Universidad de La Laguna

Grado en Ingeniería Informática

Computabilidad y Algoritmia

PRÁCTICA 9

Simulación de máquinas de Turing en JFLAP

Factor de ponderación: 7

1. Objetivos

El objetivo de esta práctica consiste en introducir los fundamentos básicos de las máquinas de Turing [1]. Se comprobará y verificará el funcionamiento de algunos ejemplos y se diseñarán máquinas de Turing que cumplan con un propósito determinado. Para simular el comportamiento de las máquinas de Turing diseñadas utilizaremos la herramienta JFLAP [2]. JFLAP es un paquete de gran interés en el ámbito de la enseñanza de lenguajes formales pues constituye una herramienta interactiva para visualización y simulación de autómatas finitos y máquinas de Turing, entre otros. Puesto que en esta práctica utilizaremos JFLAP para diseñar y simular máquinas de Turing, será necesario descargar e instalar la herramienta siguiendo estos sencillos pasos [3].

Para esta práctica será necesario realizar los ejercicios propuestos en este enunciado y llevarlos resueltos a la clase práctica de laboratorio. Durante la sesión presencial se les podrá proponer la resolución de nuevos ejercicios.

Tal y como venimos insistiendo a lo largo de las prácticas de la asignatura, si el alumnado tiene dudas respecto a cualquiera de estos aspectos, debiera acudir al foro de discusiones de la asignatura para plantearlas allí. Se espera que, a través de ese foro, el alumnado intercambie experiencias y conocimientos, ayudándose mutuamente a resolver dichas dudas. También el profesorado de la asignatura intervendrá en las discusiones que pudieran suscitarse, si fuera necesario.

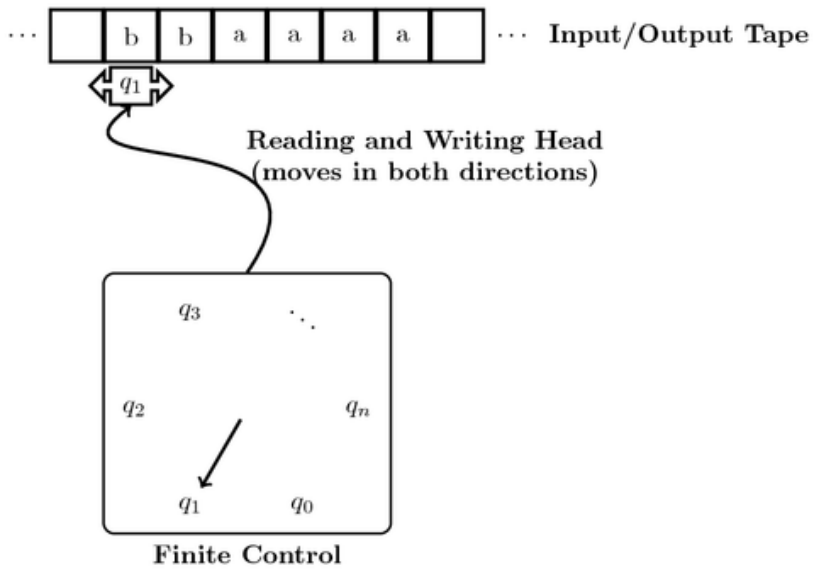

Figura 1: Estructura general de una máquina de Turing

2. Máquinas de Turing

La clase de autómatas que hoy conocemos como máquinas de Turing fue propuesta en 1936 por el matemático Alan Turing. La pretensión de Turing era el estudio de los procesos algorítmicos usando un modelo computacional. Podemos considerar las máquinas de Turing como una generalización de los autómatas que hemos estudiado en temas anteriores de la asignatura. Las máquinas de Turing (TM, por *Turing Machine*) son similares a los autómatas finitos en tanto en cuanto poseen un mecanismo de control y un flujo de entrada que se representa como una cinta (ver Figura 1). La diferencia fundamental es que una máquina de Turing puede mover su cabeza de lectura hacia delante (derecha) y hacia atrás (izquierda) sobre la cinta de entrada, y además, es capaz tanto de leer como de escribir en dicha cinta.

