UNIVERSIDAD DE LA LAGUNA

GRADO EN INGENIERÍA INFORMÁTICA

Computabilidad y Algoritmia

PRÁCTICA 9. Diseño y simulación de máquinas de Turing en JFLAP

Presentado por:

Aarón Ramírez Valencia
alu0101438238@ull.edu.es
07 / 11 / 2024

Recordatorio:

Una máquina de Turing se puede definir formalmente como una 7-upla M ≡ (Q, Σ, Γ, q0,ƀ, F, δ) donde el significado de cada uno de los elementos es:

- Q es el conjunto de estados (Q finito y $Q \neq \varnothing$).
- Γ es el alfabeto de la cinta.
- Σ es el alfabeto de entrada (generalmente $\Sigma \subseteq (\Gamma - \{ƀ\})$).
- $q0 \in Q$ es el estado inicial o de arranque.
- $F \subseteq Q$ es el conjunto de estados de aceptación.
- $\$ \in \Gamma$ es el símbolo blanco ($\$ \notin \Sigma$).
- δ es la función de transición:

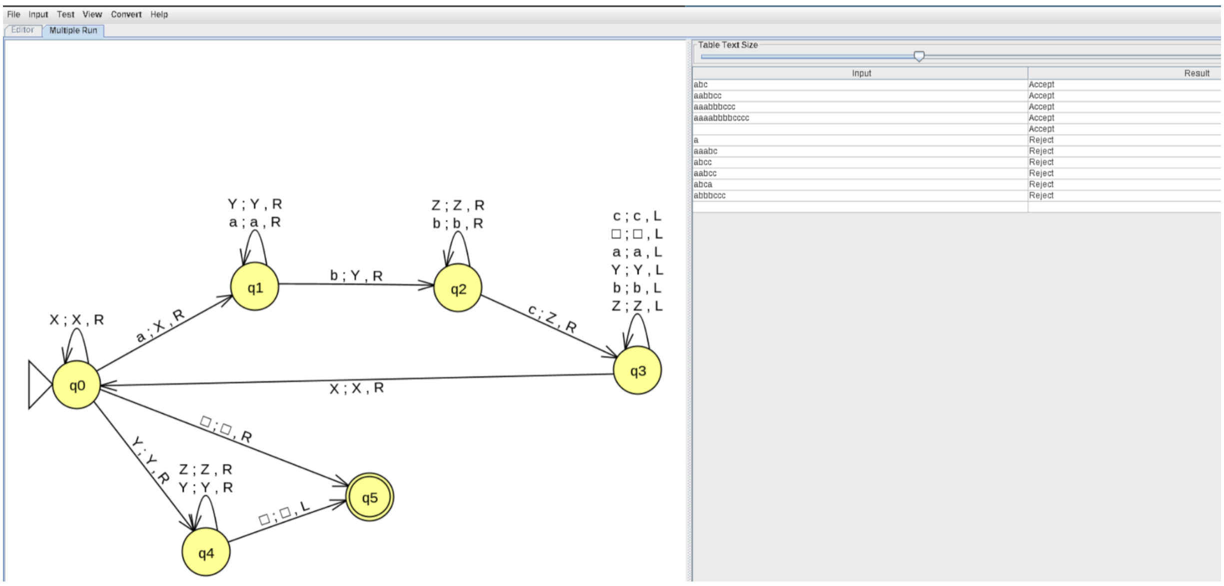
  $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$

  $(q, a) \rightarrow (p, t, X)$ con $p, q \in Q$; $a, t \in \Sigma$; $X \in \{L, R\}$

