ACTIVIDAD LABORATORIO NO.1
TRABAJO: MÁQUINAS DE TURING CON JFLAP

PRESENTADO POR:
ALEJANDRO DE MENDOZA

PRESENTADO AL PROFESOR:
ING ROGERIO ORLANDO BELTRAN CASTRO

FUNDACIÓN UNIVERSITARIA INTERNACIONAL DE LA RIOJA
BOGOTÁ D.C.

16 DE FEBRERO
2026

TABLA DE CONTENIDO

INTRODUCCIÓN

En el contexto de la teoría de la computación y los fundamentos de la informática, las Máquinas de Turing representan uno de los modelos más importantes para comprender el concepto de computabilidad y los límites de los sistemas algorítmicos. Estas máquinas constituyen un modelo abstracto capaz de simular cualquier proceso computacional, convirtiéndose en una herramienta esencial para el estudio formal de lenguajes, algoritmos y problemas decidibles o recursivamente enumerables.

El presente trabajo desarrolla un laboratorio práctico utilizando la herramienta JFLAP, un software educativo ampliamente empleado en el ámbito académico para la construcción y simulación de autómatas, gramáticas formales y Máquinas de Turing. Mediante esta herramienta, es posible visualizar paso a paso el comportamiento de las máquinas, verificar su funcionamiento y analizar el proceso de aceptación o rechazo de cadenas dentro de un lenguaje determinado.

En esta actividad se abordó el diseño e implementación de Máquinas de Turing que reconocen y procesan lenguajes recursivamente enumerables. En particular, se trabajó sobre dos ejercicios fundamentales: en primer lugar, la construcción de una máquina que acepta el lenguaje:

$$\mathrm{L} = \{0^{\mathrm{n}}\mathrm{n}\ 1^{\mathrm{n}} : \mathrm{n} > 0\}$$

El cual está compuesto por cadenas formadas por una secuencia de ceros seguida de una secuencia de unos, con la condición de que ambas cantidades sean exactamente iguales. Este lenguaje es un ejemplo clásico que no puede ser reconocido por autómatas finitos, pero sí por una Máquina de Turing mediante una estrategia de marcado y emparejamiento.

En segundo lugar, se diseñó una Máquina de Turing capaz de calcular el número consecutivo (sucesor) de un número binario dado como entrada. Este ejercicio permite evidenciar cómo las Máquinas de Turing pueden no solo reconocer lenguajes, sino también ejecutar procedimientos computacionales concretos, como la suma binaria con propagación de acarreo.

El objetivo principal de este laboratorio es comprender de manera práctica el funcionamiento interno de las Máquinas de Turing, desarrollando la capacidad de definir estados, transiciones y estrategias algorítmicas dentro de JFLAP. Para validar cada diseño, se realizaron múltiples simulaciones con cadenas aceptadas y rechazadas, permitiendo comprobar el correcto comportamiento de las máquinas construidas.

Este estudio demuestra la relevancia de los modelos formales en la informática teórica, así como la utilidad de herramientas de simulación como JFLAP para fortalecer la comprensión de conceptos fundamentales relacionados con lenguajes formales, computabilidad y procesos algorítmicos. En conclusión, el laboratorio constituye una aplicación práctica esencial para consolidar el vínculo entre la teoría matemática y la implementación computacional de máquinas abstractas.

DESARROLLO ACTIVIDAD

A continuación, se presenta el desarrollo paso a paso del laboratorio de Máquinas de Turing utilizando la herramienta JFLAP. El proceso fue abordado de manera estructurada, construyendo cada máquina de forma progresiva y verificando su funcionamiento mediante simulaciones con diferentes cadenas de prueba. En las siguientes secciones se documentan las decisiones de diseño adoptadas, la definición de estados y transiciones, así como los resultados obtenidos durante la ejecución y validación de cada ejercicio.

En primer lugar, se describe la construcción de una Máquina de Turing capaz de reconocer el lenguaje:

$$\mathrm{L} = \{0^{\mathrm{n}}\mathrm{n}\ 1^{\mathrm{n}} : \mathrm{n} > 0\}$$

Detallando la estrategia de marcado y emparejamiento utilizada para comprobar que la cantidad de ceros y unos sea exactamente la misma, respetando además el orden requerido en la cadena.

Posteriormente, se desarrolla una segunda Máquina de Turing diseñada para calcular el número consecutivo (sucesor) de un número binario dado como entrada. En este caso, se explica el procedimiento de suma binaria implementado sobre la cinta, incluyendo la propagación de acarreo necesaria cuando el número termina en uno o varios bits iguales a 1.

Cada etapa del laboratorio incluye la verificación mediante ejemplos de cadenas aceptadas y rechazadas, lo que permite comprobar el correcto funcionamiento de las máquinas diseñadas y reforzar la comprensión práctica del modelo computacional propuesto por Alan Turing.

Diseño De Máquina De Turing Que Acepta El Lenguaje Propuesto

En este primer ejercicio se diseñó e implementó una Máquina de Turing utilizando la herramienta JFLAP, con el objetivo de reconocer el lenguaje:

$$\mathrm{L} = \{0^{\mathrm{n}}\mathrm{n}\ 1^{\mathrm{n}} : \mathrm{n} > 0\}$$

Este lenguaje está formado por cadenas compuestas por una cantidad n de ceros seguidos de exactamente la misma cantidad n de unos, con la condición de que n sea mayor que cero.

Debido a que este tipo de lenguaje no puede ser reconocido por autómatas finitos, se requiere el uso de una Máquina de Turing que permita realizar un proceso de comparación y emparejamiento entre símbolos.

Para lograrlo, se empleó una estrategia basada en el marcado progresivo de símbolos en la cinta, emparejando cada 0 con un 1 correspondiente hasta verificar que ambas cantidades coinciden. A continuación, se presenta el desarrollo paso a paso del diseño y validación de la máquina en JFLAP.

1. Análisis del lenguaje

- El lenguaje $\mathrm{L} = \{0^{\mathrm{n}}\mathrm{n}\ 1^{\mathrm{n}} : \mathrm{n} > 0\}$ exige que:
- Todos los ceros aparezcan primero
- Todos los unos aparezcan después
- La cantidad de ceros sea igual a la cantidad de unos
- No se acepta la cadena vacía (ε)

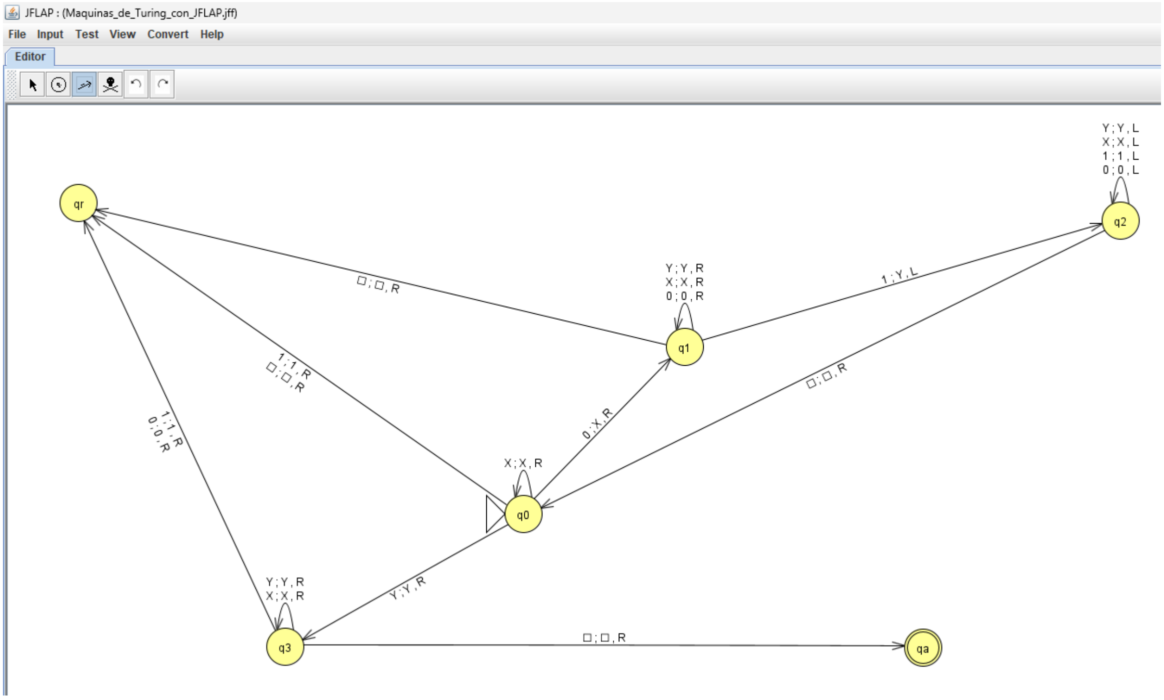
Ejemplos aceptados:

- 01
- 0011
- 000111
- 00001111
- 0000011111

Ejemplos rechazados:

- ε
- 1
- 001
- 0111
- 0101

Figura 1. Diagrama completo de la Máquina de Turing construida en JFLAP para reconocer el lenguaje $\mathrm{L} = \{0^{\mathrm{n}}\mathrm{n}\ 1^{\mathrm{n}} : \mathrm{n} > 0\}$



2. Estrategia utilizada

- La Máquina de Turing funciona mediante un algoritmo de emparejamiento:
- Buscar el primer 0 sin marcar y reemplazarlo por X
- Buscar el primer 1 sin marcar y reemplazarlo por Y
- Regresar al inicio de la cinta
- Repetir el proceso hasta que no queden ceros
- Verificar que no queden unos sobrantes
- Aceptar si todo fue emparejado correctamente

Figura 2. Resultados de las pruebas realizadas en JFLAP, mostrando cadenas aceptadas y rechazadas por la máquina.

Table Text Size

Input	Result
01	Accept
0011	Accept
000111	Accept
00001111	Accept
0000011111	Accept
	Reject
1	Reject
001	Reject
0111	Reject
0101	Reject

3. Diseño de la Máquina en JFLAP

Se construyó una Máquina de Turing con los siguientes estados:

Estado	Función	Símbolos procesados
q0	Buscar un 0 sin marcar	Lee 0, X, Y, o □
q1	Buscar el 1 correspondiente	Lee 0, Y, 1, o □
q2	Regresar al inicio	Lee 0, Y, X
q3	Verificación final	Lee Y, 0, 1, o □
qa	Estado de aceptación	Estado final
qr	Estado de rechazo	Estado trampa

La máquina marca:

- 0 → X (cero procesado)
- 1 → Y (uno procesado)

4. Funcionamiento general

El funcionamiento se basa en recorrer la cinta repetidamente:

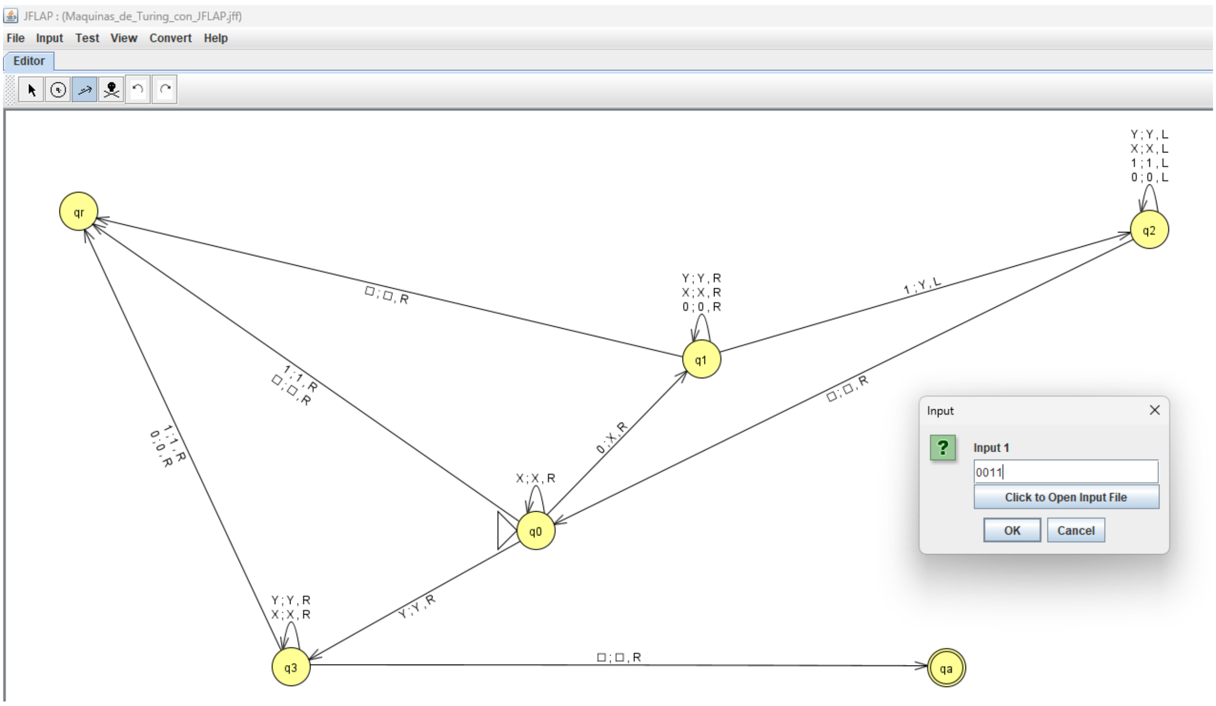
- Cada ciclo marca un par (0,1)
- Cuando ya no hay ceros, se pasa a verificación
- Si solo quedan símbolos marcados y blancos, se acepta

