INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

MATERIA: Teoría Computacional.

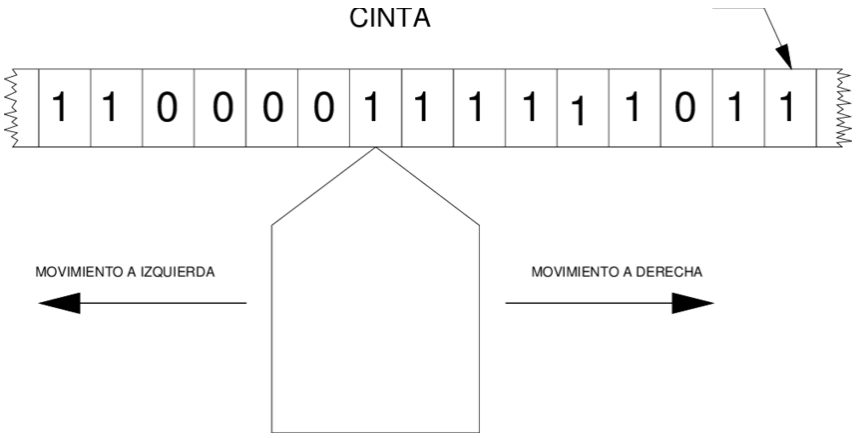
PRÁCTICA: Práctica 8. Máquina de Turing.

ALUMNO: Meza Vargas Brandon David.

PROFESOR: Jorge Luis Rosas Trigueros.

FECHA PRÁCTICA: 15-ene-2021

FECHA DE ENTREGA: 22-ene-2021



MARCO TEÓRICO

La máquina de Turing es un dispositivo creado en 1936, que representa un modelo idealizado de computación capaz de almacenar/procesar información virtualmente infinita. El sistema es una abstracción matemática que se construye de un modo extraordinariamente sencillo, pero que facilita la comprobación empirista de un abanico amplio de preguntas sobre las teorías de la computabilidad y/o de la complejidad. Su ideación marcó un gran hito en la historia de la informática, hasta el punto de ser considerada como el origen de los actuales ordenadores (y de las tecnologías afines, como las tabletas o los teléfonos móviles).

El artífice de esta fue Alan M. Turing, lógico y matemático inglés que pretendió toda su vida la concepción de un modelo teórico con el que dar respuesta a las incógnitas de su disciplina, de forma automática y accesible a todos.

Una máquina de Turing es una 7-tupla

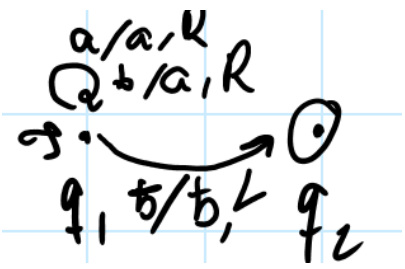
- M = (Q, Σ, Γ, s, Ƀ, F, δ)
- Q es un conjunto finito de estados
- Σ es un alfabeto de entrada
- Γ es un alfabeto llamado alfabeto de la cinta
- s ϵ Q es el estado inicial
- Ƀ ϵ Γ es el símbolo blanco (y no está en Σ)
- F ⊆ Q es el conjunto de estados finales o de aceptación
- δ: Q×Γ -> Q×Γ×(L,R) es una función parcial que se llama función de transición

Veamos un ejemplo de una máquina de Turing que sustituye las b's por a's a partir de donde empieza a procesar y hacia la derecha.
Termina en el último símbolo antes del primer Ƀ a la derecha.
:

Q={ q1, q2 }
Σ={ a, b }
Γ={ a, b, Ƀ }
F={ q2 }
s= q1

δ(q1,a)=(q1,a,R)
δ(q1,b)=(q1,a,R)
δ(q1,Ƀ)=(q2,Ƀ,L)

Su autómata
:



rocesar abba

Descripción instantánea
(qi,wσv)

(q1,abba)
⊢(q1,abba)
⊢(q1,aaba)
⊢(q1,aaaa)
⊢(q1,aaaaЬ)
⊢(q2,aaaa)
La máquina de Turing para.

La secuencia de todos los movimientos que conducen a una configuración de parada se llama
COMPUTACIÓN.

MATERIAL Y EQUIPO

Para la realización de esta práctica son necesarias las siguientes herramientas:

- Un equipo de cómputo que cumpla con los requerimientos para el uso del lenguaje de programación Python.
- Tener instalado el lenguaje de programación Python.
- Contar con un IDE para programar con Python, cualquiera es útil

DESARROLLO

El desarrollo de esta práctica consistió en programar en Python un par de máquinas de Turing, dos para ser exactos.