Ejercicios

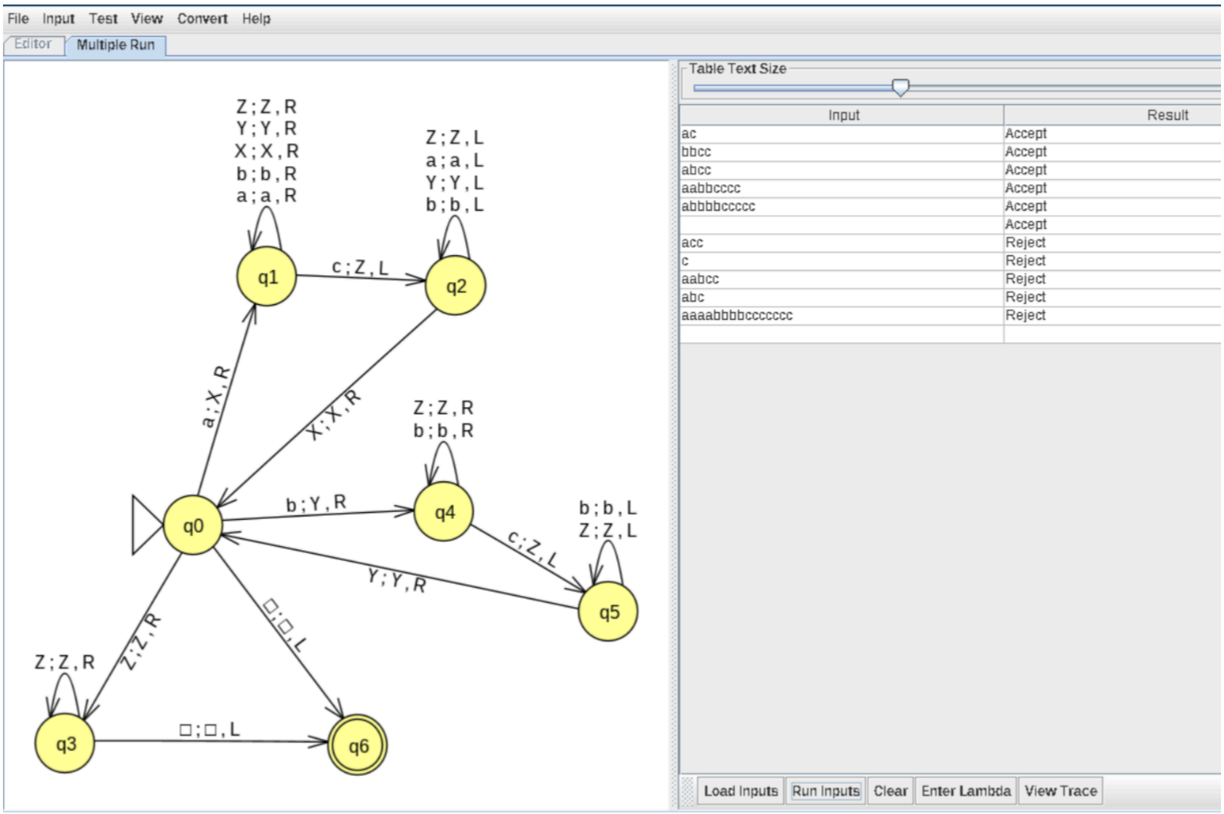
1. Diseñar y simular en JFLAP máquinas de Turing que acepte el lenguaje $L = \{a^n b^n c^n \mid n \geq 0\}$.

Explicación: Por cada a que encontremos la sustituiremos por una X, cada b por una Y y cada c por una Z. Una vez hayamos convertido todas las cadenas se debería llegar al blanco y en caso contrario no es válida. La cadena vacía es aceptada.



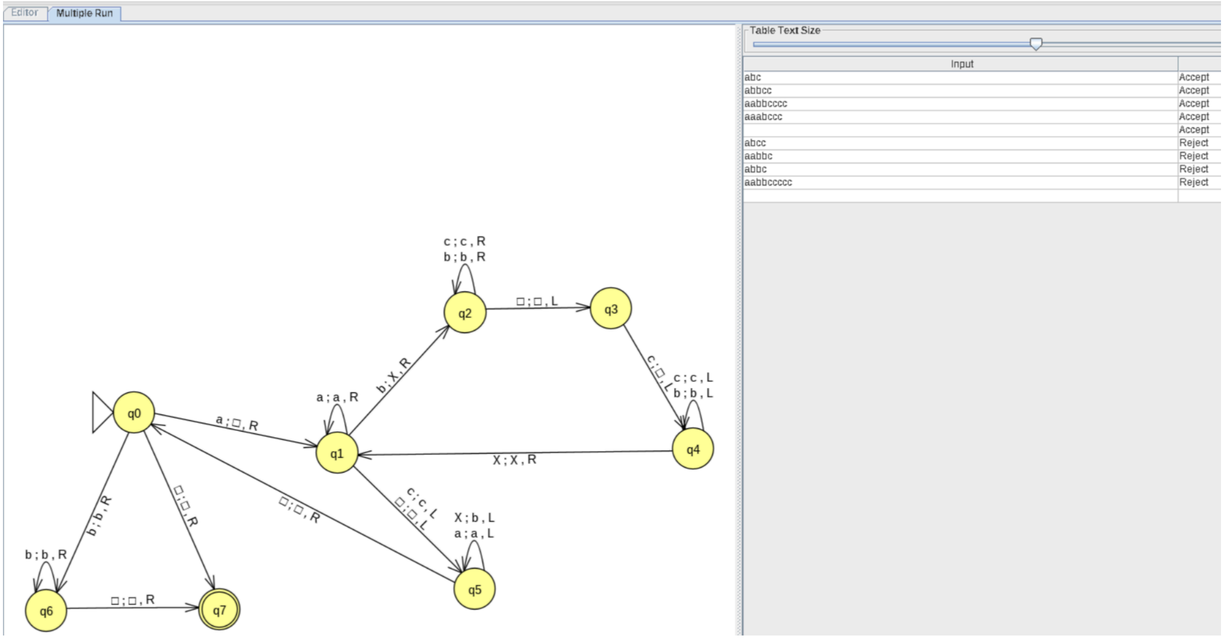
2. Diseñar y simular en JFLAP máquinas de Turing que acepte el lenguaje $L = \{a^n b^m c^{n+m} \mid n \geq 0, m \geq 0\}$.