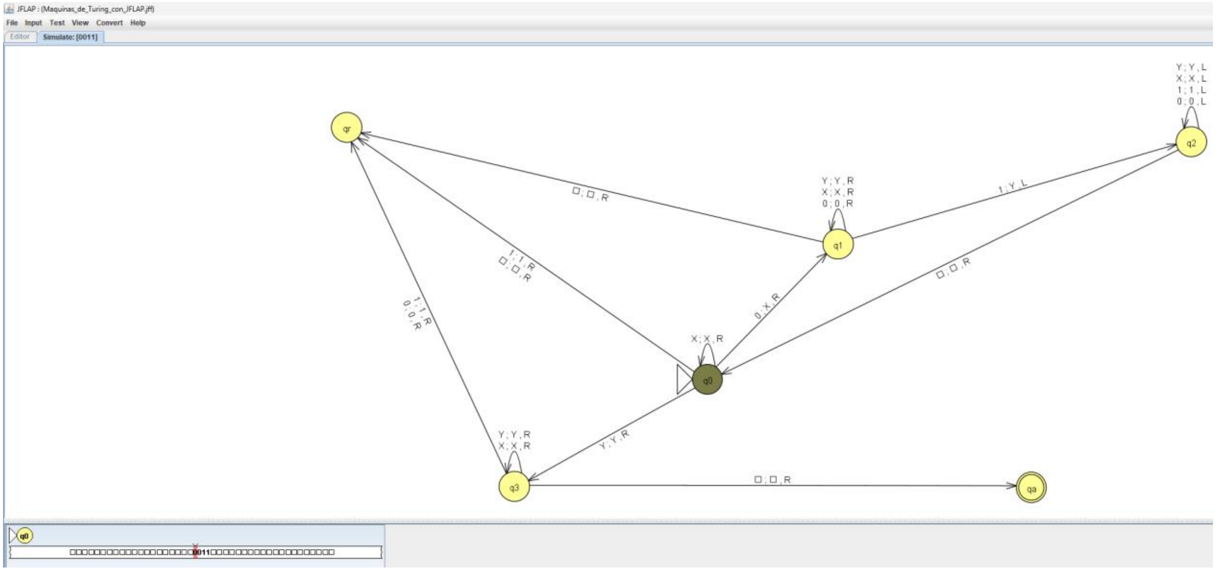
Ejemplo con entrada 0011:

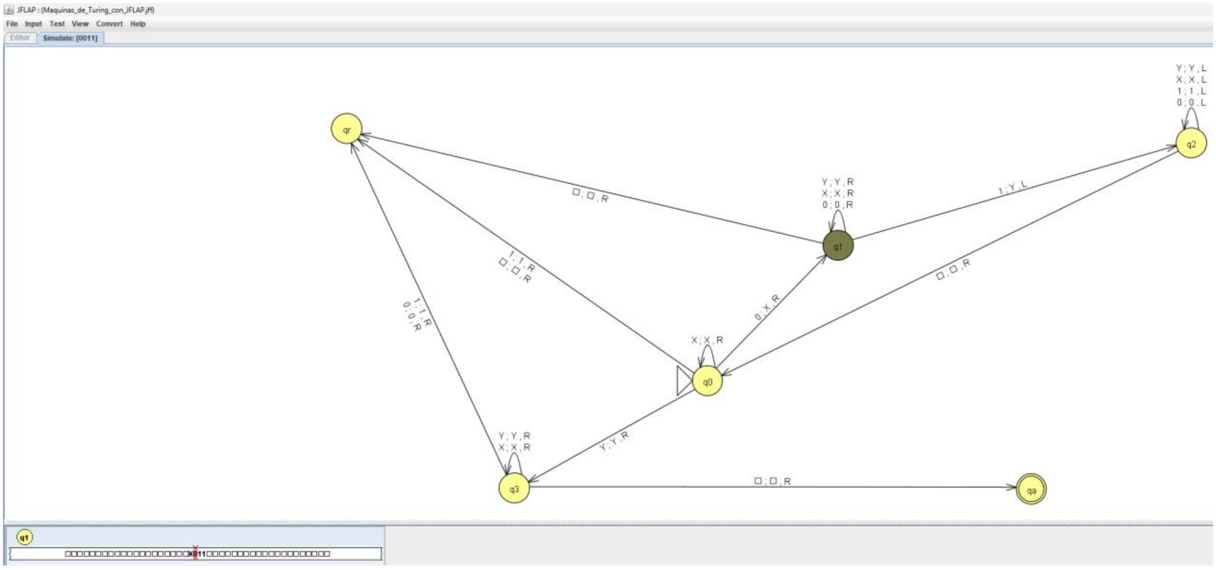
- X011
- X0Y1
- XXY1
- XXYY
- Verificación → Acepta

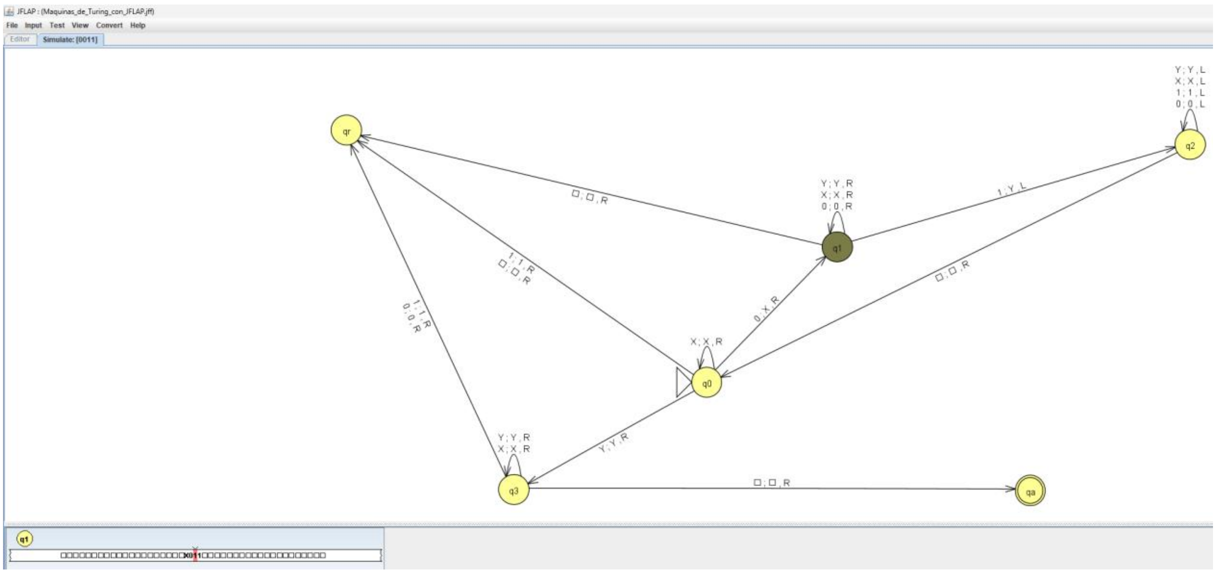
5. Imágenes del Desarrollo Para 0011:

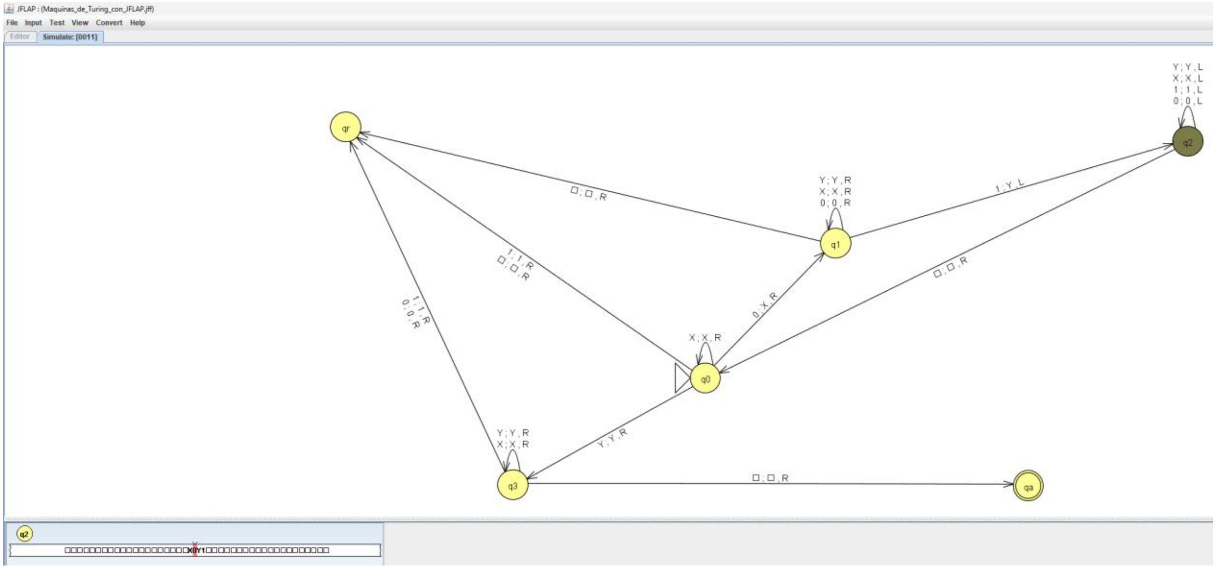
A continuación, denoto las imágenes de la ejecución completa del código, si por algún motivo no se llega a ver correctamente, tener presente que se adjunta el archivo con el desarrollo de este ejercicio en JFLAP de nombre “Maquinas_de_Turing_con_JFLAP_Ejercicio_1”, de igual manera tener presente que en la parte de debajo de grafico de flujo se ve cómo va corriendo en consola el desarrollo y cómo van cambiando los datos, hasta llegar a la última imagen donde la consola se ve en verde lo que indica su aceptación:

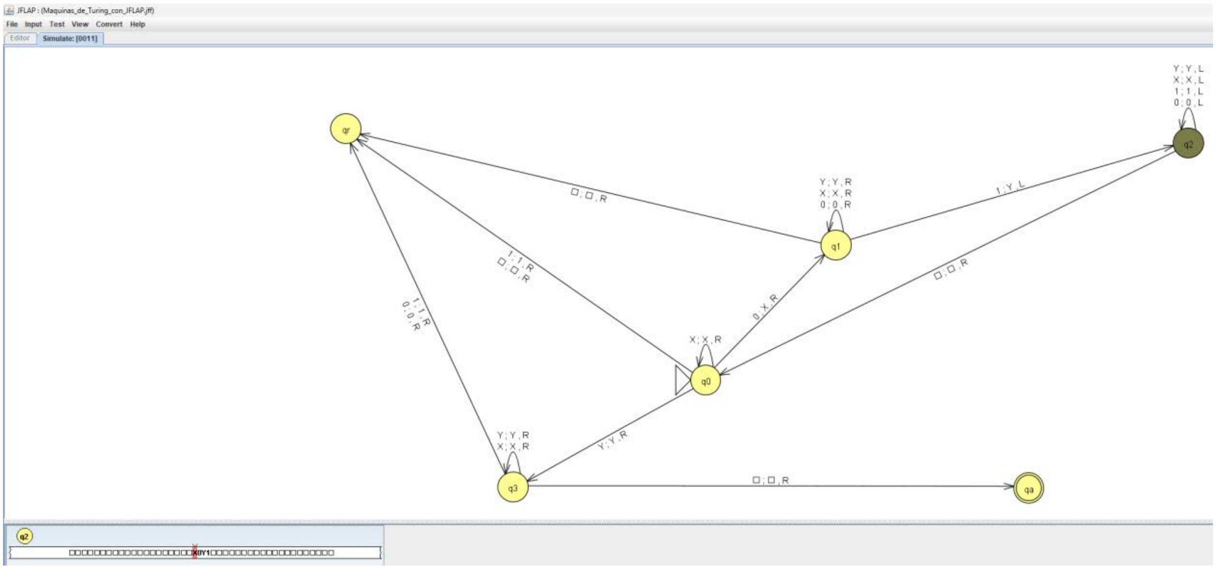


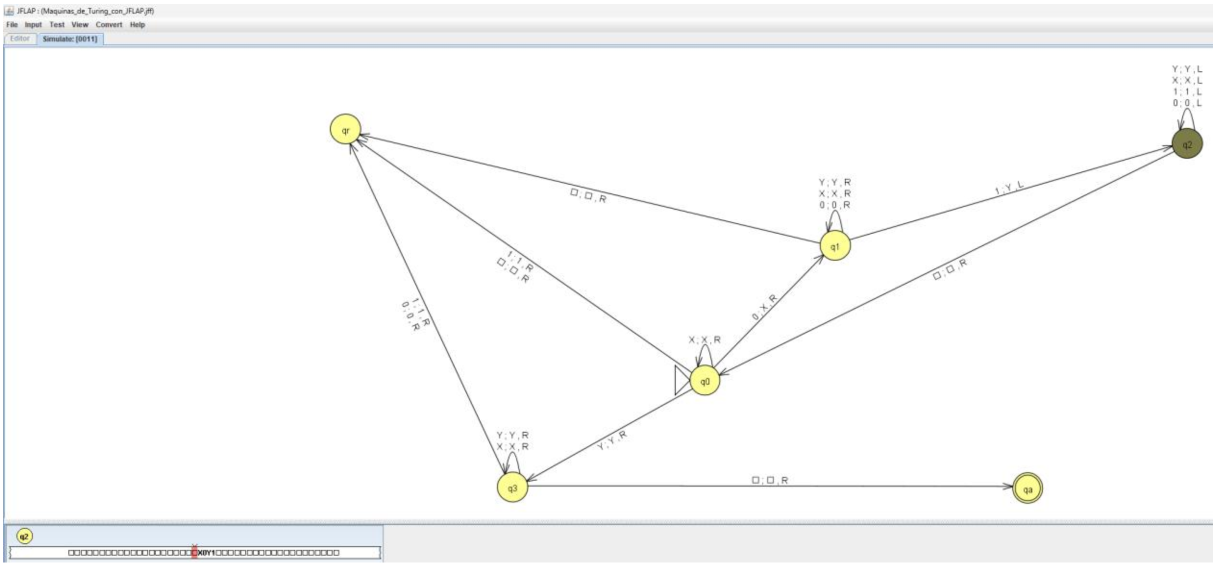


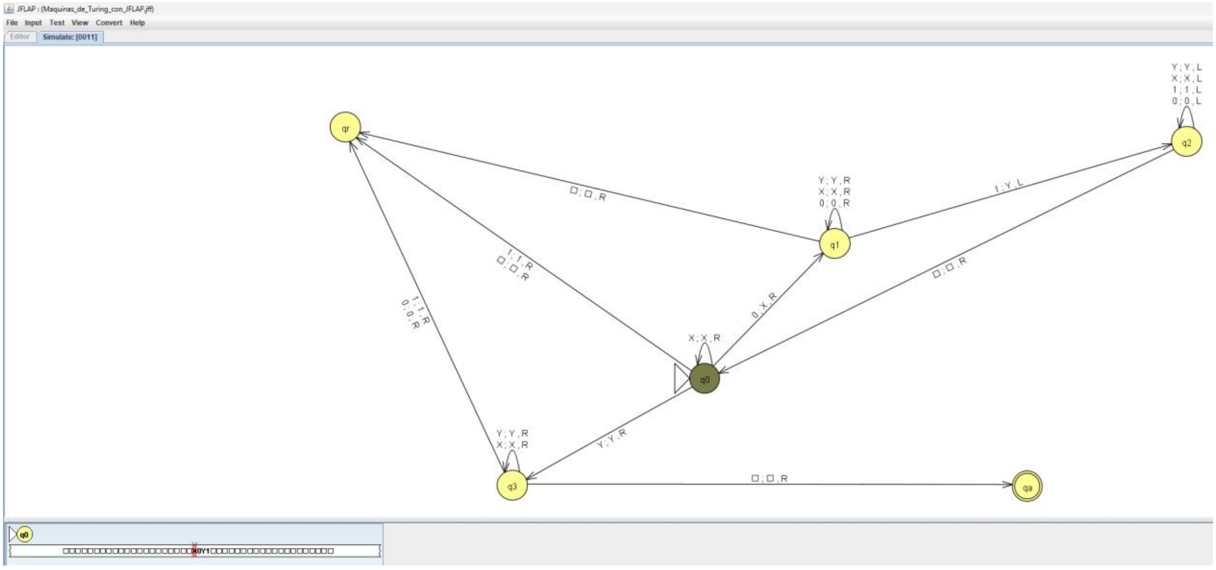


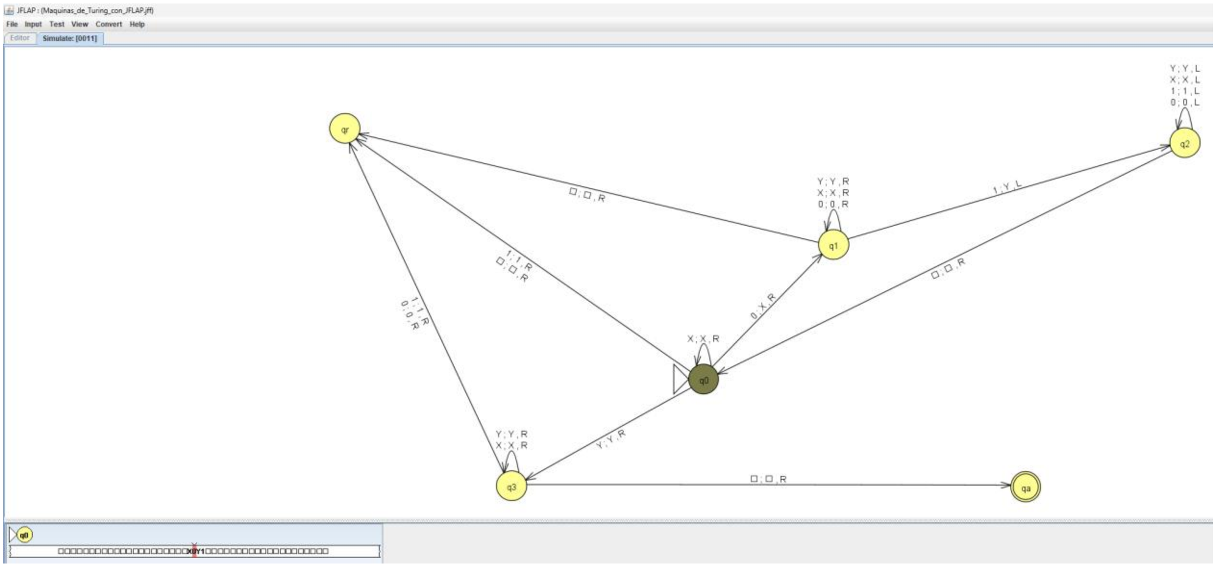


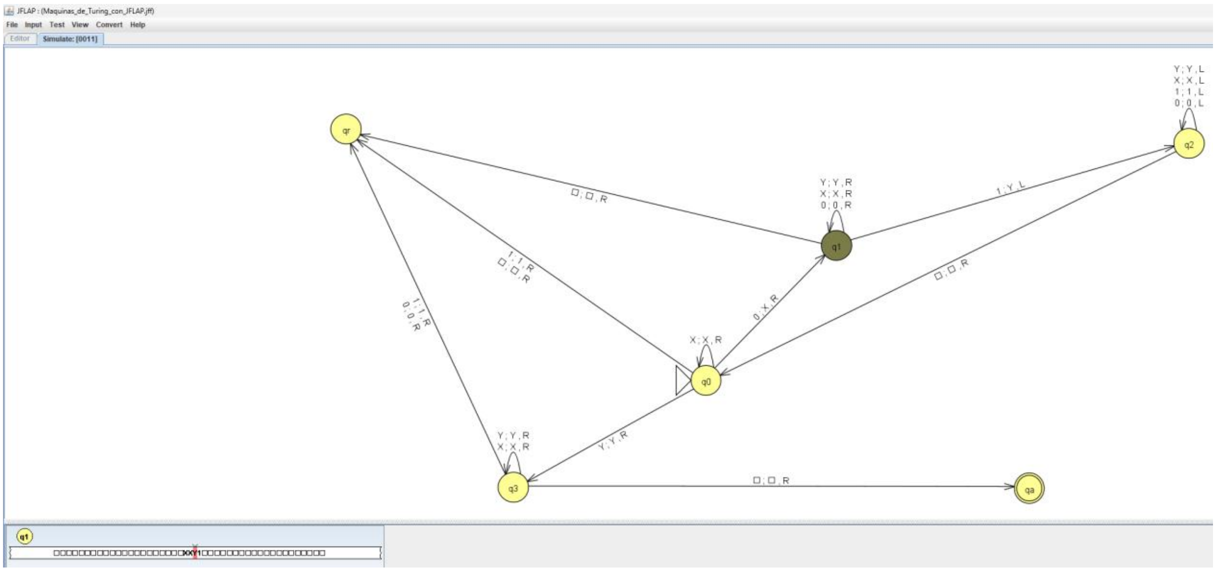


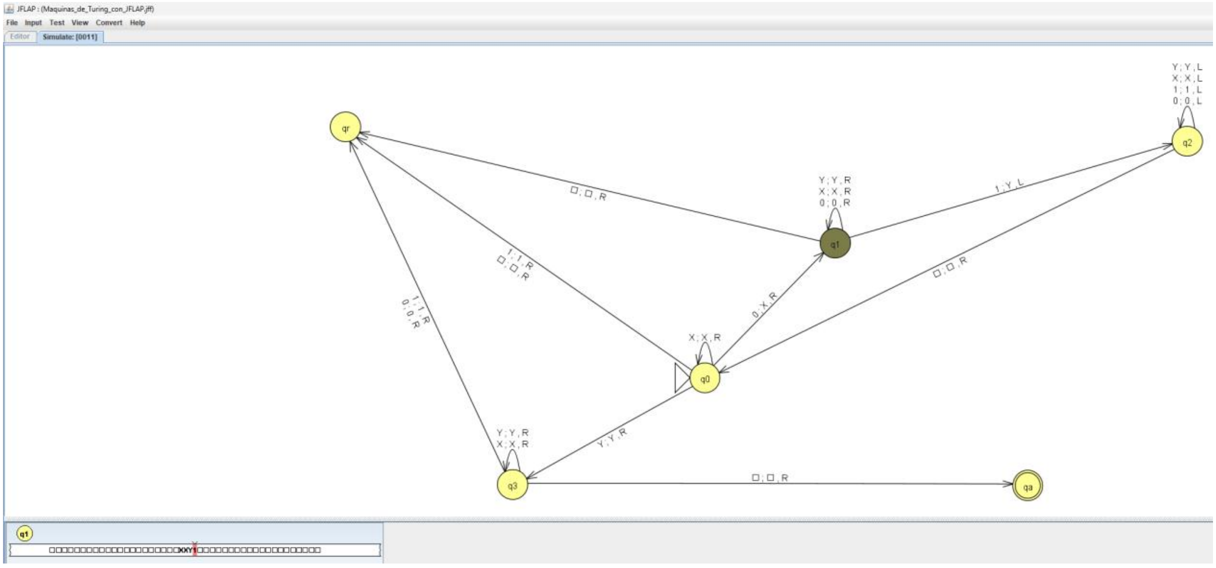


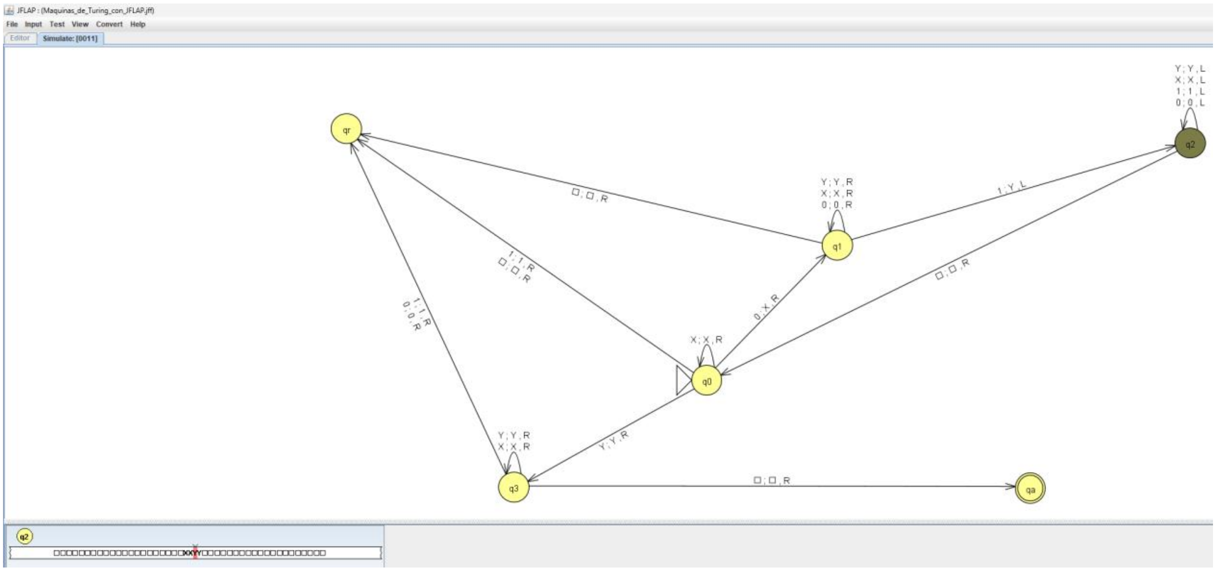


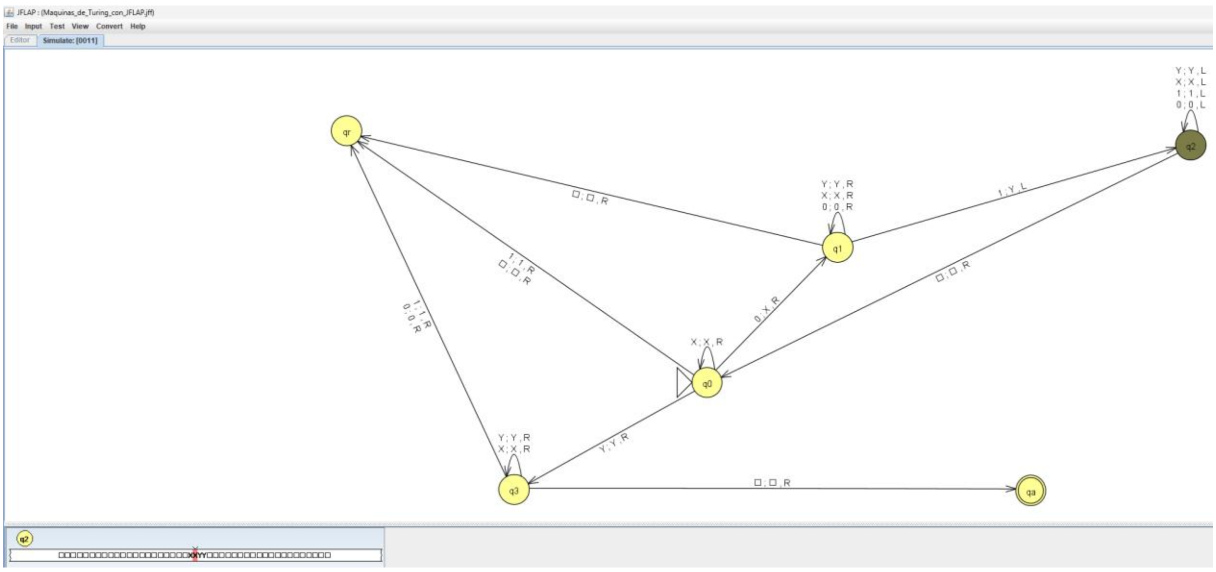

JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
qr
q0
q1
q2
q3
qa
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
□ ; □ , R
0 ; X , R
X ; X , R
1 ; 1 , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
Y ; Y , R
□ ; □ , R

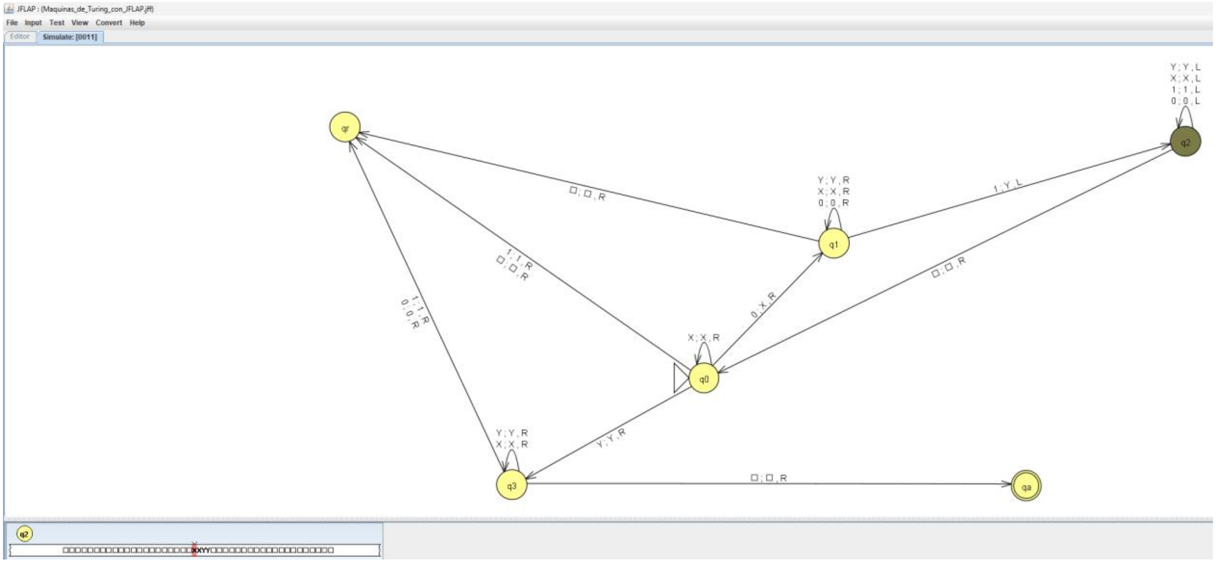

JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
qr
q0
q1
q2
q3
qa
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