Primeramente, empezamos la práctica con un ejemplo sencillo, este ejemplo es la máquina de Turing que transforma las b's en a's, en la figura 1 se muestra la parte del código donde se ven las transiciones necesarias para que se funcione esta MT;

```
if __name__ == "__main__":
    # machine to convert a string of A's and B's to
    # all A's and accept
    m = TuringMachine("ABBABB", [1])

    m.addTransition(0,'A',0,'A','R')
    m.addTransition(0,'B',0,'A','R')
    m.addTransition(0,'_',1,'_','L')

    # run the TM
    m.execute()
```

Figura 1. Transiciones de la MT.

En la figura 2 vemos una cadena y el recorrido que hace para transformar las b´s en a´s;

```
ABBABB
^
ABBABB
 ^
AABABB
  ^
AAAABB
   ^
AAAABB
    ^
AAAAAB
     ^
AAAAAA_
      ^
AAAAAA_
     ^
Accept
>>>
```

Figura 2. Recorrido de la MT.

Otro ejemplo que el profesor nos proporcionó en esta práctica fue el de la máquina de Turing que acepta cadenas de longitud par, en la figura 3 vemos las transiciones que hacen funcionar a esta máquina;

```
if __name__ == "__main__":
    # Maquina que acepta cadenas de longitud par
    # all A's and accept
    m = TuringMachine("ABAB", [1])

    m.addTransition(0,'A',2,'C','R')
    m.addTransition(0,'B',2,'C','R')
    m.addTransition(0,'D',1,'D','L')
    m.addTransition(2,'A',2,'A','R')
    m.addTransition(2,'A',2,'A','R')
    m.addTransition(2,'B',2,'B','R')
    m.addTransition(2,'D',3,'D','L')
    m.addTransition(2,'_',3,'_','L')
    m.addTransition(3,'A',4,'D','L')
    m.addTransition(3,'B',4,'D','L')
    m.addTransition(4,'A',4,'A','L')
    m.addTransition(4,'B',4,'B','L')
    m.addTransition(4,'C',0,'C','R')


    # run the TM
    m.execute()
```

Figura 3. Transiciones de la MT.

En la figura 4 tenemos un ejemplo de una cadena que acepta y el recorrido que hace de acuerdo a las transiciones que vemos en la figura 3;

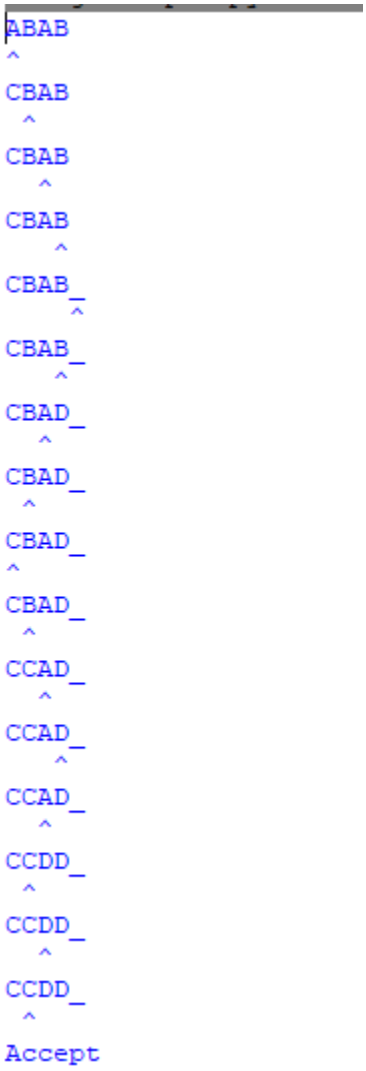


Figura 4. Recorrido de la MT.

Ahora sí, pasemos con los ejercicios a realizar

EJERCICIO 1

El primer ejercicio es hacer una máquina de Turing que acepte el siguiente lenguaje:

{w tiene el mismo número de a´s que de b´s}.

