Instituto Politécnico Nacional

ESCUELA SUPERIROR DE COMPUTACIÓN

PILAS

Práctica 6

Alumno:
Alcántara Covarrubias Erik
Profesor:
Juarez Martinez Genaro
Grupo: 4CM6

1 Introducción

Fundamentalmente, el autómata a pila es un autómata finito no determinista con transiciones- ϵ y una capacidad adicional: una pila en la que se puede almacenar una cadena de "símbolos de pila". La presencia de una pila significa que, a diferencia del autómata finito, el autómata a pila puede "recordar" una cantidad infinita de información.

Sin embargo, a diferencia de las computadoras de propósito general, que también tienen la capacidad de recordar una cantidad arbitrariamente grande de información, el autómata a pila solo puede acceder a la información disponible en su pila de acuerdo con la forma de manipular una pila FIFO (first-in-first-out way, primero en entrar primero en salir).

Así, existen lenguajes que podrían ser reconocidos por determinados programas informáticos, pero no por cualquier autómata a pila. De hecho, los autómatas a pila reconocen todos los lenguajes independientes del contexto y solo estos.

2 Marco teórico

Los autómatas de pila, en forma similar a como se usan los autómatas finitos, también se pueden utilizar para aceptar cadenas de un lenguaje definido sobre un alfabeto A.
Los autómatas de pila pueden aceptar lenguajes que no pueden aceptar los autómatas finitos. Un autómata de pila cuenta con una cinta de entrada y un mecanismo de control que puede encontrarse en uno de entre un número finito de estados. Uno de estos estados se designa como estado inicial, y además algunos estados se llaman de aceptación o finales.
A diferencia de los autómatas finitos, los autómatas de pila cuentan con una memoria auxiliar llamada pila. Los símbolos (llamados símbolos de pila) pueden ser insertados o extraídos de la pila, de acuerdo con el manejo $last - in - first - out (LIFO)$.

Las transiciones entre los estados que ejecutan los autómatas de pila dependen de los símbolos de entrada y de los símbolos de la pila. El autómata acepta una cadena x si la secuencia de transiciones, comenzando en estado inicial y con pila vacía, conduce a un estado final, después de leer toda la cadena x.

La notación formal de un autómata a pila incluye siete componentes. Escribimos la especificación de un autómata a pila P de la forma siguiente:

$$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$$

El significado de cada uno de los componentes es el siguiente:

- Q: Un conjunto finito de estados, como los estados de un autómata finito.
- Σ: Un conjunto finito de símbolos de entrada, también análogo al componente correspondiente de un autómata finito.
- Γ: Un alfabeto de pila finito. Este componente, que no tiene análogo en los autómatas finitos, es el conjunto de símbolos que pueden introducirse en la pila.
- δ: La función de transición. Como en el autómata finito, δ controla el comportamiento del autómata. Formalmente, δ toma como argumento $\delta(q, a, X)$, donde:

  1. q es un estado de Q.
  2. a es cualquier símbolo de entrada de Σ o $a = \epsilon$, la cadena vacía, que se supone que no es un símbolo de entrada.
  3. X es un símbolo de la pila, es decir, pertenece a Γ.

  La salida de δ es un conjunto finito de pares (p, γ), donde p es el nuevo estado y γ es la cadena de símbolos de la pila que reemplaza X en la parte superior de la pila.

- q_0: El estado inicial. El autómata a pila se encuentra en este estado antes de realizar ninguna transición.

- Z_0: El símbolo inicial. Inicialmente, la pila dels autómata a pila consta de una instancia de este símbolo y de nada más.
- F: El conjunto de estados de aceptación o estados finales.

3 Explicación del problema e implementación del algoritmo

3.1 Problema a resolver

Programar un autómata de pila que sirva para reconocer el lenguaje libre de contexto $0^n1^n|n >= 1$.

Adicionalmente, el programa debe de contar con las siguientes características:

1. La cadena puede ser ingresada por el usuario o automáticamente. Si es aleatoriamente, la cadena no podrá ser mayor a 100,000 caracteres.
2. Mandar a un archivo y en pantalla la evaluación del autómata a través de descripciones instantáneas (IDs).
3. Animar el autómata de pila, solo si la cadena es menor igual a 10 caracteres.