□ ; □ , R
0 ; X , R
X ; X , R
1 ; 1 , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
Y ; Y , R
□ ; □ , R

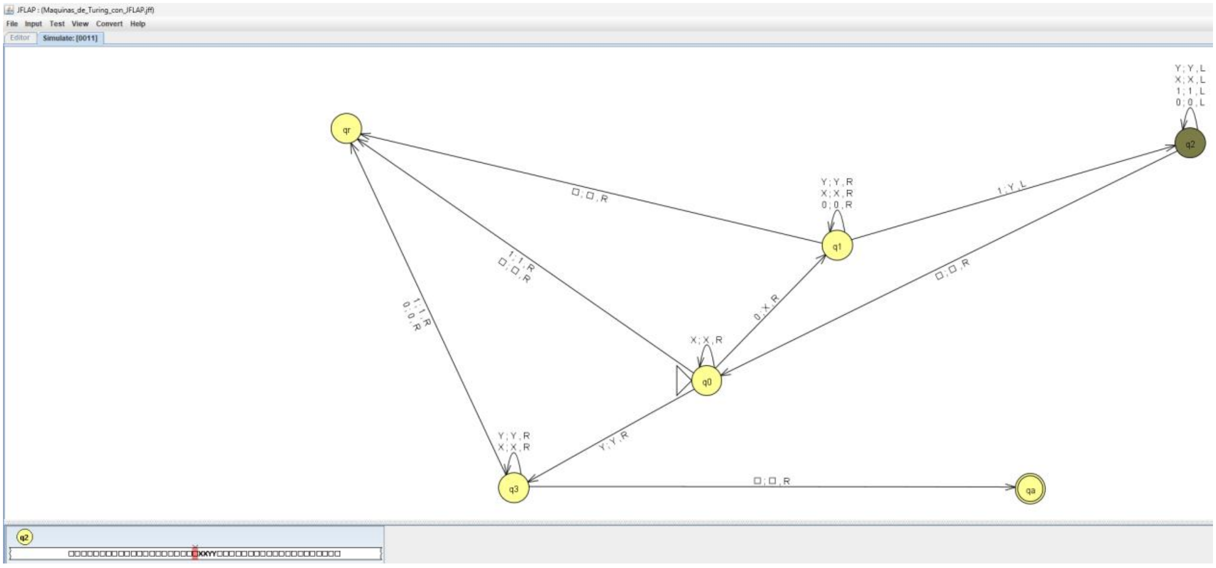

JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
qr
q0
q1
q2
q3
qa
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
□ ; □ , R
0 ; X , R
X ; X , R
1 ; 1 , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
Y ; Y , R
□ ; □ , R

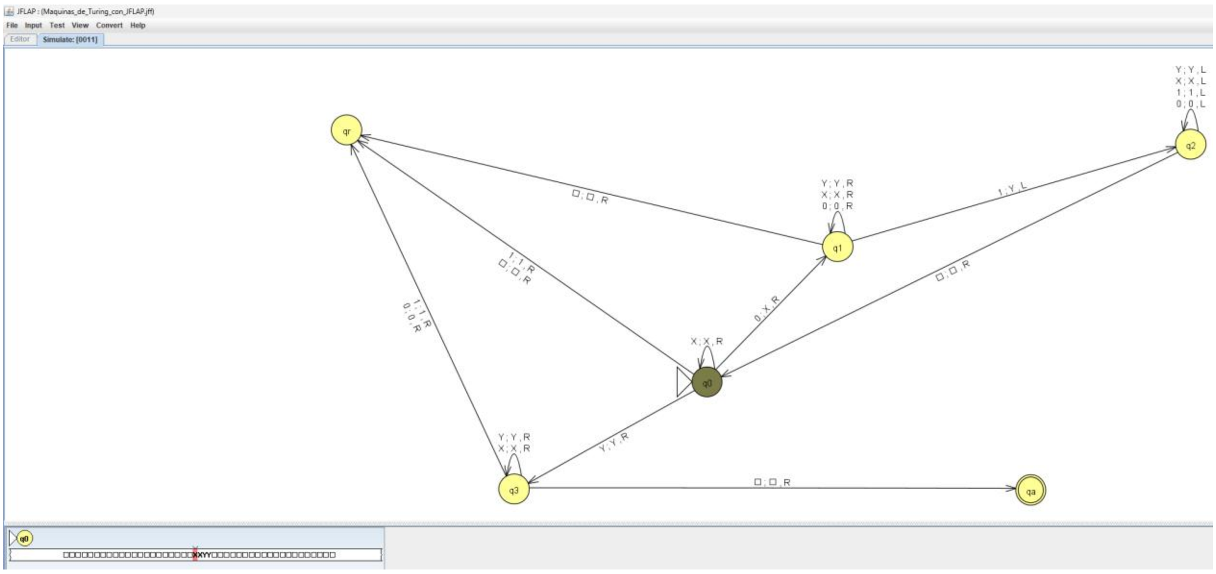

JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
qr
q2
□ ; □ , R
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
q1
1 ; 1 , R
□ ; □ , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
0 ; X , R
X ; X , R
q0
Y ; Y , R
X ; X , R
Y ; Y , R
q3
□ ; □ , R
qa

