INSTITUTO POLITECNICO NACIONAL

Escuela Superior de Cómputo

PRACTICA 7

Autor:
Gomez Hernandez Alan Javier

26 diciembre, 2021—

PALINDROMES

RESUMEN

Realizar un programa que construya palindromes de un lenguaje binario. El lenguaje deberá solicitar únicamente la longitud del palindrome a calcular, de esta manera el programa deberá construir el palindrome de manera aleatoria. La longitud máxima que podría alcanzar un palindrome será de 100,000 caracteres. La salida del programa se irá a un archivo de texto y ahí especificarán qué regla se selecciono y la cadena resultante hasta llegar a la cadena final. El programa deberá ofrecer dos opciones: que el usuario defina la longitud del palindrome o que lo genere todo de manera automática. La gramática libre de contexto que construye palindromes, se define con las siguientes reglas de producción.

- **(1) P -¿ e**
- **(2) P -¿ 0**
- **(3) P -¿ 1**
- **(4) P -¿ 0P0**
- **(5) P -¿ 1P1**

INTRODUCCION

La historia de la complejidad computacional, en sus casi 40 años (en 2011 se cumplen 40 años de la publicacion del historico articulo de Cook [7] y la deÖnicion de la clase NP), es la historia de una elusiva busqueda de cotas inferiores. Puede conjeturarse que la mayor parte de las cotas superiores obtenidas hasta el momento son ajustadas y que son nuestras debiles cotas inferiores las responsables de la gigantesca brecha existente entre las cotas obtenidas hasta el momento. Son muy pocos los problemas para los cuales la complejidad computacional puede brindar un clara estimacion-cuanticacion de su complejidad intrinseca, uno de estos pocos problemas es el reconocimento de palindromos que denotaremos con el simbolo

$$L_{pal}$$

El palíndromo es una palabra o frase que se lee igual de derecha a izquierda que de izquierda a derecha. Para el palíndromo el camino de ida y el camino de vuelta es uno y el mismo.

Este problema fue parte de la introducción a las gramáticas libres de contexto o Context Free Grammars con el objetivo de generar cadenas que fueran parte del lenguaje de los palíndromos formados por ceros y unos como por ejemplo

las cadenas 0011, 1111, 11011 entre otras. Dicho de otra forma, una cadena w forma parte del

$$L_{pal}$$

si y sólo si

$$w = w^R$$

. Para generar el pal´ındromo se utilizo la siguiente GIC [1]

$$G_{pal} = (P, 0, 1, A, P)$$

Donde A representa el conjunto de las siguientes 5 producciones.

- (1) P -¿ e
- (2) P -¿ 0
- (3) P -¿ 1
- (4) P -¿ 0P0
- (5) P -¿ 1P1

El programa cuenta con modo manual y automático en los cuales se introduce la longitud $0 \leq n \leq 1000$ de la cadena que se desea obtener. Y se muestra el proceso de producción en pantalla y un archivo de texto.

DESARROLLO

0.1 MAIN-PALINDROMO (CODIGO)

```
# -*- coding: utf-8 -*-
from __future__ import print_function
from palindromo import palindromo
import random

separador = '*-'*20

def main():
    continuar = True
    while continuar:
        opcion = imprimir_menu()
        if opcion == 1:
            entrada_consola()
        elif opcion == 2:
            entrada_automatico()
        else:
            break
        print('=' * 100)
        opcion = input("REALIZAR OTRA [s/n]: ")
        if opcion.lower() != 's':
            continuar = False

    print('Saliendo del programa...')

def imprimir_menu():
    print("""\n\nGramatica libre de contexto Gpal = ({P},{0,1},A,P)
    Donde A es:
    1. P -> e
    2. P -> 0
    3. P -> 1
    4. P -> 0P0
    5. P -> 1P1
    """)
    print('\nBIENVENIDO A PALINDROMES\n%sMENU%s' % (separador, separador))
    print("""
        1 -> Ingresar datos (manual)
        2 -> Modo automatico
        4 -> Salir
    """)
    try:
        opcion = int(input("Selecciona una opcion valida: "))
        return opcion
```

```
    except Exception as e:
        print('Error ', e)
        return 0
def entrada_automatico():
    longitud = random.randint(0, 1000)
    print("Generando palindromo con longitud = %s" % longitud)
    palindromo(longitud)

def entrada_consola():
    longitud = int(input('Introduce un numero entre 0 y 1000: '))
    if longitud>1000 or longitud<0:
        print('Algo salio mal =(')
        return 0
    print("Generando palindromo con longitud = %s" % longitud)
    palindromo(longitud)

main()
```