3.2 Implementación

```
    import random
import time
import os
from time import sleep

# It's a stack
class Pila(object):
    def __init__(self):
        self.largo = -1
        self.espacios = []

    def final(self):
        """
        If the length of the list is -1, then the list is empty
        :return: The final method returns a boolean value.
        """
        if self.largo == -1:
            return True
        else:
            return False

    def extraer(self):
        """
        It returns the last element of the list, and then removes it from the list
        :return: The value of the last element in the list.
        """
        if self.final():
            return 'e'
        else:
            valor = self.espacios[self.largo]
            self.largo -= 1
            return valor

    def insertar(self, elemento):
        """
```

```
        It inserts an element into the array.

        :param elemento: The element to be inserted
        """
        self.largo += 1
        self.espacios[self.largo:] = [elemento]

    def revelar(self):
        """
        It takes the length of the string, and then iterates through the string, adding
        a new string, and then returns the new string
        :return: The string of the spaces in the list.
        """
        i = self.largo
        cadena = ''
        while(i>-1):
            cadena += self.espacios[i]
            i -= 1
        return cadena

def DES(cadena, decision):
    """
    It takes a string and a boolean as arguments, and if the boolean is true, it will pr
    in a way that makes it look like a stack is being used to process the string

    :param cadena: The string to be tested
    :param decision: True or False, if True, the program will show the animation, if Fals
    the steps of the program
    """
    pila = Pila()
    archivo = open('Practica6/HISPILA.txt', 'w')
    pila.insertar('Zo')
    estado = 'q'
    auxiliar = cadena
    cadena = cadena + ' '
    archivo.write('La cadena es: ' + auxiliar + '\n')
    for simbolo in cadena:
        if auxiliar == '':
            auxiliar = 'e'
        if decision:
            time.sleep(1)
            ANIMACION(estado, auxiliar, pila)
        else:
            print('(%s, %s, %s)' %(estado, auxiliar, pila.revelar()), end='')
        archivo.write('(%s, %s, %s)' %(estado, auxiliar, pila.revelar()))
        if estado == 'q':
            if simbolo == '0':
                pila.insertar('X')
            elif simbolo == '1':
                if pila.extraer() == 'Zo':
                    pila.insertar('Zo')
                    break
                estado = 'p'
            else:
                estado='q'
                break
```

```
        elif estado == 'p':
            if simbolo == '1':
                if pila.extraer() == 'Zo':
                    estado = 'f'
                    pila.insertar('Zo')
                    break
            elif simbolo == '0':
                pila.insertar('X')
                auxiliar = auxiliar[1:]
                break
            elif simbolo == ' ':
                estado = 'f'
            else:
                break
        auxiliar = auxiliar[1:]
        archivo.write('->')

    if auxiliar == '':
        auxiliar = 'e'
    if (pila.revelar() == 'Zo') and auxiliar == 'e' and estado=='f':
        if decision:
            time.sleep(1)
            ANIMACION(estado, auxiliar, pila)
        else:
            print('(%s, %s, %s)' %(estado, auxiliar, pila.revelar()))
            print('\n')
        archivo.write('(%s, %s, %s)' %(estado, auxiliar, pila.revelar()))
        print('Esta cadena es valida')
        archivo.write('\nEsta cadena es valida')
    else:
        print('Esta cadena no es valida')
        archivo.write('\nEsta cadena no es valida')
    archivo.close()

def ANIMACION(estado, cadena_aux, stack):
    """
    It clears the screen, prints the current state, the current string, and the current
    waits for 0.9 seconds

    :param estado: The current state of the automaton
    :param cadena_aux: The string that is being processed
    :param stack: is the stack that is used in the program
    """

    pila = 'Zo'
    if stack.revelar() != '':
        pila = stack.revelar()
    if os.name =="nt":
        os.system("cls")
    else:
        os.system("clear")

    print("\n")
    print(cadena_aux + " -> " +estado+" -> " + pila)
    sleep(0.9)
```

```
if __name__ == "__main__":

    while True:

        print("\n*****PROGRAMA 6: PILA*****")
        print("1.-Colocar la cadena")
        print("2.-Cadena aleatoria")
        print("3.-Salir")
        OP = int(input("Elija una opcion: "))

        if OP == 1:
            cadena = input("Escribe el numero binario: ")
        elif OP ==2:
            i = 0
            LONGITUD = random.randint(1, 100000)
            cadena = ''
            while i < LONGITUD:
                cadena += random.choice(['0', '1'])
                i += 1
        elif OP ==3:
            print("Adios!!\n")
            exit()
        else:
            print("No existe esa opcion\n")
            break

        print("El numero es: ", cadena)
        largo = len(cadena)
        if largo <= 10:
            animacion = True
        else:
            print("No se puede realizar la animacion\n")
            animacion = False

        DES(cadena, animacion)
```

3.3 Capturas de Resultados

4 Conclusión

5 Bibliografía

- Introduction to Automata Theory, Languages, and Computation John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman Pearson/Addison Wesley, 2nd edition
- Software Foundation, P. (2022, 13 enero). 3.10.2 Documentation. Documentación de Python. Recuperado 15 de febrero de 2022, de https://docs.python.org/es/3/