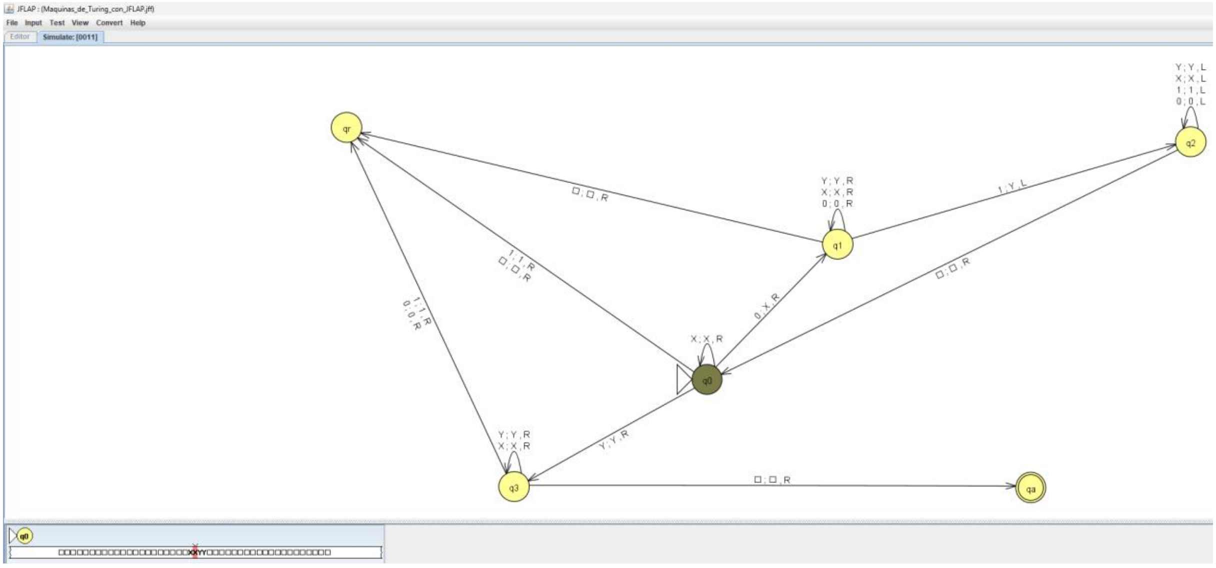

JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
qr
q2
□ ; □ , R
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
q1
1 ; 1 , R
□ ; □ , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
0 ; X , R
X ; X , R
q0
Y ; Y , R
X ; X , R
Y ; Y , R
q3
□ ; □ , R
qa


JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
qr
q2
□ ; □ , R
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
q1
1 ; 1 , R
□ ; □ , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
0 ; X , R
X ; X , R
q0
Y ; Y , R
X ; X , R
Y ; Y , R
q3
□ ; □ , R
qa

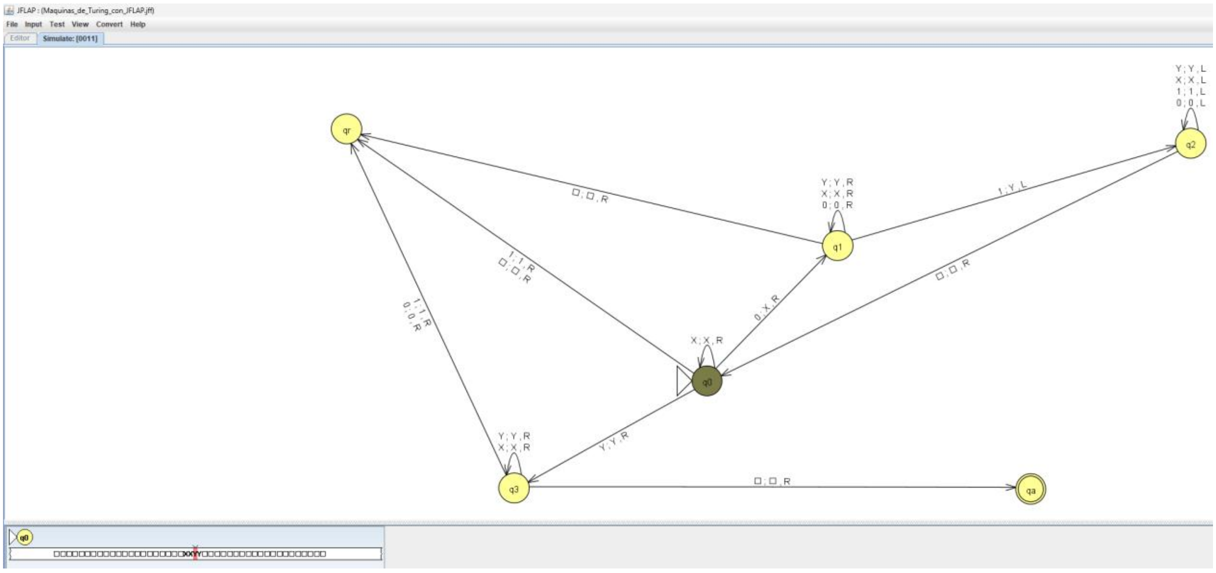

JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
qr
q0
q1
q2
q3
qa
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
Y ; Y , R
X ; X , R
0 ; 0 , R
□ ; □ , R
1 ; Y , L
0 ; X , R
X ; X , R
1 ; 1 , R
Y ; Y , R


JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
qr
q0
q1
q2
q3
qa

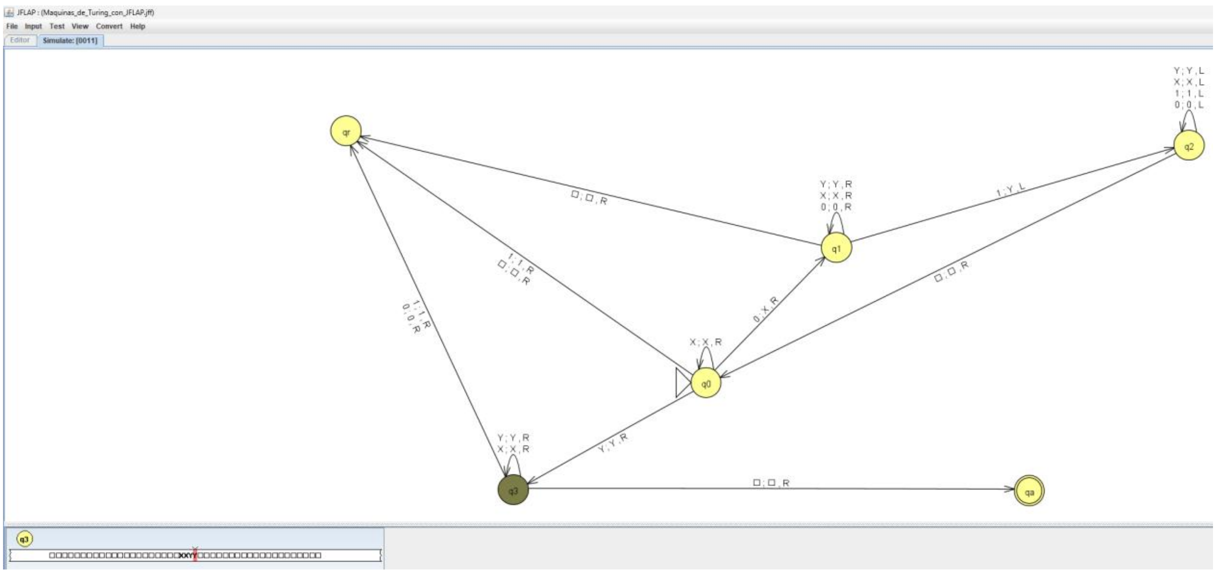

JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
qr
q0
q1
q2
q3
qa



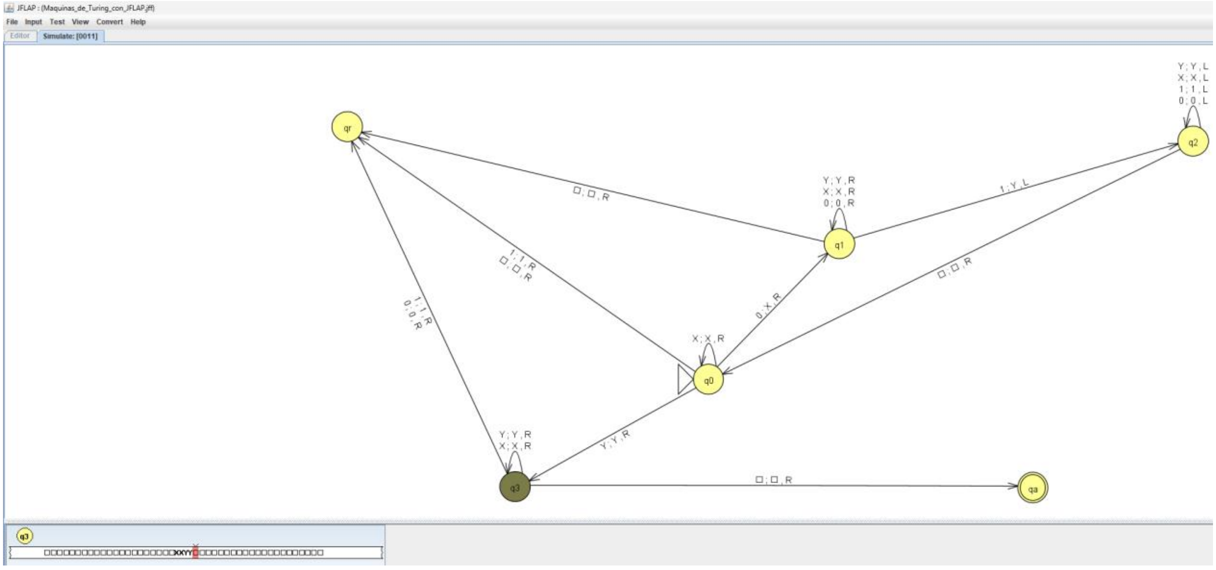
JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
qr
q2
□ ; □ , R
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
q1
1 ; 1 , R
□ ; □ , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
0 ; X , R
X ; X , R
q0
Y ; Y , R
X ; X , R
Y ; Y , R
q3
□ ; □ , R
qa
q0

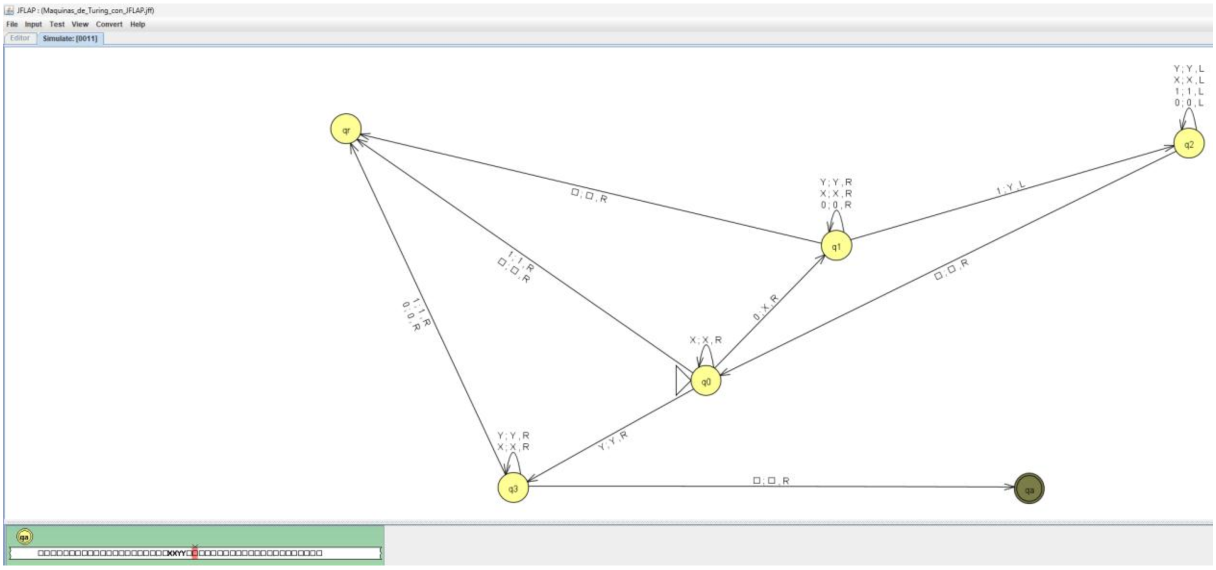


JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
qr
q2
□ ; □ , R
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
q1
1 ; 1 , R
□ ; □ , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
0 ; X , R
X ; X , R
q0
Y ; Y , R
X ; X , R
Y ; Y , R
q3
□ ; □ , R
qa
q0



JFLAP : (Maquinas_de_Turing_con_JFLAP.jff)
File Input Test View Convert Help
Editor
Simulate: [0011]
Y ; Y , L
X ; X , L
1 ; 1 , L
0 ; 0 , L
qr
q2
□ ; □ , R
Y ; Y , R
X ; X , R
0 ; 0 , R
1 ; Y , L
q1
1 ; 1 , R
□ ; □ , R
□ ; □ , R
1 ; 1 , R
0 ; 0 , R
0 ; X , R
X ; X , R
q0
Y ; Y , R
X ; X , R
Y ; Y , R
q3
□ ; □ , R
qa
q3





5. Validación en JFLAP

Se realizaron pruebas con cadenas aceptadas y rechazadas para comprobar el funcionamiento.

