Práctica 9. "Maquina de Turing"

Carmona Serrano Ian Carlo
Ingeniería en Inteligencia Artificial 5BM1
Teoría de la Computación , ESCOM- IPN

6 de Enero 2024

1 Introducción

La Máquina de Turing, es un dispositivo matemático fundamental en la teoría de la computación. En esta práctica, se busca desarrollar un programa que implemente una Máquina de Turing capaz de reconocer el lenguaje $\{0^n1^n | n >= 1\}$. Este lenguaje describe secuencias donde una cantidad igual de ceros y unos está presente, organizados de manera simétrica. La tabla de transiciones provista por John Hopcroft para este ejercicio servirá como guía para la computación de la máquina. El programa permitirá ingresar cadenas definidas por el usuario o generarlas automáticamente, con una longitud máxima de 1000 caracteres. La salida será dirigida a un archivo de texto, detallando las descripciones instantáneas en cada paso de la computación.

2 Marco Teórico

2.1 Máquinas de Turing

Las Máquinas de Turing son dispositivos teóricos, propuestos por Alan Turing en 1936, que modelan un sistema de cómputo con capacidad ilimitada de almacenamiento y manipulación de datos. Consisten en una cinta infinita dividida en celdas, una cabeza de lectura/escritura y un conjunto de estados y reglas de transición que gobiernan su comportamiento. Las Máquinas de Turing pueden simular cualquier algoritmo computable y son fundamentales en el estudio de la computabilidad y la complejidad.

2.2 Lenguaje 0^n1^n

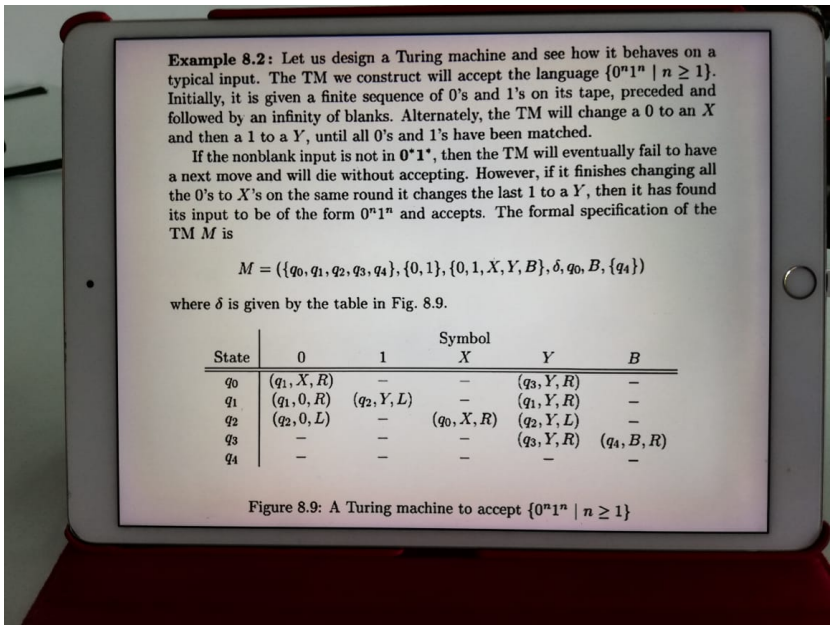
El lenguaje 0^n1^n representa un conjunto de cadenas donde una cantidad igual de ceros y unos están presentes, apareciendo en orden consecutivo y simétrico. Para cada n mayor o igual a 1, la cadena consta de n ceros seguidos por n unos. Es un ejemplo clásico de un lenguaje que no es regular pero que puede ser reconocido por una Máquina de Turing, demostrando así su poder computacional.

3 Desarrollo

En esta práctica, se implementará una Máquina de Turing para reconocer el lenguaje 0^n1^n. Se describirá la estructura y funcionamiento de la máquina, así como su implementación en código.

3.1 Implementación de la Máquina de Turing

La implementación de la Máquina de Turing se llevará a cabo en un lenguaje de programación específico, utilizando las reglas de transición proporcionadas en el enunciado. Se describirá detalladamente el diseño del programa, incluyendo la estructura de datos utilizada, el proceso de entrada de cadenas y la lógica de la máquina.