Explicación: **Fase 1:** Marcar las ‘a’ sustituyéndolas por una X, luego buscar la primera ‘c’ y sustituirla por una Z. Repetir este proceso hasta que no queden más ‘a’ sin marcar.
Fase 2: Después de marcar todas las ‘a’, realizar el mismo proceso para las ‘b’. Marcar las ‘b’ sustituyéndolas por una Y, luego buscar la primera ‘c’ y sustituirla por una Z. Repetir este proceso hasta que no queden más ‘b’ sin marcar.
Fase 3: Verificar que no haya símbolos sin marcar. Después de emparejar todas las ‘a’y las ‘b’ con los ‘c’, verificar que no haya símbolos ‘c’ sin marcar (todos deberían ser ‘Y’).
La cadena vacía es aceptada.



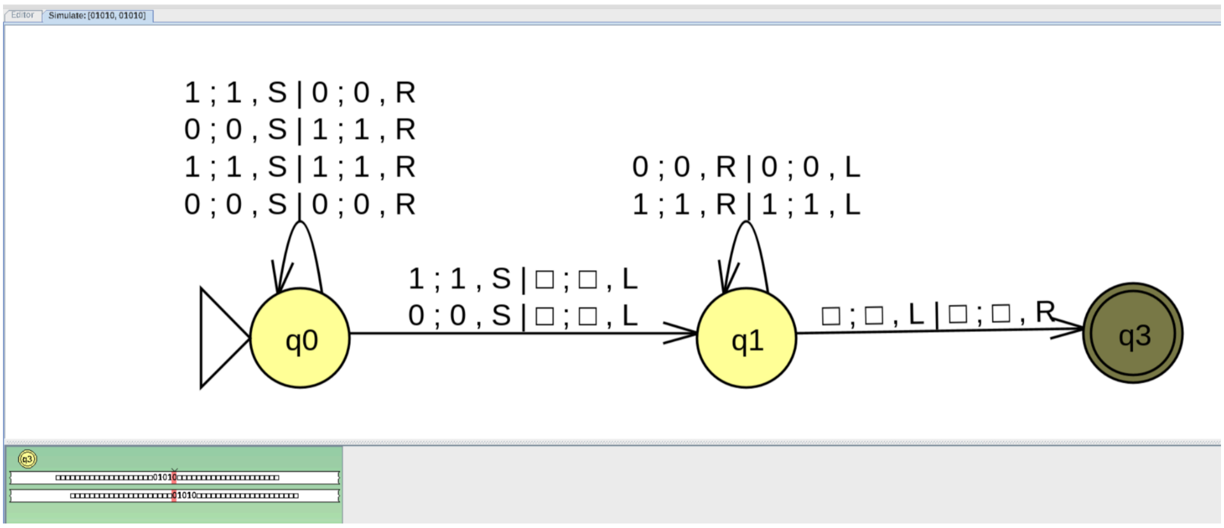
3. Diseñar y simular en JFLAP máquinas de Turing que acepte el lenguaje L = $\{a^n b^m c^{n*m} \mid n \geq 1, m \geq 1\}$.
Explicación:
Paso 1: Reemplaza la primera ‘a’ por un blanco, luego encuentra y reemplaza el primer ‘b’ por ‘X’, y la última ‘c’ por un blanco. Repite esto en el estado ‘q1’ hasta que ya no queden ‘b’.
Paso 2: Una vez que todas las ‘b’ han sido reemplazadas por ‘X’, cambia la última ‘c’ a blanco y regresa al inicio. Reemplaza todas las ‘X’ con ‘b’, transformando la cadena en una forma más corta.
Repetición: Se repite los pasos anteriores hasta que todas las ‘a’ hayan sido convertidas en blancos. Si se encuentran blancos en el estado ‘q1’ y el proceso de marcado de ‘c’ está completo, verifica que todas las ‘c’ se han reemplazado correctamente.
Verificación Final: Al final, solo deben quedar ‘b’ y espacios en blanco en los estados ‘Q0’ y ‘Q6’. Si no quedan ‘c’ al llegar al final, la cadena cumple con el lenguaje.

