

Máquina de Turing que duplica cadena de unos

Alumno: Monroy Martos Elioth

Profesor: Genaro Juárez Martínez

Materia: Computing Selected Topics

Grupo: 3CM8

19 de septiembre de 2018

Índice

1. Introducción	3
2. Desarrollo	4
3. Resultados	8
Referencias	10

1. Introducción

Las máquinas de Turing son un dispositivo que permite idealmente resolver cualquier problema, debido a que tiene una memoria infinita, sin embargo esta implementación no es posible de realizar y deben ser acotadas a una memoria o cinta finita, de tal forma que se dice que si una máquina de Turing es capaz de resolver un problema, este problema es computable (quiere decir que una computadora lo puede resolver). En caso contrario, se dice que el problema no es computable.[1]

El ejercicio a realizar en esta actividad, es diseñar e implementar una máquina de Turing que duplique la cantidad de unos de una cadena ingresada, y que además, se muestre el procedimiento en una animación y se escriba en un archivo el log de los pasos realizados por la máquina de Turing.

2. Desarrollo

Para la elaboración del programa, se realizó el siguiente diagrama que modela los estados y las transiciones que tiene la máquina.

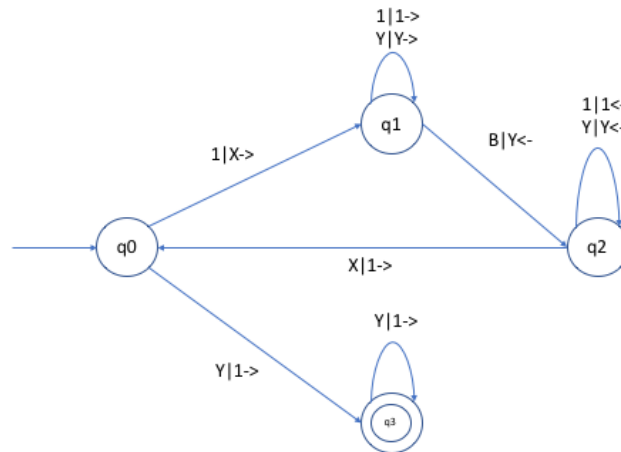


Figura 1: Estados y transiciones usados

Posteriormente, para el modelado de la máquina de Turing se realizó la siguiente implementación en Python 3.7, la cual es una clase que sirve para trabajar con la misma:

maquina.py

```
1 class MaquinaTuring(object):
2     def __init__(self, estados, alfabeto, transiciones, inicial,
3         final, cadena):
4         self.estados=estados
5         self.alfabeto=alfabeto
6         self.transiciones=transiciones
7         self.inicial=inicial
8         self.estado_actual=self.inicial
9         self.final=final
10        self.apuntador=0
11        self.blanco="B"
12        self.cadena=list(cadena)
```

```

12     self.direccion=""
13     def evaluar(self):
14         if len(self.cadena)-1<self.apuntador:
15             caracter=self.blanco
16         else:
17             caracter=self.cadena[self.apuntador]
18
19         if caracter in self.alfabeto:
20             transicion=(self.estado_actual,caracter)
21             if transicion in self.transiciones:
22                 siguiente=self.transiciones[transicion]
23                 if len(self.cadena)-1<self.apuntador:
24                     self.cadena.append(self.blanco)
25                 if self.apuntador<0:
26                     self.cadena.insert(0,self.blanco)
27                 self.cadena[self.apuntador]=siguiente[1]
28                 if siguiente[2]=="R":
29                     self.apuntador+=1
30             else:
31                 self.apuntador=self.apuntador-1
32
33             self.estado_actual=siguiente[0]
34             self.direccion=siguiente[2]
35             if self.estado_actual in self.estados:
36                 return True
37             else:
38                 return False
39         else:
40             return False
41     else:
42         return False
43     def esFinal(self):
44         if self.estado_actual in self.final:
45             if len(self.cadena)-1<self.apuntador:
46                 return True
47         return False

```

Por si sola está máquina no tiene funcionalidad, por lo cual fue necesario el uso de la misma en otro archivo, el cual se encarga de manejar todo el funcionamiento del programa, como recibir la entrada (cadena de unos), y utilizar tkinter para realizar la animación de la máquina de Turing.

```

1 from tkinter import *
2 from tkinter import font as tkfont
3 import time
4 from maquina import MaquinaTuring

```

```

5
6 entrada = input("Ingrese la cadena a duplicar: ")
7 estados=["q0","q1","q2","q3"]
8 alfabeto=["1","Y","X","B"]
9 transiciones={
10     ("q0", "1"): ("q1", "X", "R"),
11     ("q0", "Y"): ("q3", "1", "R"),
12     ("q1", "1"): ("q1", "1", "R"),
13     ("q1", "Y"): ("q1", "Y", "R"),
14     ("q1", "B"): ("q2", "Y", "L"),
15     ("q2", "1"): ("q2", "1", "L"),
16     ("q2", "X"): ("q0", "1", "R"),
17     ("q2", "Y"): ("q2", "Y", "L"),
18     ("q3", "Y"): ("q3", "1", "R"),
19 }
20
21 maquina = MaquinaTuring(estados, alfabeto, transiciones, estados
22     [0], estados[3], entrada)
23
24 master = Tk()
25 master.title("Turing")
26 c = Canvas(master, width=400, height=300)
27 c.pack()
28 tipo_letra = tkfont.Font(family="Helvetica", size=20)
29 flecha = c.create_line(175, 150, 175, 175, arrow=LAST, width=2)
30 texto = c.create_text(165, 200, text="".join(maquina.cadena),
31     font=tipo_letra, anchor=W)
32 estado = c.create_text(120, 150, text=maquina.estado_actual,
33     font=tipo_letra, anchor=W)
34
35 archivo = open("salida.txt", "w")
36 while not maquina.esFinal():
37     print("Cadena: {}".join(maquina.cadena))
38     print("Estado actual: "+str(maquina.estado_actual)+",
39         apuntador: "+str(maquina.apuntador+1)+"\n")
40
41     archivo.write("Cadena: {}".join(maquina.cadena)+"\n")
42     archivo.write("Estado actual: "+str(maquina.estado_actual)+",
43         apuntador: "+str(maquina.apuntador+1)+"\n")
44     if not maquina.evaluar():
45         print("_____")
46         archivo.write("_____")
47         archivo.write("\n")
48     print("Siguiente estado: "+str(maquina.estado_actual))