Example 8.2: Let us design a Turing machine and see how it behaves on a typical input. The TM we construct will accept the language $\{0^n1^n \mid n \geq 1\}$. Initially, it is given a finite sequence of 0's and 1's on its tape, preceded and followed by an infinity of blanks. Alternately, the TM will change a 0 to an X and then a 1 to a Y, until all 0's and 1's have been matched.

If the nonblank input is not in $\mathbf{0^*1^*}$, then the TM will eventually fail to have a next move and will die without accepting. However, if it finishes changing all the 0's to X's on the same round it changes the last 1 to a Y, then it has found its input to be of the form 0^n1^n and accepts. The formal specification of the TM M is

$$M = (\{q_0, q_1, q_2, q_3, q_4\}, \{0, 1\}, \{0, 1, X, Y, B\}, \delta, q_0, B, \{q_4\})$$

where δ is given by the table in Fig. 8.9.

State	Symbol 0	1	X	Y	B
q_0	(q_1, X, R)	–	–	(q_3, Y, R)	–
q_1	$(q_1, 0, R)$	(q_2, Y, L)	–	(q_1, Y, R)	–
q_2	$(q_2, 0, L)$	–	(q_0, X, R)	(q_2, Y, L)	–
q_3	–	–	–	(q_3, Y, R)	(q_4, B, R)
q_4	–	–	–	–	–

Figure 8.9: A Turing machine to accept $\{0^n1^n \mid n \geq 1\}$

Figure 1: Tabla

3.2 Descripciones instantáneas y Salida del Programa

El programa generará descripciones instantáneas en cada paso de la computación. Estas descripciones se guardarán en un archivo de texto para su posterior revisión y análisis. Se explicará el formato de las descripciones instantáneas y cómo estas representan el estado de la máquina en cada paso.

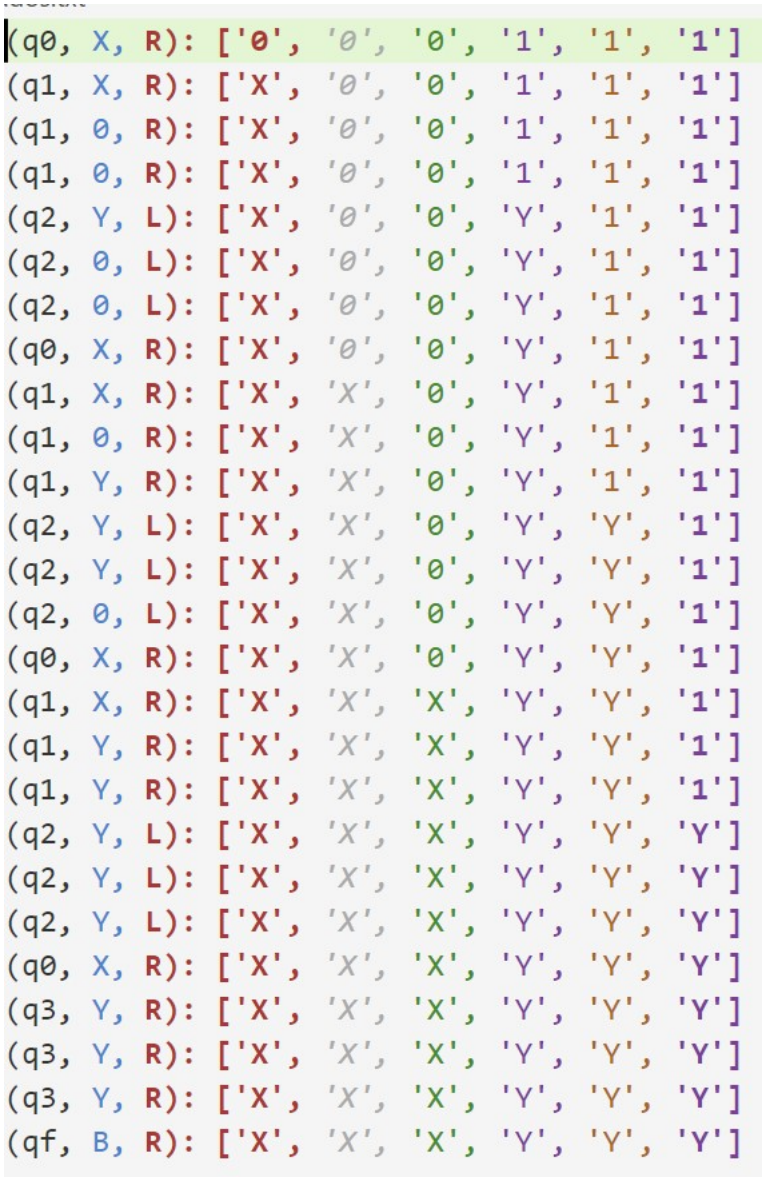


Figure 2: Txt

3.3 Animación de la Máquina de Turing

Para cadenas menores o iguales a 16 caracteres, se implementará una animación que muestre el funcionamiento de la Máquina de Turing. Esta animación servirá para visualizar el proceso de computación y comprender mejor el funcionamiento de la máquina.