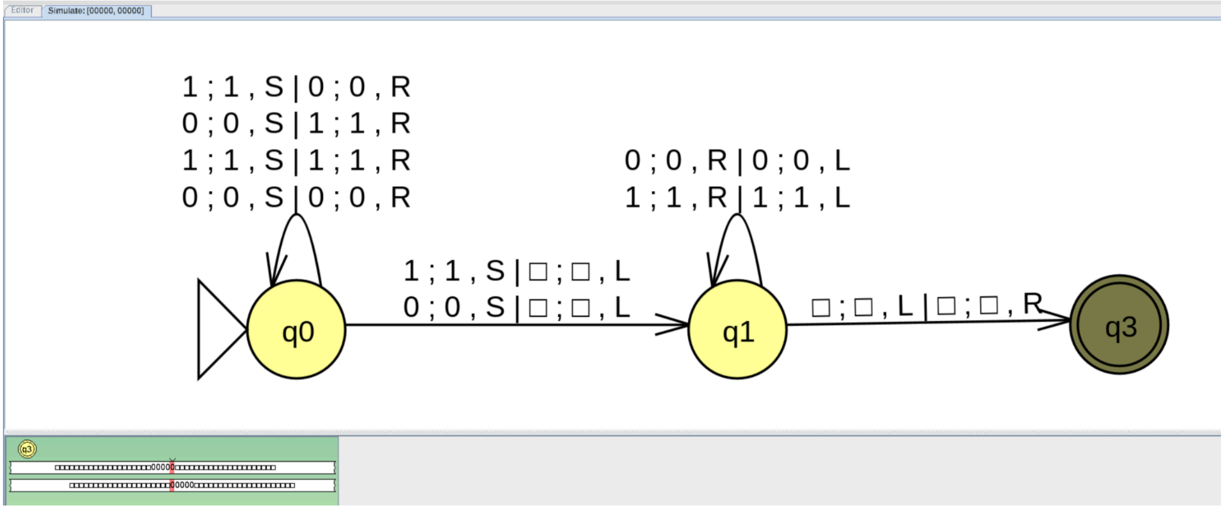
Aceptación: Si se alcanza un símbolo en blanco en ‘Q6’, la cadena es aceptada en el estado final ‘Q7’. También se acepta si la entrada estaba vacía inicialmente.