A continuación, se detallan las cadenas aceptadas y las rechazadas para dar mejor claridad al funcionamiento y explicación, que se complementan con las imágenes paso por paso del desarrollo y su resolución:

Cadenas aceptadas (5)

Entrada	Resultado
01	Accept
0011	Accept
000111	Accept
00001111	Accept
0000011111	Accept

Cadenas rechazadas (5)

Entrada	Resultado
ε	Reject
1	Reject
001	Reject
0111	Reject
0101	Reject

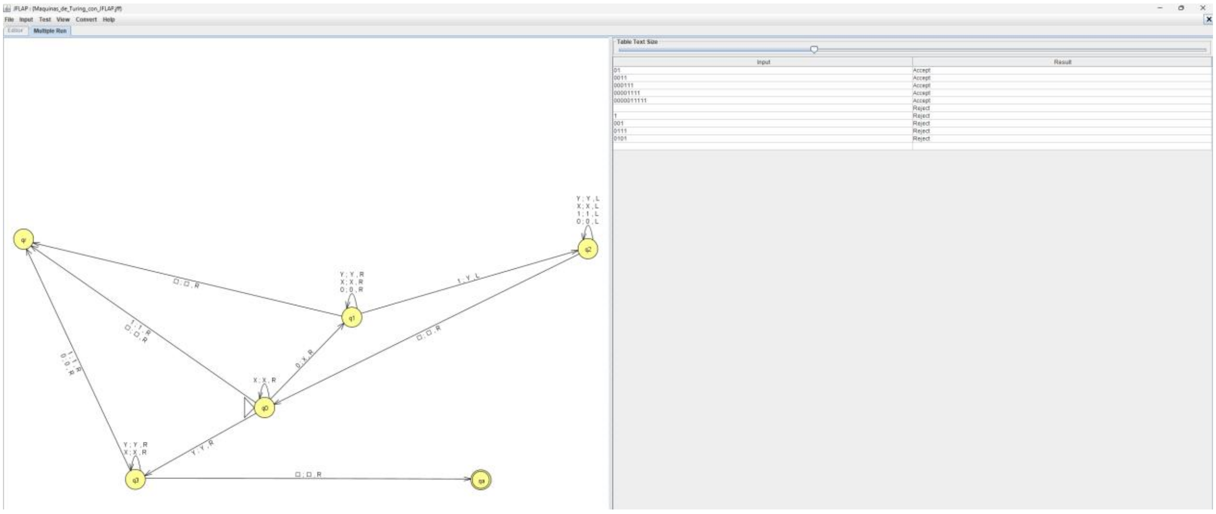
6. Resultado final

La simulación confirmó que la Máquina de Turing alcanza correctamente el estado de aceptación qa, como muestra el mensaje:

"Accepting configuration found!"

Esto demuestra que el diseño implementado reconoce correctamente el lenguaje $0^n 1^n$

7. Imagen de Todo el Desarrollo:



8. Conclusión Final Primer Ejercicio

En este ejercicio se diseñó una Máquina de Turing capaz de reconocer el lenguaje $\mathrm{L} = \{0^\mathrm{n} \mathrm{n}\ 1^\mathrm{n} : \mathrm{n} > 0\}$ mediante un proceso de marcado y emparejamiento entre ceros y unos. Las pruebas realizadas en JFLAP demostraron que la máquina acepta únicamente las cadenas que cumplen con la condición de igualdad y orden, rechazando aquellas que no pertenecen al lenguaje. Este ejercicio evidencia la capacidad de las Máquinas de Turing para reconocer lenguajes no regulares que no pueden ser procesados por autómatas finitos.

Máquina De Turing Diseñada Calcula El Número Consecutivo (Sucesor) De Un Número Binario Dado Como Entrada

En este segundo ejercicio se diseñó e implementó una Máquina de Turing utilizando JFLAP con el objetivo de calcular el número consecutivo (sucesor) de un número binario dado como entrada. Es decir, para una cadena binaria que representa un número n, la máquina debe producir en la cinta el resultado n+1.

Este problema permite evidenciar que las Máquinas de Turing no solo pueden reconocer lenguajes formales, sino también ejecutar procedimientos computacionales concretos, como la suma binaria mediante propagación de acarreo.

1. Análisis del problema

La máquina recibe como entrada un número binario válido, por ejemplo:

- Entrada: 1 → Salida: 10
- Entrada: 10 → Salida: 11
- Entrada: 111 → Salida: 1000
- Entrada: 1011 → Salida: 1100

El objetivo es simular el algoritmo de suma binaria de manera automática sobre la cinta.

2. Estrategia utilizada

La Máquina de Turing implementa el siguiente algoritmo:

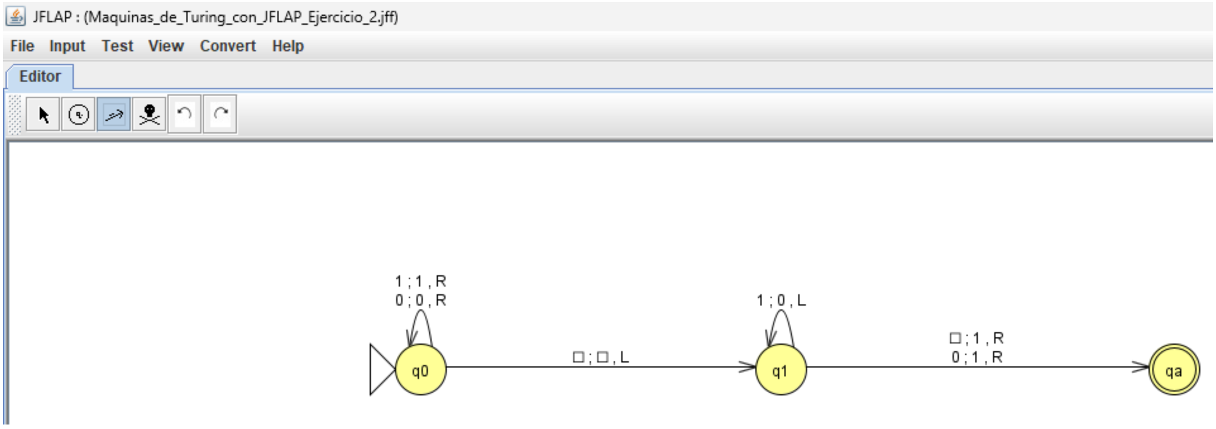
- Desplazarse hasta el final de la cadena (bit menos significativo).
- Sumar 1 aplicando reglas de acarreo:
    - Si encuentra un 0, lo convierte en 1 y finaliza.
    - Si encuentra un 1, lo convierte en 0 y continúa hacia la izquierda.
    - Si todos los bits eran 1, se añade un nuevo 1 al inicio.

3. Diseño de la Máquina de Turing

Para resolver el problema se construyó una máquina simple con tres estados:

Estado	Función
q0	Recorrer la cinta hasta el final del número
q1	Aplicar la suma con acarreo hacia la izquierda
qa	Estado de aceptación

Grafo completo de la Máquina de Turing del Ejercicio 2 en JFLAP:



Nota importante sobre el comportamiento de esta máquina:

A diferencia del Ejercicio 1, donde la máquina reconocedora acepta o rechaza cadenas según pertenezcan o no al lenguaje L = {0^n1^n}, esta Máquina de Turing actúa como una máquina computadora que siempre acepta cualquier número binario válido como entrada. Su función no es reconocer un lenguaje específico, sino realizar una transformación sobre la entrada: calcular el sucesor (n+1) del número binario n proporcionado. Por lo tanto, cualquier cadena formada únicamente por los símbolos {0, 1} será procesada exitosamente, modificando la cinta para mostrar el resultado de la operación aritmética y alcanzando el estado de aceptación qa.

En este sentido, no existen "cadenas rechazadas" en el contexto tradicional, ya que la máquina está diseñada para computar un resultado, no para decidir membresía en un lenguaje.

4. Funcionamiento general

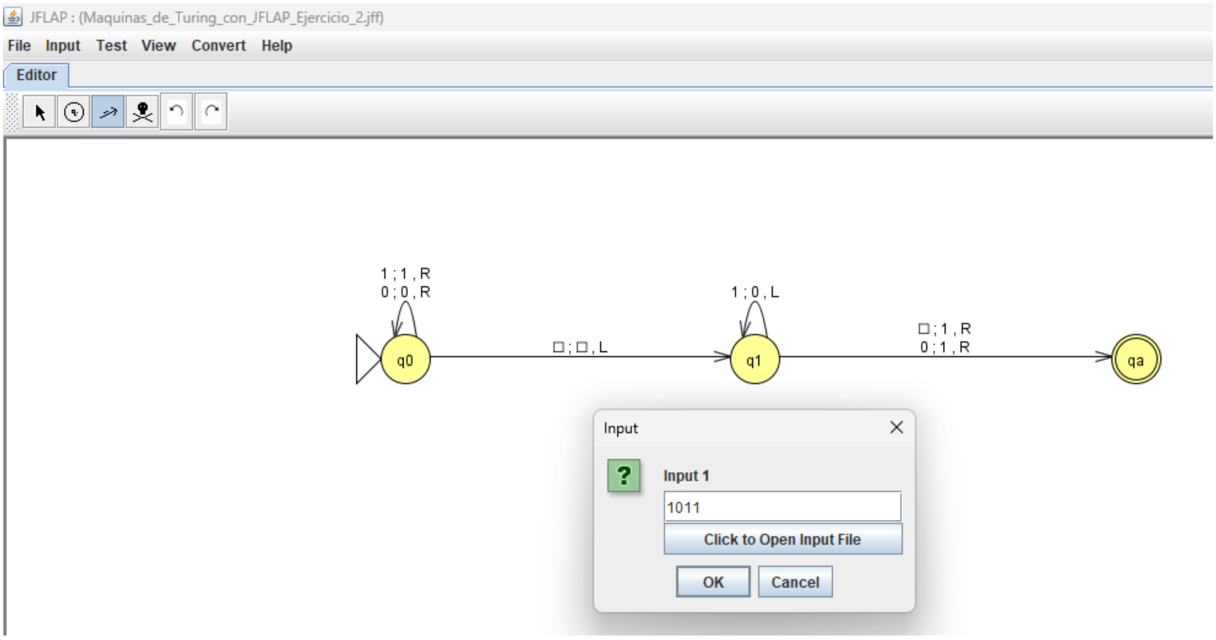
La máquina primero se mueve hacia la derecha hasta encontrar el símbolo blanco (□). Luego retrocede una posición y comienza el proceso de suma.