0.2 PALINDROMO (CODIGO)

```
# -*- coding: utf-8 -*-
from __future__ import print_function
import random
def palindromo(repeticiones):
    archivo = open('palindromo-historia.txt', 'w')
    cadena = 'P'
    base_random = ''
    archivo.write('Longitud = %s\n' %repeticiones)
    archivo.write(cadena + '\n')
    print(cadena)
    if repeticiones % 2 == 1:
        cadena = generar_cadena(cadena, (repeticiones-1)/2, archivo)
        base_random = random.choice(['0', '1'])
    else:
        cadena = generar_cadena(cadena, repeticiones/2, archivo)
    cadena = cadena.replace('P', base_random)
    if base_random == '':
        base_random = 'e'
    archivo.write('Cadena final con base P=%s -> %s\n' %(base_random, cadena))
    print('Cadena final con base P=%s -> %s' %(base_random, cadena))
    archivo.close()

def generar_cadena(cadena, repeticiones, archivo):
    if repeticiones > 0:
        regla = random.choice(['0', '1'])
```

```
        cadena = cadena.replace('P', regla+'P'+regla)
        print('%s    Regla usada: %sP%s ' %(cadena, regla, regla))
        archivo.write('%s    Regla usada: %sP%s \n' %(cadena, regla, regla))
        repeticiones = repeticiones - 1
        cadena = generar_cadena(cadena, repeticiones, archivo)
    return cadena
```

0.3 FUNCIONAMIENTO Y PRUEBAS

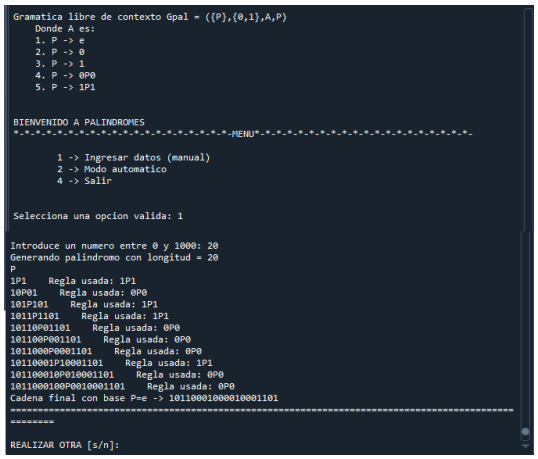

Figure 1: Menú Manual

Figure 2: Historia del automata

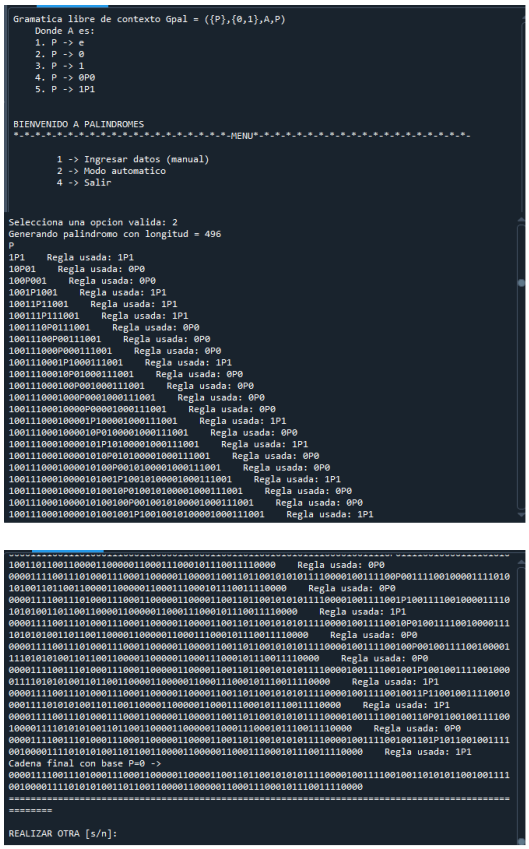

Figure 3: Menu autómatico, sin embargo se genera una cadena larga y no entra en la imagen, se recorto,pero se hace de forma correcta todo el proceso

1 CONCLUSION

Para esta práctica siguiendo la gramática libre de contexto Gpal = (P,0,1,A,P), para cadenas donde pueden ser tan grandes como dicta el problema siguiendo las reglas aplicadas a los números binarios dados, generados en la cadena tanto de forma manual o automático, de manera que se van aplicando las reglas con las funciones que le fuimos proporcionando. No hubo algún problema para ir dando las funciones a utilizar y dar la regla correspondiente y la cerradura, lo único que podría mejorar de esta al igual que la pasada, aplicarla con otros caracteres y ver como funciona.

References

[1] J. E. Hopcroft, R. Motwani, and J. D. Ullman, Introduccion a La Teorıa De Automatas, Lenguajes Y Computacion. Addison-Wesley, 2007.

[2] J. D. Ullman, "Finite Automata." http://infolab.stanford.edu/ ullman/ialc/spr10/slides/fa1.pdf, 2010. [Consultado: 2021-12-20].