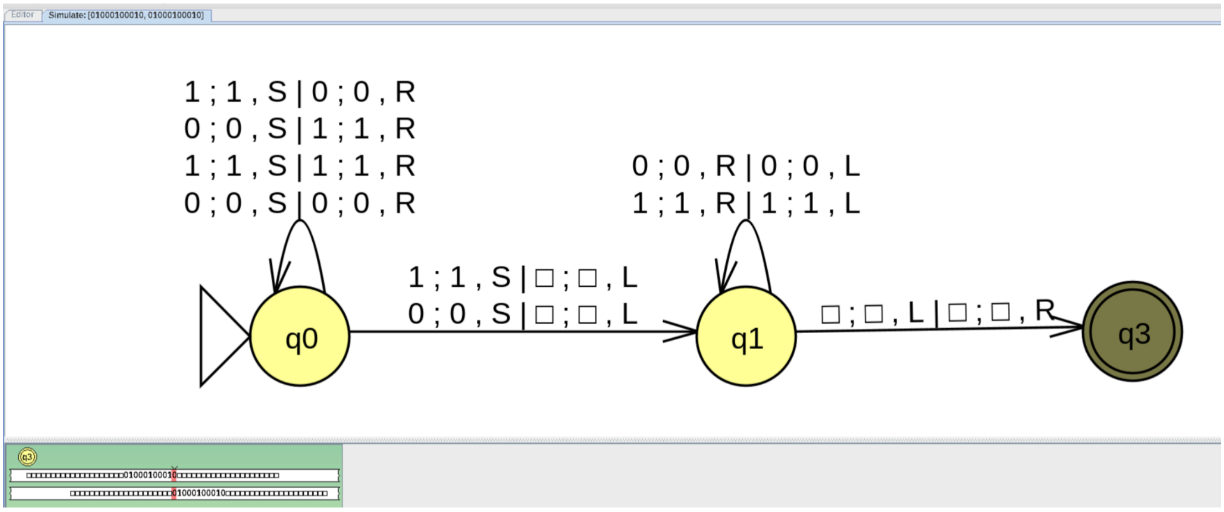


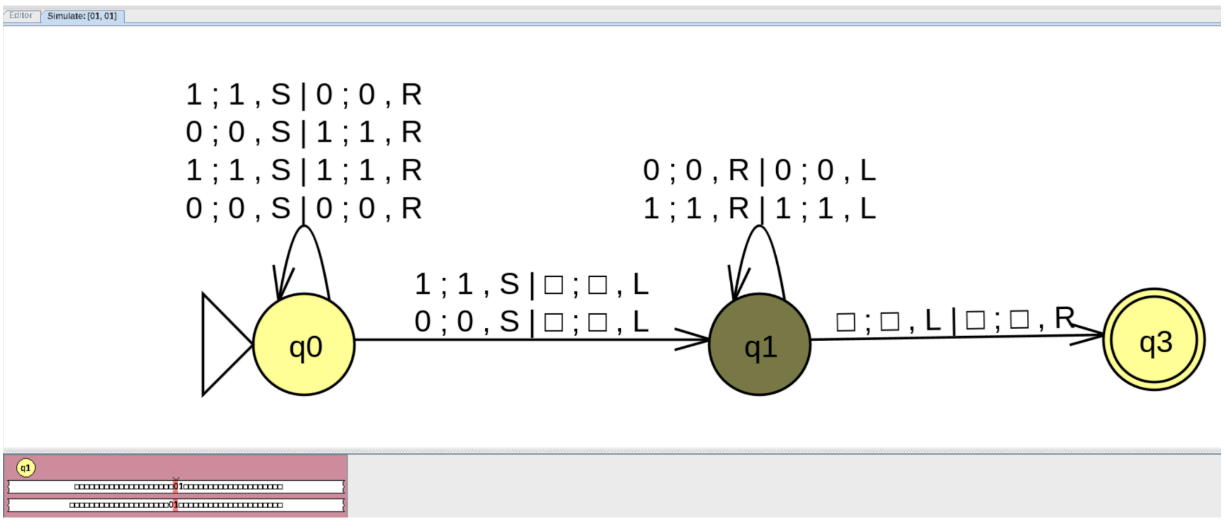
4. Diseñar y simular en JFLAP máquinas de Turing que acepte el lenguaje L = {w | w = w^{-1}} sobre el alfabeto Σ = {0, 1}.