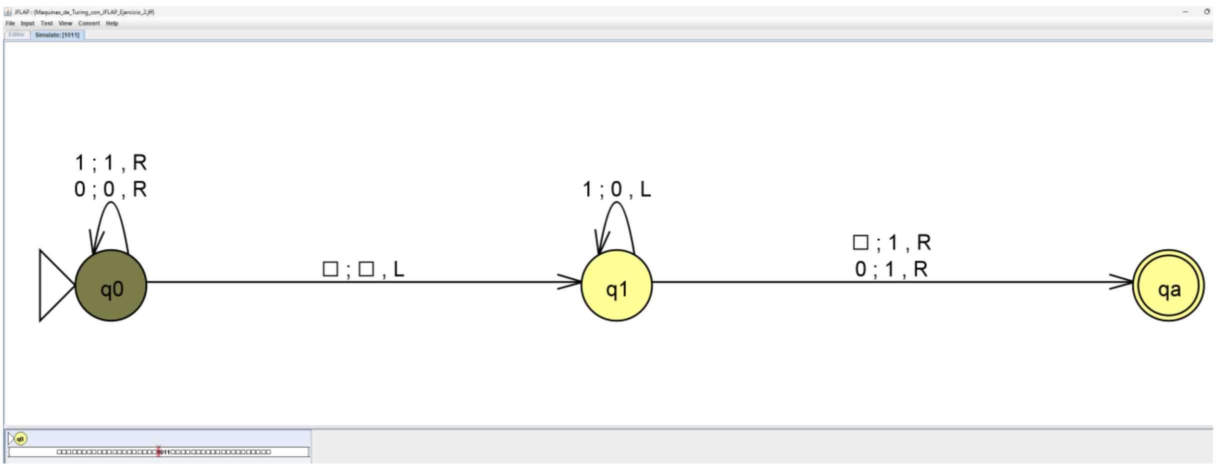
Ejemplo con entrada 1011:

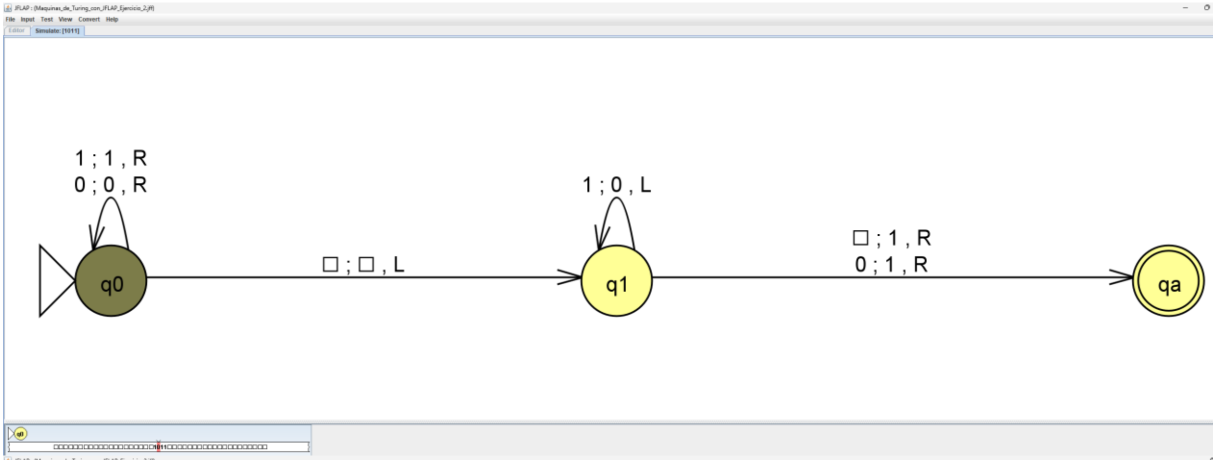
- El último bit es 1 → se convierte en 0 y se propaga el acarreo.
- El siguiente bit también es 1 → se convierte en 0.
- Se encuentra un 0 → se convierte en 1.
- Resultado final: 1100

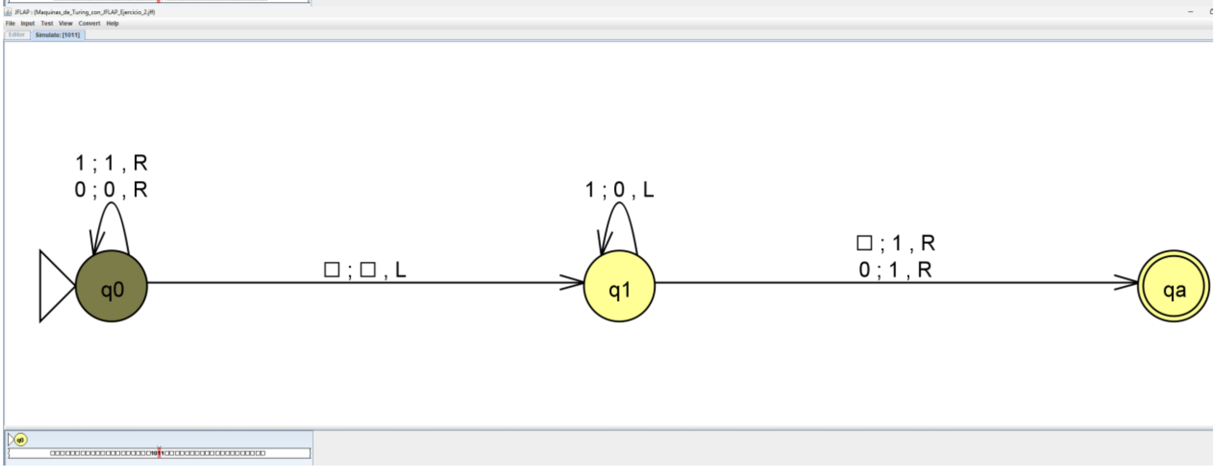
Captura de simulación mostrando el resultado final en cinta o mensaje de aceptación.

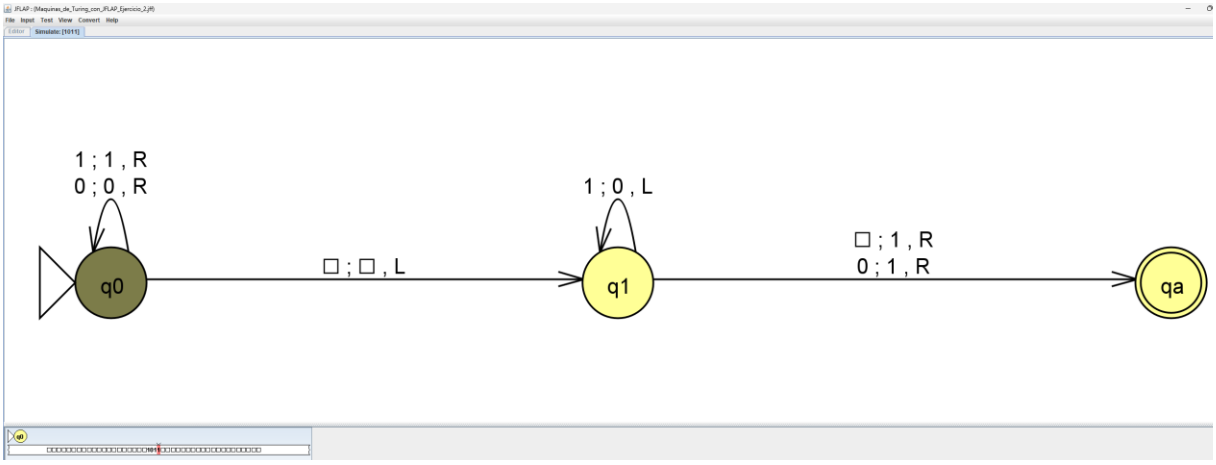
A continuación, denoto las imágenes de la ejecución completa del código, si por algún motivo no se llega a ver correctamente, tener presente que se adjunta el archivo con el desarrollo de este ejercicio en JFLAP de nombre “Maquinas_de_Turing_con_JFLAP_Ejercicio_2”, de igual manera tener presente que en la parte de debajo de grafico de flujo se ve cómo va corriendo en consola el desarrollo y cómo van cambiando los datos, hasta llegar a la última imagen donde la consola se ve en verde lo que indica su aceptación:

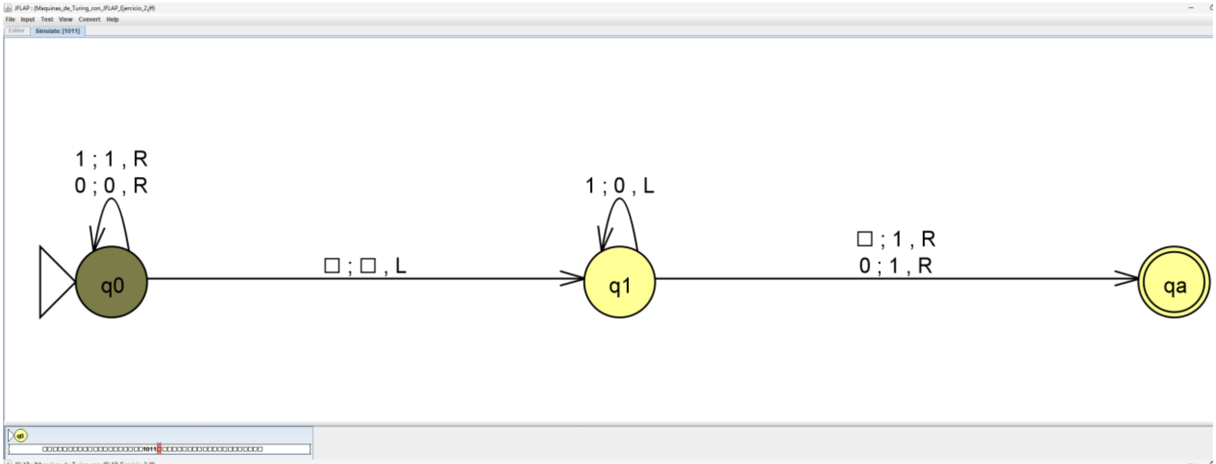


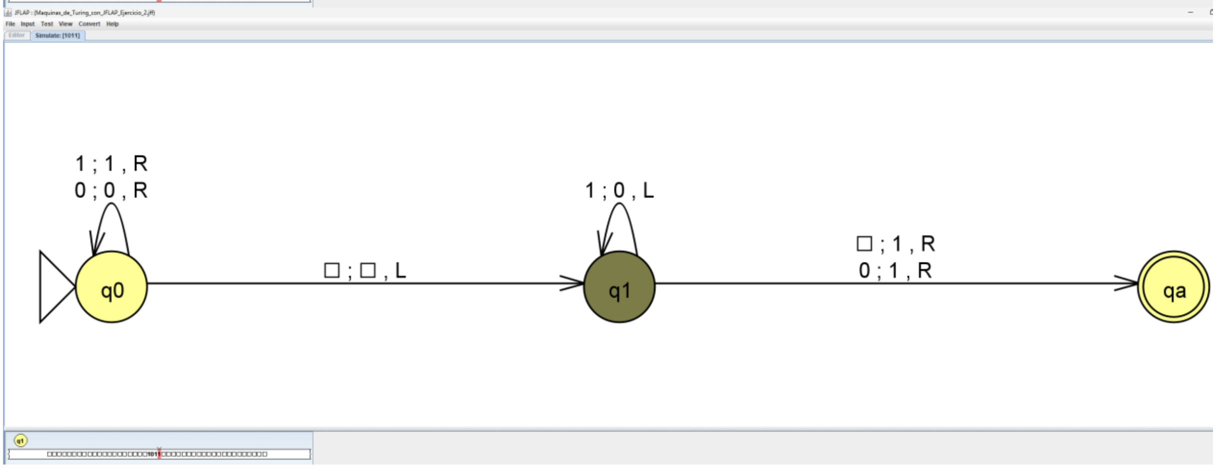


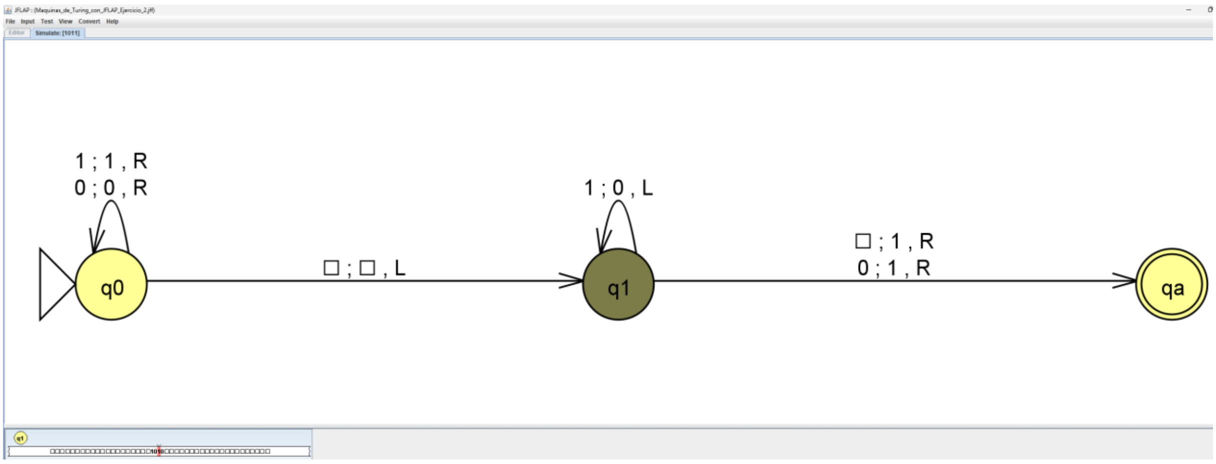

1 ; 1 , R
0 ; 0 , R
q0
□ ; □ , L
1 ; 0 , L
q1
□ ; 1 , R
0 ; 1 , R
qa

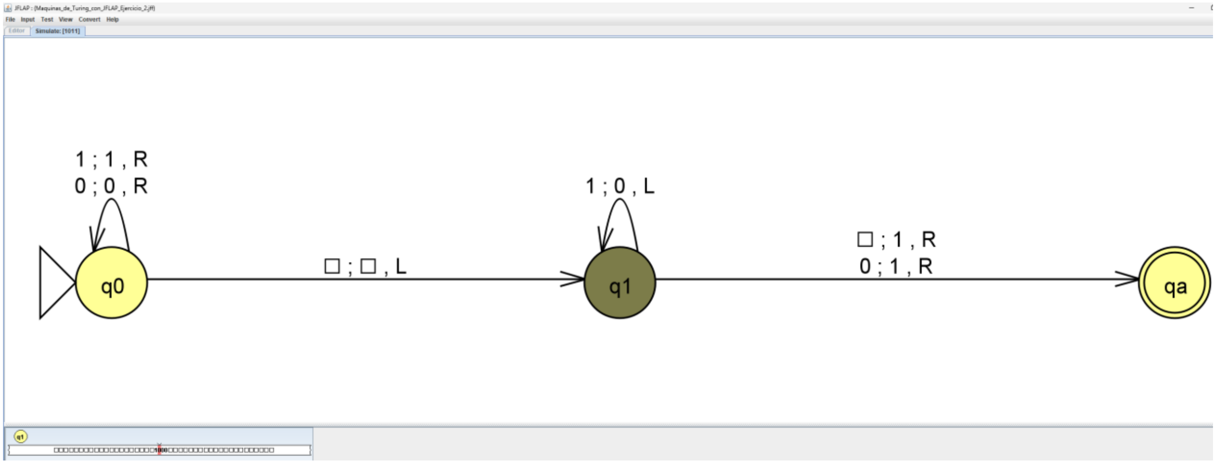

1 ; 1 , R
0 ; 0 , R
q0
□ ; □ , L
1 ; 0 , L
q1
□ ; 1 , R
0 ; 1 , R
qa

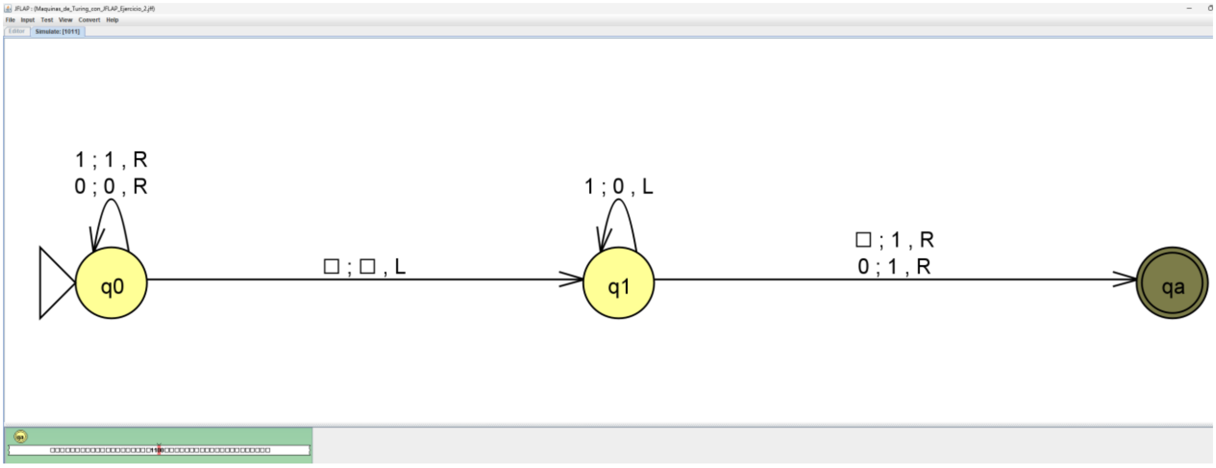

1 ; 1 , R
0 ; 0 , R
q0
□ ; □ , L
1 ; 0 , L
q1
□ ; 1 , R
0 ; 1 , R
qa


1 ; 1 , R
0 ; 0 , R
q0
□ ; □ , L
1 ; 0 , L
q1
□ ; 1 , R
0 ; 1 , R
qa


1 ; 1 , R
0 ; 0 , R
q0
□ ; □ , L
1 ; 0 , L
q1
□ ; 1 , R
0 ; 1 , R
qa


1 ; 1 , R
0 ; 0 , R
q0
□ ; □ , L
1 ; 0 , L
q1
□ ; 1 , R
0 ; 1 , R
qa





5. Resultados de prueba

Se realizaron pruebas con diferentes números binarios para comprobar el funcionamiento correcto:

Entrada	Salida esperada
0	1
1	10
10	11
111	1000
1011	1100

Todas las pruebas fueron aceptadas correctamente en JFLAP.

Captura del panel “Multiple Run” mostrando las entradas y aceptación.

6. Conclusión Final Segundo Ejercicio

En este ejercicio se implementó exitosamente una Máquina de Turing capaz de calcular el sucesor de un número binario. La máquina aplicó correctamente el algoritmo de suma con acarreo, demostrando que este modelo computacional no solo sirve para reconocer lenguajes, sino también para realizar operaciones aritméticas básicas.

Este ejercicio refuerza el papel de las Máquinas de Turing como un modelo universal de computación.

CONCLUSIONES DE LA ACTIVIDAD

Para concluir este trabajo, puedo afirmar que el desarrollo del laboratorio permitió comprender de manera práctica el funcionamiento y alcance de las Máquinas de Turing como modelo formal de computación. A través del diseño e implementación en JFLAP, se evidenció cómo este modelo abstracto no solo es capaz de reconocer lenguajes no regulares, sino también de ejecutar procedimientos computacionales concretos, como el cálculo del sucesor binario.

A diferencia de un enfoque puramente teórico, en el cual las máquinas se describen únicamente mediante tablas formales, la implementación práctica permitió construir el conocimiento de forma progresiva y estructurada. Cada ejercicio se desarrolló paso a paso, comenzando con el análisis del problema, la definición de la estrategia, el diseño de los estados y transiciones, y finalmente la validación mediante pruebas en el simulador. Este proceso organizado facilitó la comprensión de cada componente de la máquina y permitió detectar y corregir errores durante la construcción.

En el primer ejercicio, correspondiente al lenguaje $\mathrm{L} = \{0^{\mathrm{n}}\mathrm{n}\ 1^{\mathrm{n}}\!: \mathrm{n} > 0\}$, se demostró la capacidad de la Máquina de Turing para reconocer un lenguaje que no puede ser aceptado por autómatas finitos. La estrategia de marcado y emparejamiento de símbolos permitió verificar la igualdad entre la cantidad de ceros y unos, asegurando además el cumplimiento del orden requerido. Las pruebas realizadas en JFLAP confirmaron que la máquina acepta únicamente las cadenas que pertenecen al lenguaje y rechaza aquellas que no cumplen las condiciones establecidas.

Un aspecto fundamental evidenciado en este laboratorio es la distinción entre dos tipos de Máquinas de Turing según su propósito computacional:

- Máquinas reconocedoras (Ejercicio 1): Su objetivo es decidir si una cadena pertenece o no a un lenguaje formal específico. Estas máquinas implementan un procedimiento de decisión que culmina en estados de aceptación o rechazo, permitiendo clasificar las entradas según criterios formales establecidos. En nuestro caso, la máquina del lenguaje $\mathrm{L} = \{0^{\mathrm{n}}\mathrm{n}\ 1^{\mathrm{n}}\!: \mathrm{n} > 0\}$, evalúa la estructura y composición de cada cadena, aceptando únicamente aquellas que cumplen con la condición de igualdad entre ceros y unos.

- Máquinas computadoras (Ejercicio 2): Su función es transformar la entrada aplicando un procedimiento algorítmico concreto, generando un resultado calculado sobre la cinta. A diferencia de las máquinas reconocedoras, estas no rechazan entradas válidas, sino que ejecutan una operación y producen una salida. La máquina del sucesor binario ejemplifica este comportamiento: recibe un número binario y calcula su consecutivo mediante suma aritmética con propagación de acarreo.

Esta diferenciación conceptual es esencial en teoría de la computación, ya que demuestra la versatilidad de las Máquinas de Turing no solo como reconocedoras de lenguajes formales, sino también como ejecutoras de procedimientos computacionales generales. Ambos tipos de máquinas comparten la misma estructura formal, pero difieren fundamentalmente en su propósito: decidir membresía versus computar resultados.

Por otra parte, el segundo ejercicio evidenció que las Máquinas de Turing no solo reconocen lenguajes, sino que también pueden realizar cálculos aritméticos básicos. La implementación del algoritmo de suma binaria con propagación de acarreo permitió calcular correctamente el sucesor de distintos números binarios. Este ejercicio refuerza la idea de que la Máquina de Turing constituye un modelo computacional universal, capaz de simular cualquier procedimiento algorítmico.

Asimismo, el uso de JFLAP como herramienta de simulación resultó fundamental para visualizar el comportamiento dinámico de la máquina, observar la evolución de la cinta y verificar configuraciones de aceptación o rechazo. La experimentación con múltiples cadenas permitió validar rigurosamente cada diseño y fortalecer la comprensión del proceso interno de cómputo.

En síntesis, esta actividad no solo permitió aplicar conceptos fundamentales de teoría de la computación, sino que también consolidó la importancia de la precisión en la definición de estados, transiciones y símbolos de cinta. Más allá del resultado final, el laboratorio evidenció cómo el diseño estructurado y el análisis cuidadoso son esenciales para garantizar el correcto funcionamiento de un sistema formal de cómputo.

BIBLIOGRAFÍA

A continuación, la bibliografía implementada en este desarrollo:

- Tema 1. Conceptos matemáticos utilizados. Informática Teórica (COLGII) - PER 15746 - Enero 2026.
- Tema 2. Lenguajes y gramáticas formales. Informática Teórica (COLGII) - PER 15746 - Enero 2026.
- Tema 3. Introducción a las máquinas de Turing. Informática Teórica (COLGII) - PER 15746 - Enero 2026.
- Tema 4. Extensiones para las máquinas de Turing. Informática Teórica (COLGII) - PER 15746 - Enero 2026.
- Tema 5. Máquinas de Turing restringidas. Computadoras. Informática Teórica (COLGII) - PER 15746 - Enero 2026.
- Clases virtuales con el profesor Ing. Rogerio Orlando Beltrán Castro.
- Sipser, M. (2013). Introduction to the theory of computation (3rd ed.). Cengage Learning.
- Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2007). Introduction to automata theory, languages, and computation (3rd ed.). Pearson.
- Lewis, H. R., & Papadimitriou, C. H. (1998). Elements of the theory of computation (2nd ed.). Prentice Hall.
- Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, 42(2), 230–265. https://doi.org/10.1112/plms/s2-42.1.230
- Rodger, S. H., & Finley, T. W. (2006). JFLAP: An interactive formal languages and automata package. Jones & Bartlett Learning.
- Rodger, S. H. (s.f.). JFLAP – Java Formal Languages and Automata Package. Duke University. http://www.jflap.org

AGRADECIMIENTO

Finalmente, deseo expresar mi más sincero agradecimiento al profesor Ing. Rogerio Orlando Beltrán Castro, por los conocimientos, orientación y acompañamiento brindados durante el desarrollo de esta actividad. Sus explicaciones y aportes en el área de teoría de la computación y lenguajes formales

fueron fundamentales para comprender el funcionamiento de las Máquinas de Turing y su importancia como modelo universal de cómputo.

Gracias a los conceptos compartidos en clase, fue posible realizar este laboratorio de manera estructurada, aplicando correctamente estrategias de diseño, definición de estados y transiciones, así como la validación mediante simulaciones en la herramienta JFLAP. Sin duda, esta experiencia fortaleció significativamente mi formación académica y mi comprensión de los fundamentos que sustentan la computabilidad y el estudio formal de los lenguajes.

¡¡¡Mil gracias profesor!!!

Respetuosamente,

Alejandro De Mendoza