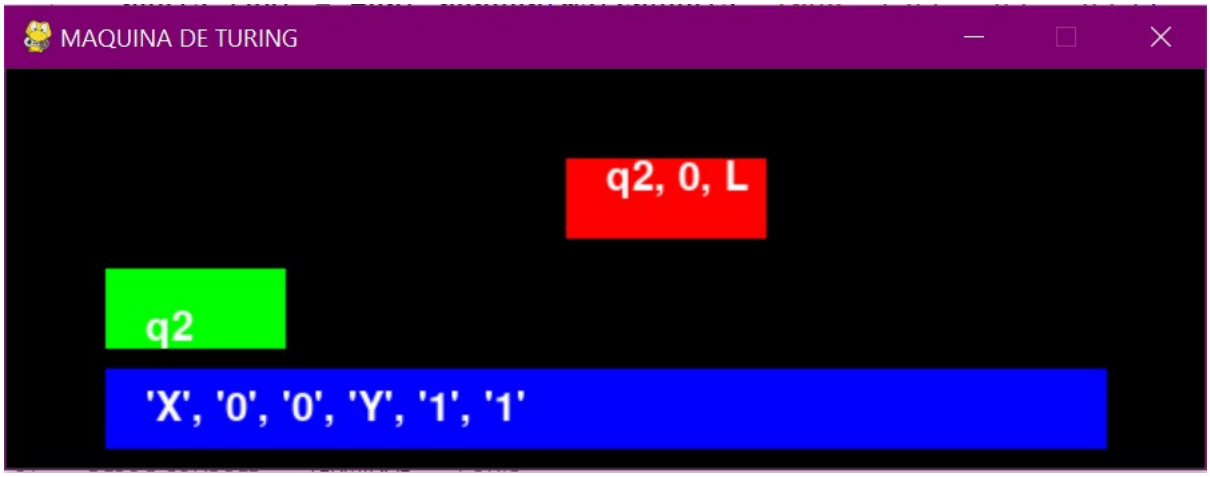


Figure 3: Animacion

Código

```
import random
import pygame
import re
import sys

def graficarMaquina():
    pygame.init()

    width, height = 600, 200
    screen = pygame.display.set_mode((width, height))
    pygame.display.set_caption("MAQUINA DE TURING")

    font = pygame.font.Font(None, 30)

    f = open('./resultados.txt', 'r')
    text = f.read()

    matchesCinta = re.findall(r'\[(.*?)\]', text)
    matchesRegla = re.findall(r'\((.*?)\)', text)
    matchesEstado = re.findall(r'q[0-9]+', text)

    running = True
    i = 0

    while running:

        screen.fill((0, 0, 0))
```

```
28
29          if i < len(matchesCinta):
30              matchCinta = matchesCinta[i]
31              matchEstado = matchesEstado[i]
32              matchRegla = matchesRegla[i]
33
34              pygame.draw.rect(screen, (0, 0, 255), (50, 150,
    500, 40))
35              pygame.draw.rect(screen, (0, 255, 0), (50, 100,
    90, 40))
36              pygame.draw.rect(screen, (255, 0, 0), (280, 45,
    100, 40))
37
38              estado_text = font.render(matchEstado, True,
    (255, 255, 255))
39              cinta_text = font.render(matchCinta, True, (255,
     255, 255))
40              regla_text = font.render(matchRegla, True, (255,
     255, 255))
41
42              screen.blit(estado_text, (70, 120))
43              screen.blit(cinta_text, (70, 160))
44              screen.blit(regla_text, (300, 45))
45
46              i += 1
47              pygame.display.flip()
48              pygame.time.wait(500)
49
50          for event in pygame.event.get():
51              if event.type == pygame.QUIT:
52                  running = False
53
54      pygame.quit()
55      sys.exit()
56
57  def maquina_turing(cinta):
58
59      f = open('resultados.txt', 'w')
60
61      cinta = list(cinta)
62
63      estado = 'q0'
64      index = 0
65
66      f.write(f"({str(estado)}, X, R): {str(cinta)}\n")
67
68      while estado != "qf":
69          if estado == 'q0':
70              if index >= len(cinta):
71                  return False
```

```
72
73              if cinta[index] == '0':
74                  cinta[index] = 'X'
75                  index += 1
76                  estado = 'q1'
77
78                  f.write(f"({str(estado)}, X, R): {str(cinta)
    }\n")
79
80              elif cinta[index] == 'Y':
81                  cinta[index] = 'Y'
82                  index += 1
83                  estado = 'q3'
84
85                  f.write(f"({str(estado)}, Y, R): {str(cinta)
    }\n")
86
87              else:
88                  return False
89          elif estado == 'q1':
90
91              if index >= len(cinta):
92                  return False
93
94              if cinta[index] == '0':
95                  cinta[index] = '0'
96                  index += 1
97
98                  f.write(f"({str(estado)}, 0, R): {str(cinta)
    }\n")
99
100             elif cinta[index] == '1':
101                 cinta[index] = 'Y'
102                 index -= 1
103                 estado = 'q2'