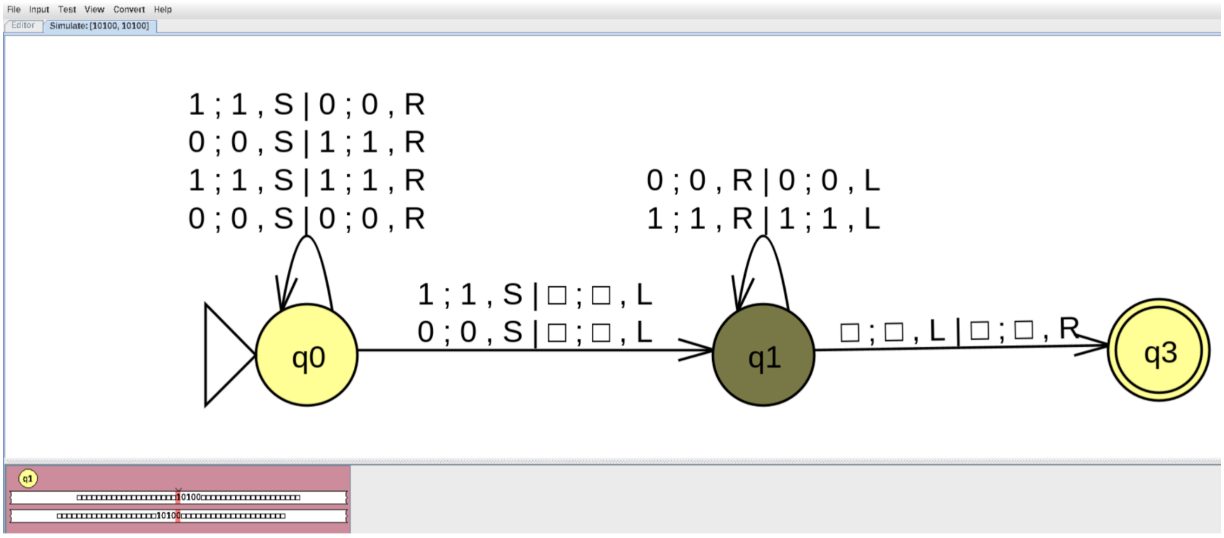
Explicación: Simplemente son 2 cintas y ambas tendrán la cadena a probar y para saber si es palindroma simplemente movemos el cabezal de la segunda cinta hasta el final de la cadena y a partir de ahí vamos mirando carácter por carácter si es el mismo. Si llega hasta el final, w = w^{-1}.

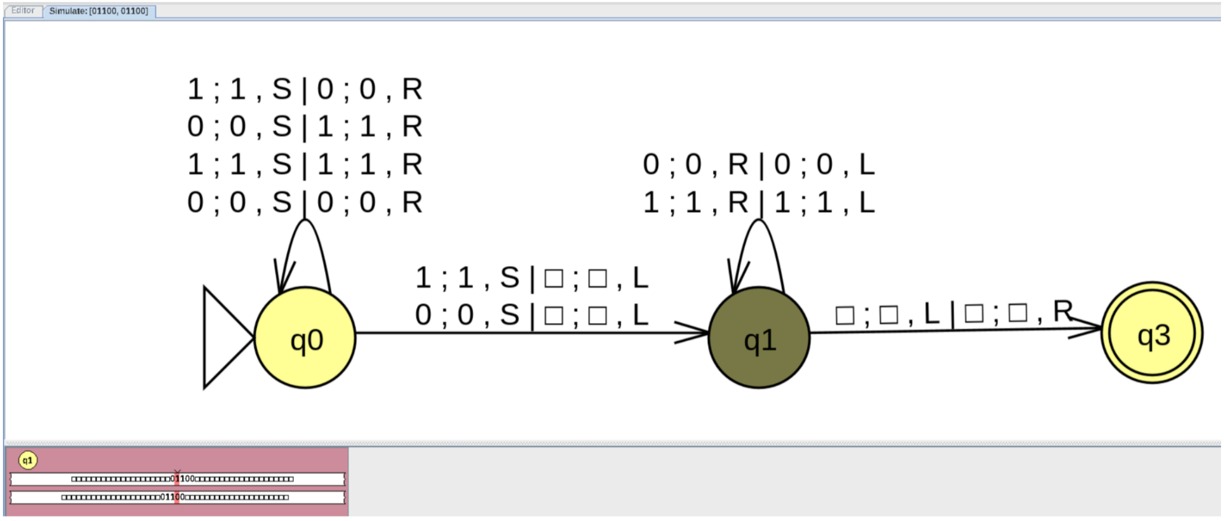



Simulate: [00000, 00000]
1 ; 1 , S | 0 ; 0 , R
0 ; 0 , S | 1 ; 1 , R
1 ; 1 , S | 1 ; 1 , R
0 ; 0 , S | 0 ; 0 , R
0 ; 0 , R | 0 ; 0 , L
1 ; 1 , R | 1 ; 1 , L
1 ; 1 , S | □ ; □ , L
0 ; 0 , S | □ ; □ , L
□ ; □ , L | □ ; □ , R
q0
q1
q3


Simulate: [01000100010, 01000100010]
1 ; 1 , S | 0 ; 0 , R
0 ; 0 , S | 1 ; 1 , R
1 ; 1 , S | 1 ; 1 , R
0 ; 0 , S | 0 ; 0 , R
0 ; 0 , R | 0 ; 0 , L
1 ; 1 , R | 1 ; 1 , L
1 ; 1 , S | □ ; □ , L
0 ; 0 , S | □ ; □ , L
□ ; □ , L | □ ; □ , R
q0
q1
q3