Para solución de este problema emplee lo que nos enseñó el profe el día de hoy, que es usar otros símbolos para ir checando que el número de a´s y b´s sea el mismo y llevar, por así decirlo, un control, la solución la podemos ver en la figura 5;

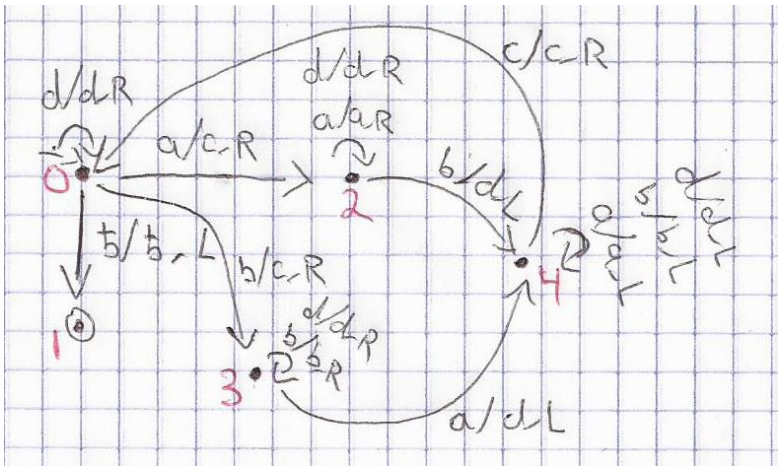


Figura 5. MT que acepta cadenas con el mismo número de a's que de b's.

Una vez hecho esto, pasamos a codificarlo en Python, tengo que decir que es algo tardado colocar las transiciones y más de una vez me paso que repetí una de estas, pero es necesario ya que codificando esto nos aseguramos que nuestra máquina de Turing este correcta, el código lo vemos en la figura 6;

```
if __name__ == "__main__":
    # machine to convert a string of A's and B's to
    # all A's and accept
    m = TuringMachine("BAAB", [1])

    m.addTransition(0,'A',2,'C','R')
    m.addTransition(0,'B',3,'C','R')
    m.addTransition(0,'D',0,'D','R')
    m.addTransition(0,'_',1,'_','L')
    m.addTransition(2,'A',2,'A','R')
    m.addTransition(2,'D',2,'D','R')
    m.addTransition(2,'B',4,'D','L')
    m.addTransition(3,'A',4,'D','L')
    m.addTransition(3,'B',3,'B','R')
    m.addTransition(3,'D',3,'D','R')
    m.addTransition(4,'A',4,'A','L')
    m.addTransition(4,'B',4,'B','L')
    m.addTransition(4,'D',4,'D','L')
    m.addTransition(4,'C',0,'C','R')


    # run the TM
    m.execute()
```

Figura 6. Transiciones de la MT.

En la figura 7 vemos la prueba con una cadena;

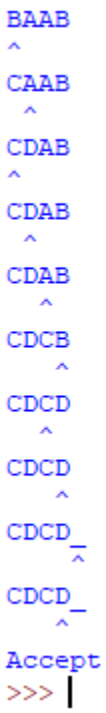


Figura 7. Prueba de nuestra MT.

En la figura 8 se prueba una cadena que no es aceptada, la cadena en cuestión es BAABB;

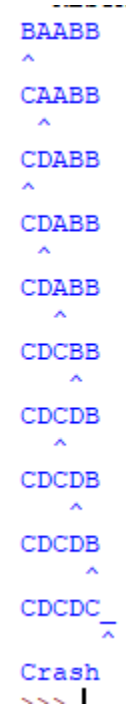


Figura 8. Prueba de nuestra MT

La palabra crash nos indica que la cadena no fue aceptada.

EJERCICIO 2

El segundo ejercicio es hacer una máquina de Turing que acepte el siguiente lenguaje:

$\{a^n b^n c^n\}$

Para la solución de este problema me apoye de la máquina de Turing que acepta el lenguaje $\{a^n b^n\}$, con esa misma máquina evalué una cadena del tipo $a^n b^n c^n$ y fui agregando los estados, así como las transiciones que vi que le hacían falta a la MT para formar la nueva que acepte el lenguaje solicitado en este ejercicio. De esta forma podemos ver la solución en la figura 9;