104
105                 f.write(f"({str(estado)}, Y, L): {str(cinta)
    }\n")
106
107             elif cinta[index] == 'Y':
108                 cinta[index] = 'Y'
109                 index += 1
110
111                 f.write(f"({str(estado)}, Y, R): {str(cinta)
    }\n")
112             else:
113                 return False
114         elif estado == 'q2':
115
116             if index >= len(cinta):
```

```
117                 return False
118
119             if cinta[index] == '0':
120                 cinta[index] = '0'
121                 index -= 1
122
123                 f.write(f"({str(estado)}, 0, L): {str(cinta)
    }\n")
124             elif cinta[index] == 'X':
125                 cinta[index] = 'X'
126                 index += 1
127                 estado = 'q0'
128
129                 f.write(f"({str(estado)}, X, R): {str(cinta)
    }\n")
130
131             elif cinta[index] == 'Y':
132                 cinta[index] = 'Y'
133                 index -= 1
134
135                 f.write(f"({str(estado)}, Y, L): {str(cinta)
    }\n")
136             else:
137                 return False
138         elif estado == 'q3':
139             if index >= len(cinta):
140                 estado = 'qf'
141
142                 f.write(f"({str(estado)}, B, R): {str(cinta)
    }\n")
143
144             elif cinta[index] == 'Y':
145                 cinta[index] = 'Y'
146                 index += 1
147
148                 f.write(f"({str(estado)}, Y, R): {str(cinta)
    }\n")
149             else:
150                 return False
151     return True
152
153 print("Elija el modo:")
154 print("1. Automatico")
155 print("2. Manual")
156 opc = input("Inserte su opcion deseada: ")
157 entrada = ""
158
159 if opc == '1':
160     cantidadCeros = random.randint(1, 500)
161     cantidadUnos = random.randint(1, 500)
```

```
    print("Cantidad ceros: ", cantidadCeros)
    print("Cantidad unos: ", cantidadUnos)

    for i in range(cantidadCeros):
        entrada += '0'
    for i in range(cantidadUnos):
        entrada += '1'
else:
    entrada = input("Inserte su cadena: ")


print("Su entrada fue: ", entrada)

esValida = maquina_turing(entrada)

if esValida:
    print("Cadena aceptada")
else:
    print("No es valida")

if len(entrada) <= 16:
    graficarMaquina()
```

Listing 1: Practica 9

4 Conclusión

La implementación exitosa de la Máquina de Turing para reconocer el lenguaje 0^n1^n ha demostrado su capacidad para procesar y validar cadenas según las reglas de transición especificadas. La generación de descripciones instantáneas y la animación para cadenas más cortas han facilitado la comprensión del funcionamiento interno de la máquina. Este ejercicio ha sido fundamental para comprender la relevancia de las Máquinas de Turing en el procesamiento de lenguajes formales y su aplicación en la resolución de problemas computacionales específicos.

5 Bibliografía

Ullman, J.D. (2009-10). "CS154: Introduction to Automata and Complexity Theory". Sitio web: http://infolab.stanford.edu/ ullman/ialc/spr10/spr10.htmlLECTURE