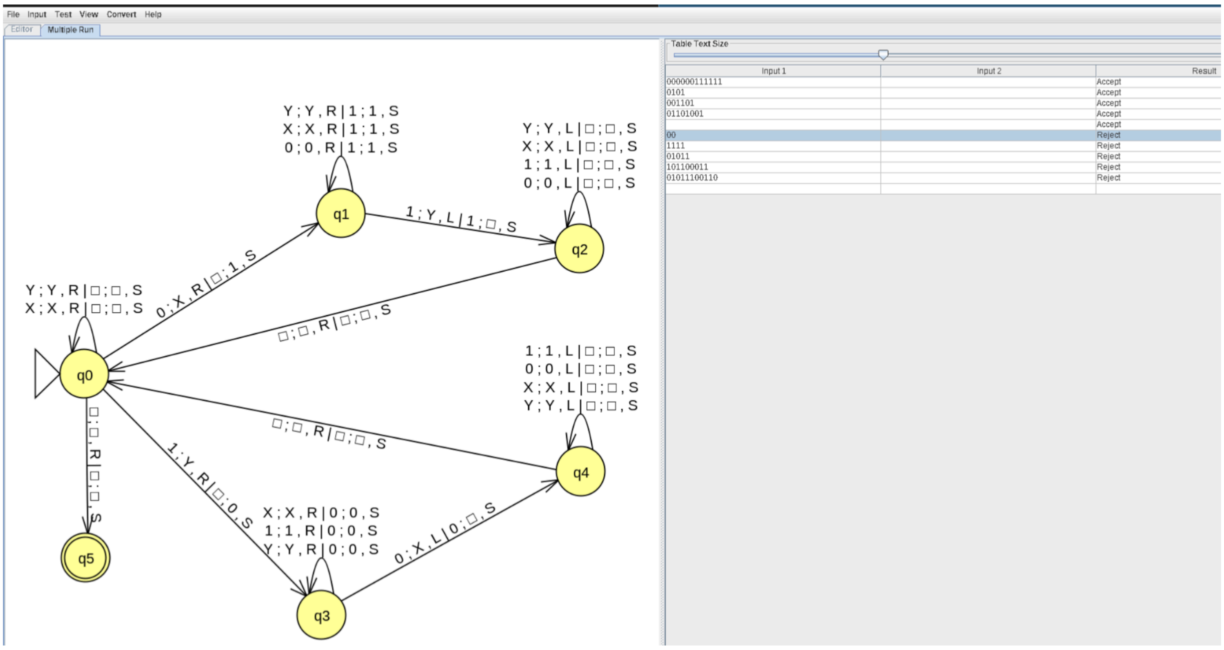

Simulate: [01, 01]
1 ; 1 , S | 0 ; 0 , R
0 ; 0 , S | 1 ; 1 , R
1 ; 1 , S | 1 ; 1 , R
0 ; 0 , S | 0 ; 0 , R
0 ; 0 , R | 0 ; 0 , L
1 ; 1 , R | 1 ; 1 , L
1 ; 1 , S | □ ; □ , L
0 ; 0 , S | □ ; □ , L
□ ; □ , L | □ ; □ , R
q0
q1
q3





5. Diseñar y simular en JFLAP máquinas de Turing que acepte el lenguaje formado por todas las cadenas binarias que tienen igual número de ceros que de unos (en cualquier orden de aparición).

Explicación: La máquina de Turing recorre la cadena marcando cada '0' encontrado y luego busca el primer '1' sin marcar para emparejarlos; después, repite el proceso alternando, marcando primero un '1' y buscando un '0' sin marcar. Si logra emparejar todos los '0' y '1' sin dejar símbolos sueltos, acepta la cadena; si encuentra un símbolo sin su par, la rechaza.