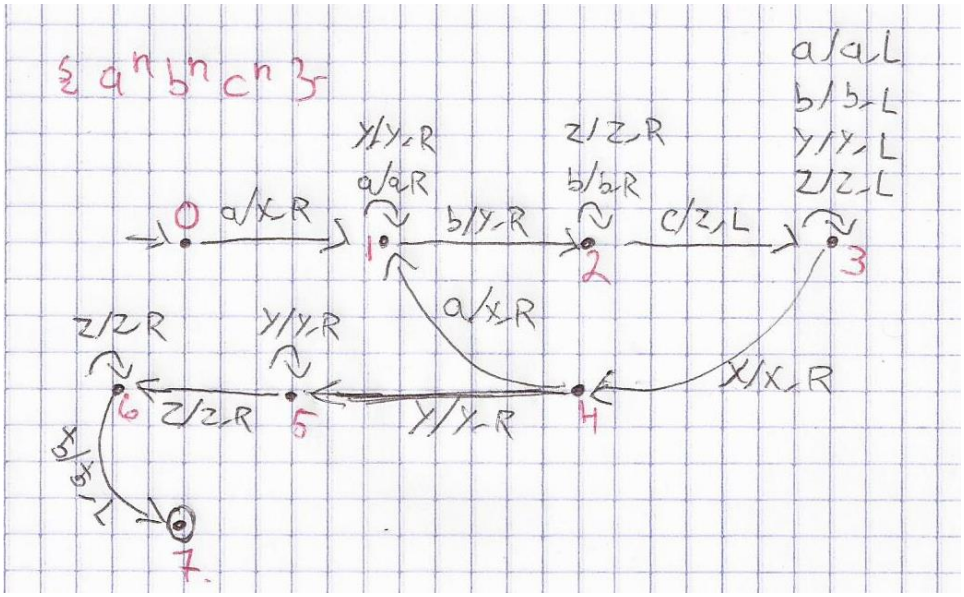


Figura 9. MT que acepta el lenguaje $a^n b^n c^n$

Una vez hecho esto, pasamos a codificarlo en Python colocando las transiciones correspondientes, esto lo vemos en la figura 10;

```
if __name__ == "__main__":
    # maquina que acepta el lenguaje a^nb^nc^n
    # all A's and accept
    m = TuringMachine("AAABBBCCC", [7])

    m.addTransition(0,'A',1,'X','R')
    m.addTransition(1,'A',1,'A','R')
    m.addTransition(1,'Y',1,'Y','R')
    m.addTransition(1,'B',2,'Y','R')
    m.addTransition(2,'B',2,'B','R')
    m.addTransition(2,'Z',2,'Z','R')
    m.addTransition(2,'C',3,'Z','L')
    m.addTransition(3,'A',3,'A','L')
    m.addTransition(3,'B',3,'B','L')
    m.addTransition(3,'Y',3,'Y','L')
    m.addTransition(3,'Z',3,'Z','L')
    m.addTransition(3,'X',4,'X','R')
    m.addTransition(4,'A',1,'X','R')
    m.addTransition(4,'Y',5,'Y','R')
    m.addTransition(5,'Y',5,'Y','R')
    m.addTransition(5,'Z',6,'Z','R')
    m.addTransition(6,'Z',6,'Z','R')
    m.addTransition(6,'_',7,'_','L')



    # run the TM
    m.execute()
```

Figura 10. Transiciones de la MT

En la figura 11 vemos la prueba con una cadena que acepta el lenguaje, la cual es AABBCC;

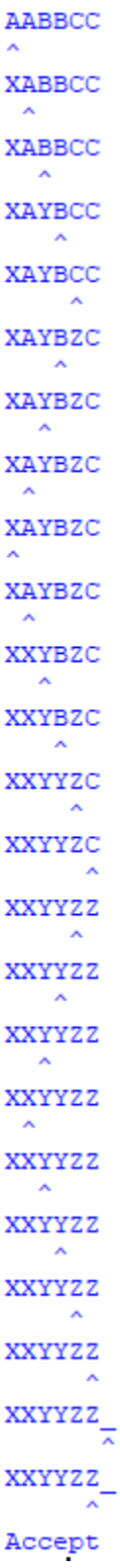


Figura 11. Prueba de la MT.

En la figura 12 podemos ver la prueba con una cadena que no es aceptada por la máquina de Turing, esta cadena es ABBC;

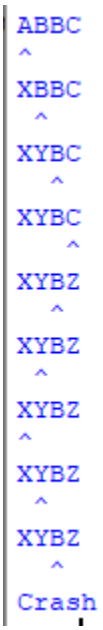


Figura 12. Prueba de la MT.

CONCLUSIONES Y RECOMENDACIONES

A partir de los ejercicios realizados en la práctica, pude comprender de una mejor manera lo revisado en las clases referente al tema de máquinas de Turing, pues, para ser honesto, al principio me costó algo entenderlas ya que se me hacían algo confusas, pero esta práctica me ayudo a entender mejor el tema y con los ejercicios realizados practiqué.

Puedo concluir que las máquinas de Turing son maquinas capaces de procesar información de una forma muy eficiente y eficaz, que, virtualmente es infinita y representan algo fundamental en la computación.

La práctica se realizó de una forma correcta, ya que el profesor respondió las dudas que se iban presentando de una manera clara.

BIBLIOGRAFÍA

- Qu, P., Yan, J., Zhang, Y. y Gao, G. (2017). Parallel Turing Machine, a Proposal. Journal of Computer Science and Technology, 32, 269-285.
- Apuntes de la clase de Teoría Computacional por el profesor Jorge Luis Rosas Trigueros.