, del símbolo A

```
1 Fun Analizar A();
2 if pal == c then
3 |   Palabra ← SigPal();
4 |   Analizar B();
5 else
6 |   Error();
7 Fin Fun;
```

Para el símbolo B se tiene la función:

Algorithm .5: Función Analizar $B()$, del símbolo B

```
1 Fun Analizar B();
2 if pal == d then
3 |   Palabra ← SigPal();
4 else if pal == b then
5 |   Return(true);
6 else
7 |   Error();
8 Fin Fun;
```

3. Problemas

Implementar un analizador sintáctico de descenso recursivo con la siguiente gramática G:

$$S \to uBDz$$
$$B \to wC$$
$$C \to vC$$
$$C \to \lambda$$
$$D \to EF$$
$$E \to y$$
$$E \to \lambda$$
$$F \to x$$
$$F \to \lambda$$

4. Procedimiento

Lo siguiente es bansándose en la gramática G:

1. Justificar si la gramática G es LL(1).
2. Si no lo es convertirla a gramática LL(1).
3. Después de convertirla justificar si es LL(1).
4. Construir la tabla de primeros y siguientes.
5. Escriba las justificaciones de la generación de las cadenas $uwvyxz$ por la gramática G.
6. Construir el analizador Sintáctico de descenso recursivo de la gramática G, implementando las funciones de los símbolos no terminales.

5. Actividades

Realice las siguientes actividades:

1. Investigar en internet las gramáticas libres de contexto, para determinar la forma de sus producciones.
2. Investigue que es un árbol de derivación.
3. Investigue que es una árbol de análisis.

4. Convertir la gramática en gramática LL(1), tal como se expresa en la sección de procedimiento.

5. Generar la secuencia de caracteres $uwvyxz$, con la gramática G. Aplicando la lectura deonde se usa: Entrada y función.

6. Justificar si se genera la siguiente cadena $uwvvvyxxz$ con la gramática G. Aplicando la lectura donde se usa: Entrada y función.

7. Construya el analizador de descenso recursivo para la gramática G