Modificación

L = { $a^n b^{2n} c^n$} n>= 0

Explicación:

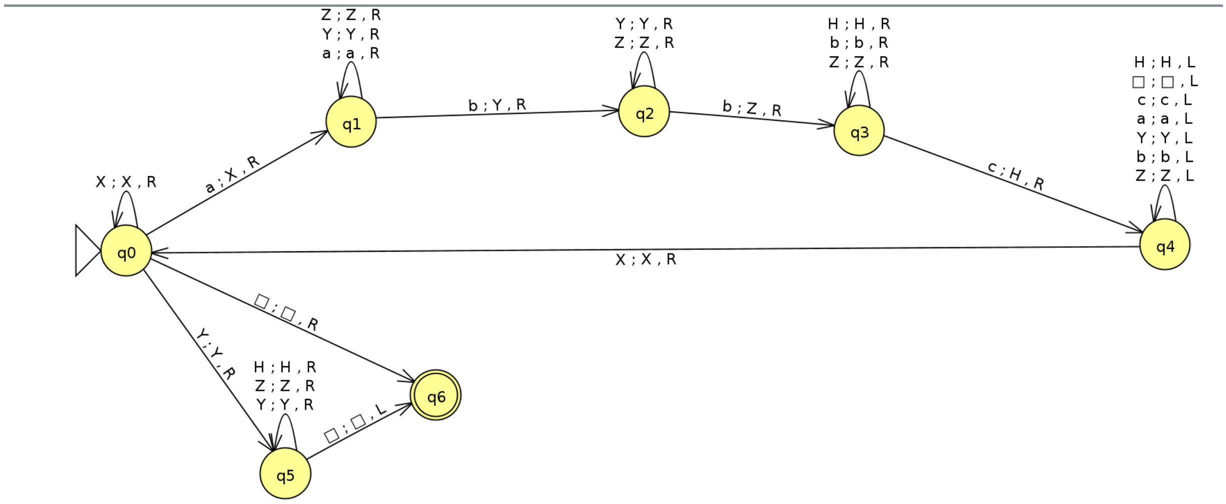
Esta máquina de turing está basada en la MT del ejercicio 1. *(Por cada a que encontremos la sustituiremos por una X, cada b por una Y y cada c por una Z. Una vez hayamos convertido todas las cadenas se debería llegar al blanco y en caso contrario no es válida. La cadena vacía es aceptada.)* Simplemente se añade un nuevo estado que cogerá la b añadida y se sustituirá por H, así quedando cada conjunto de subcadenas que se van a ir aceptando con la forma XYZH, siendo la sustitución:

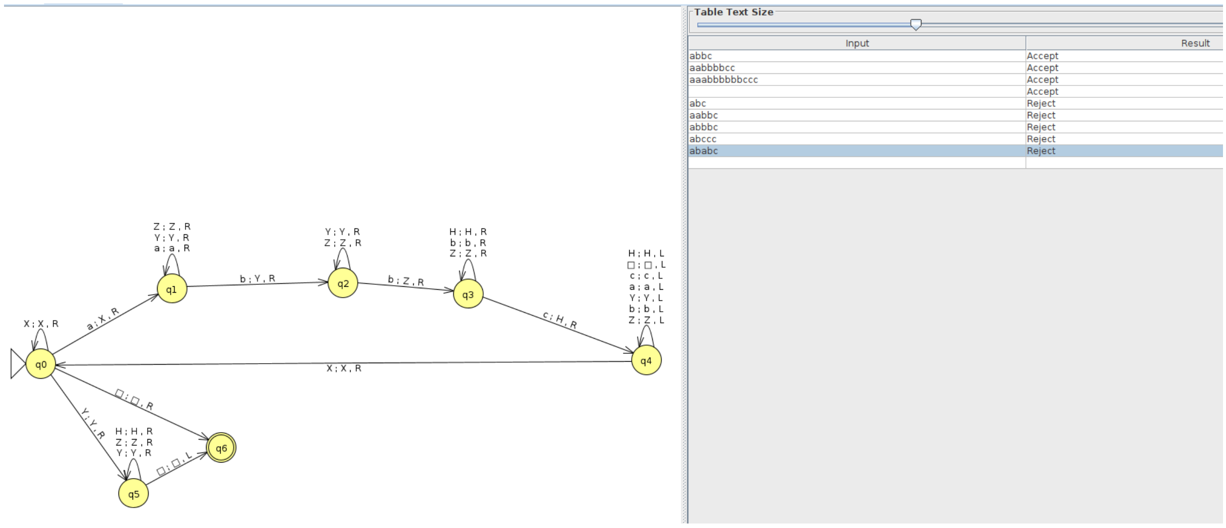
- a →X

- 1ª b →Y

- 2ª b →Z

- c →H





Input	Result
abbc	Accept
aabbbbcc	Accept
aaabbbbbbccc	Accept
	Accept
abc	Reject
aabbc	Reject
abbbc	Reject
abccc	Reject
ababc	Reject