
Instituto Politécnico Nacional
Escuela Superior de Cómputo

Teoría Computacional



Practica 7: Gramáticas Independientes del Contexto



Alumna: Ramírez Galindo Karina

Profesor: Rosas Trigueros Jorge Luis

Fecha de realización: 11-11-2019

Fecha de entrega: 02-12-2019

Marco Teórico

🌈 **GRAMATICA INDEPENDIENTE DEL CONTEXTO (GIC)**

Una gramática independiente del contexto (GIC) es una 4-tupla $G=(N,\Sigma,S,P)$ [1] [2]

Donde:

- Σ es el alfabeto de entrada
- N es una colección finita de terminales
- S es un no terminal determinado (símbolo inicial)
- $P \subseteq N \times (N \cup \Sigma)^*$ es un conjunto de producciones

NOTA: El lenguaje generado por una GIC se denota por LCG y se llama lenguaje independiente del contexto (LIC) [3]

Por ejemplo, puesto que toda gramática regular es una GIC, se tiene que todo lenguaje regular es un LIC. [3]

Al igual que una gramática regular, una GIC es una forma de probar cómo se generan cadenas en un lenguaje. Como con las gramáticas regulares, usaremos la

notación \Rightarrow para indicar el acto de generar como opuesto a \rightarrow el cual es parte de una regla de producción. Cuando derivamos una cadena, los no terminales representan la parte de la cadena que todavía no se ha generado. En el caso de las gramáticas regulares, la parte de la cadena no generada siempre aparece al final. En las GIC que no son gramáticas regulares, puede haber más de un trozo no generado y pueden aparecer en cualquier lugar de la cadena. Cuando la derivación se completa, todos los trozos no generados habrán sido sustituidos por cadenas (posiblemente vacías) de símbolos terminales. [1] [2]




EJEMPLO:

Considérese la GIC

$$S \rightarrow aSb \mid \epsilon$$

La inducción sobre n prueba que esta gramática independiente del contexto genera el lenguaje independiente del contexto $\{a^n b^n \mid n \geq 0\}$. Este lenguaje no es regular. Por tanto, hay lenguajes independientes del contexto que no son lenguajes regulares. Es decir, el conjunto de los lenguajes independientes del contexto contiene al conjunto de los lenguajes regulares. [3]

Material y equipo

-  Sistema Operativo Ubuntu
-  Python 3
-  Ply.lex

Desarrollo de la práctica

Objetivo: diseñar y programar gramáticas independientes del contexto en python.

Ejercicios:

Diseñar una gramática independiente del contexto y probarla con PLY para:

1) $\{a^n b^n \mid n \geq 0\}$

Posibles cadenas:

$$w = \{\epsilon, ab, aabb, aaabbb, aaaabbbb, \dots\}$$

La gramática queda de la siguiente forma:

$$S \rightarrow aSb \mid \epsilon$$

```
Abrir  [icon]
1 tokens = ('a','b');
2 t_a = r'a'
3 t_b = r'b'
4
5 def t_error(t):
6     print("CARACTER ILEGAL ", t.value[0])
7     t.lexer.skip(1)
8
9 import ply.lex as lex
10 lex.lex()
11
12 def p_S(p):
13     ''' S : a S b
14         | empty '''
15     pass
16
17
18 def p_empty(p):
19     'empty :'
20     pass
21
22 s= ' '
23
24 def p_error(p):
25     global s
26     if p:
27         print("ERROR DE SINTAXIS EN", p.value)
28         print(s,"NO ESTA EN EL LENGUAJE")
29     else:
30         print("ERROR DE SINTAXIS EN EOF")
31         print(s,"NO ESTA EN EL LENGUAJE")
32
33
34 import ply.yacc as yacc
35 yacc.yacc()
36
37 while 1:
```

Imagen 1

Código del ejercicio 1 – Parte 1

```
36
37 while 1:
38     try:
39         s = input('>')
40     except EOFERROR:
41         break
42     if not s:
43         continue
44     t = yacc.parse(s)
45
46
47
```