```

```

45 print("_____")
46
47 archivo.write("Siguiente estado: "+str(maquina.estado_actual))
48 archivo.write("_____")
49 archivo.write("\n")
50
51 master.update()
52 time.sleep(.8)
53 c.itemconfigure(texto, text="".join(maquina.cadena))
54 c.itemconfigure(estado, text=str(maquina.estado_actual))
55 if maquina.direccion == 'R':
56     c.move(flecha, 9, 0)
57 else:
58     c.move(flecha, -9, 0)
59
60 print("Resultado: "+"".join(maquina.cadena))
61 archivo.write("Resultado: "+"".join(maquina.cadena)+"\n")
62 master.mainloop()

```

3. Resultados

Para comprobar la funcionalidad del programa, se realizó la siguiente prueba, se ingresó la cadena “11”. Los resultados obtenidos fueron los siguientes:

```
MacBook-Pro-de-Elioth:MT eliothmonroy$ python3 main.py
Ingrese la cadena a duplicar: 11
Cadena: 11
Estado actual: q0, apuntador: 1

Siguiente estado: q1
-----
Cadena: X1
Estado actual: q1, apuntador: 2

Siguiente estado: q1
-----
Cadena: X1
Estado actual: q1, apuntador: 3

Siguiente estado: q2
-----
Cadena: X1Y
Estado actual: q2, apuntador: 2

Siguiente estado: q2
-----
Cadena: X1Y
```

Figura 2: Resultados prueba (1)


```

Cadena: 11
Estado actual: q0, apuntador: 1
Siguiete estado: q1-----
Cadena: X1
Estado actual: q1, apuntador: 2
Siguiete estado: q1-----
Cadena: X1
Estado actual: q1, apuntador: 3
Siguiete estado: q2-----
Cadena: X1Y
Estado actual: q2, apuntador: 2
Siguiete estado: q2-----
Cadena: X1Y
Estado actual: q2, apuntador: 1
Siguiete estado: q0-----
Cadena: 11Y
Estado actual: q0, apuntador: 2
Siguiete estado: q1-----
Cadena: 1XY
Estado actual: q1, apuntador: 3
Siguiete estado: q1-----
Cadena: 1XY
Estado actual: q1, apuntador: 4
Siguiete estado: q2-----
Cadena: 1XY
Estado actual: q2, apuntador: 3
Siguiete estado: q2-----
Cadena: 1XY
Estado actual: q2, apuntador: 2
Siguiete estado: q0-----
Cadena: 11YY
Estado actual: q0, apuntador: 3
Siguiete estado: q3-----
Cadena: 111Y
Estado actual: q3, apuntador: 4
Siguiete estado: q3-----
Resultado: 1111

```

Figura 3: Resultados prueba (2)

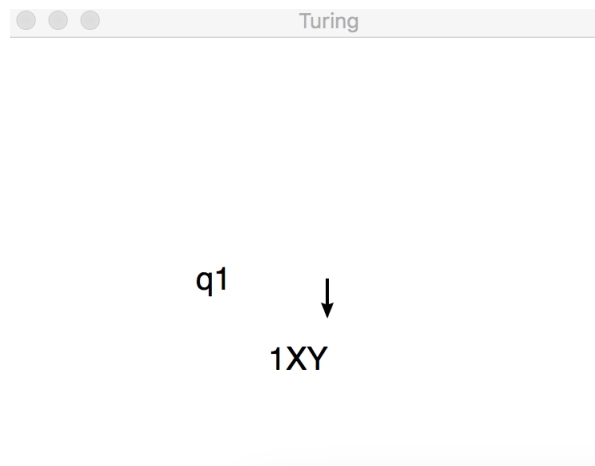


Figura 4: Resultados prueba (3)

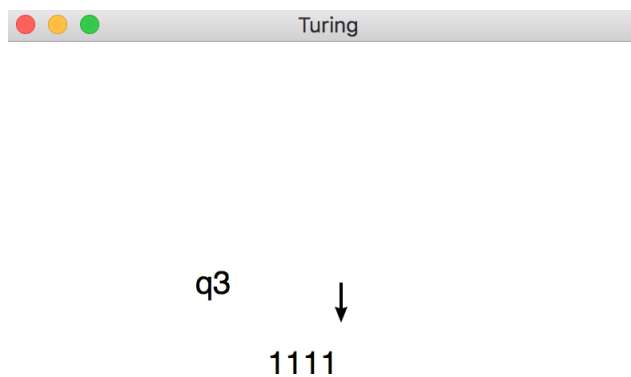


Figura 5: Resultados prueba (4)

Referencias

- [1] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introducción a La Teoría De Autómatas, Lenguajes Y Computación*. Addison-Wesley, 2007.