Imagen 2

Código del ejercicio 1 – Parte 2

```
ubuntu@ubuntu:~/Documentos/p7$ python3 ejercicio1.py
Generating LALR tables
>ab
>aabb
>aaabbb
>aaaabbbb
>aaaaabbbbb
>aaaaaabbbbb
>aba
ERROR DE SINTAXIS EN a
aba NO ESTA EN EL LENGUAJE
>abb
ERROR DE SINTAXIS EN b
abb NO ESTA EN EL LENGUAJE
>aab
ERROR DE SINTAXIS EN EOF
aab NO ESTA EN EL LENGUAJE
>
```

Imagen 3

Compilación del ejercicio 1

$$2) \{a^m b^n c^p d^q \mid m + n \geq p + q\}$$

Posibles cadenas:

$$w = \{\varepsilon, a, b, ac, ab, bb, aa, bd, abc, \dots\}$$

La gramática queda de la siguiente forma:

$S \rightarrow aSd \mid A \mid B$

$A \rightarrow Ad \mid C$

$B \rightarrow aBc \mid aB \mid C$

$C \rightarrow bCc \mid bC \mid \varepsilon$

```

Abrir  [icon]
1 tokens = ('a','b','c','d');
2 t_a = r'a'
3 t_b = r'b'
4 t_c = r'c'
5 t_d = r'd'
6
7 def t_error(t):
8     print("CARACTER ILEGAL ", t.value[0])
9     t.lexer.skip(1)
10
11 import ply.lex as lex
12 lex.lex()
13
14 def p_S(p):
15     ''' S : a S d
16           | A
17           | B '''
18     pass
19
20 def p_A(p):
21     ''' A : A d
22           | C '''
23     pass
24
25 def p_B(p):
26     ''' B : a B c
27           | a B
28           | C '''
29
30 def p_C(p):
31     ''' C : b C c
32           | b C
33           | empty '''
34     pass
35
36 def p_empty(p):
37     'empty :'
```

Imagen 4

Código del ejercicio 2 – Parte 1

```

38     'empty : '
39     pass
40 s= ' '
41
42 def p_error(p):
43     global s
44     if p:
45         print("ERROR DE SINTAXIS EN", p.value)
46         print(s,"NO ESTA EN EL LENGUAJE")
47     else:
48         print("ERROR DE SINTAXIS EN EOF")
49         print(s,"NO ESTA EN EL LENGUAJE")
50
51 import ply.yacc as yacc
52 yacc.yacc()
53
54 while 1:
55     try:
56         s = input('>')
57     except EOFERROR:
58         break
59     if not s:
60         continue
61     t = yacc.parse(s)
62
63
```

Imagen 5

Código del ejercicio 2 – Parte 2

```

ubuntu@ubuntu:~/Documentos/p7$ python3 ejercicio2.py
>a
>b
>ac
>ab
>
>bb
>aa
>bd
>abc
>aabd
ERROR DE SINTAXIS EN EOF
aabd NO ESTA EN EL LENGUAJE
>add
ERROR DE SINTAXIS EN EOF
add NO ESTA EN EL LENGUAJE
>
```

Imagen 2

Compilación del ejercicio 2

Conclusiones

Para esta práctica no tuve mucho problema, ya que a diferencia de las gramáticas regulares, las gramáticas independientes del contexto pueden tener más de un terminal, lo cual me facilitó un poco poder diseñar y programar las gramáticas presentadas en esta práctica.

Referencias

- [1] D. G. P. Luis Migel Pardo Vasallo, «Teoria de Automatas Finitos,» [En línea]. Available: https://ocw.unican.es/pluginfile.php/1516/course/section/1946/2-1_Introduccion.pdf.
- [2] J. E. Hopcroft, R. Motwani y J. D. Ullman, Introducción a la teoría de autómatas, lenguajes y computación, Madrid: PEARSON EDUCACIÓN S.A, 2007.
- [3] D. Kelley, Teoría de autómatas y lenguajes formales, Madrid: PEARSON EDUCACIÓN, S. A, 1995.