La TM dispone de una cabeza que puede leer y escribir símbolos en la cinta de la máquina. Esta cinta se supone que se extiende indefinidamente hacia derecha e izquierda. Una TM utiliza su cinta como dispositivo de almacenamiento auxiliar y puede no sólo insertar o extraer símbolos sino también rastrear los datos de la cinta y modificar las celdas que desee sin alterar el resto. Es conveniente que la TM utilice marcas especiales para distinguir partes de la cinta. Con este fin, la TM utiliza símbolos que no aparecen en los datos de entrada, es decir, se distingue entre el conjunto finito de símbolos llamado el alfabeto de la máquina, en el que deben estar codificados los datos de entrada iniciales y otro conjunto también finito de símbolos de cinta posiblemente mayor que el anterior y al que se denomina alfabeto de la cinta, y que representa al conjunto de los símbolos que la máquina puede leer y escribir. De este modo los símbolos de cinta de una TM pueden incluir marcas especiales que no sean símbolos del alfabeto de la máquina. El espacio en blanco, o simplemente *blanco*, es uno de los símbolos de esta clase: se supone que está

en cualquier celda de la cinta que no esté ocupada. Por ejemplo, si una TM leyera más a la derecha de los símbolos de entrada de su cinta, encontraría y leería las celdas en blanco que allí encontraría. Es decir, las celdas que inicialmente no contienen símbolos de la entrada contienen un blanco. También puede ser necesario que una TM borre una celda de la cinta, escribiendo en ella un espacio en blanco. Por estos motivos se considera que el blanco forma parte de los símbolos de cinta, pero no del alfabeto de la máquina. Para que el símbolo de espacio en blanco no nos ocasione problemas en las representaciones, es necesario adoptar algún convenio para representar el símbolo. Por ejemplo, y de forma análoga o como hacíamos para la cadena vacía, utilizaremos \$ para la representación de los espacios en blanco. Nótese que JFLAP sigue su propio convenio a la hora de denotar las $\epsilon - transiciones$ en los NFA así como el *blanco* en las TM.

En cada paso de cómputo, una TM lee un símbolo de la cinta de entrada, lo reemplaza por un nuevo símbolo (que pudiera coincidir con el que encontró originalmente) y, a continuación, cambia su estado (el nuevo estado puede ser el mismo en que se encontraba antes de la operación) y realiza un movimiento de la cabeza de lectura/escritura hacia la izquierda o a la derecha. La acción que en un determinado momento del cómputo realizará la TM dependerá del símbolo (símbolo actual) que esté "visible" en ese momento en la celda (celda actual) en que está posicionada la cabeza de lectura/escritura, y del estado actual del mecanismo de control de la máquina.

Una máquina de Turing se puede definir formalmente como una 7-upla $M \equiv (Q, \Sigma, \Gamma, q_0, \text{ƀ}, F, \delta)$ donde el significado de cada uno de los elementos es:

- Q es el conjunto de estados (Q finito y $Q \neq \emptyset$)
- Γ es el alfabeto de la cinta
- Σ es el alfabeto de entrada (generalmente $\Sigma \subseteq (\Gamma - \{\text{ƀ}\})$)
- $q_0 \in Q$ 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:
  $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$
  $(q,\ a) \rightarrow (p,\ t,\ X) \quad \text{con } p, q \in Q; \quad a, t \in \Sigma; \quad X \in \{L, R\}$

Tal y como ya se ha mencionado, el alfabeto de entrada generalmente será un subconjunto del de la cinta, pero el símbolo blanco no puede formar parte del alfabeto de entrada. Por su parte, la función de transición transforma pares (q, a), formados por el estado actual q y el símbolo a que se lee en la posición donde se encuentra la cabeza de lectura/escritura, en ternas (p, t, X) donde p es el estado siguiente, t es el símbolo que se escribe en la cinta y X es un movimiento de la cabeza de lectura/escritura que puede ser a la izquierda (L) o a la derecha (R). Por último, cabe destacar que la función de transición δ de una TM es una función parcial, lo cual quiere decir que no tiene porqué estar definida para todos los pares (q, a) del conjunto de partida.

3. Diseño de máquinas de Turing

Consideremos una máquina de Turing que acepte el lenguaje siguiente:

$$L = \{0^n 1^n \mid n \geq 1\}$$

Inicialmente la máquina contendrá una secuencia de símbolos "0" seguida de una secuencia de símbolos "1" seguida por un número infinito de símbolos \$ (utilizaremos el símbolo \$ para representar el símbolo blanco). La máquina tendrá que comprobar que el número de ceros a la izquierda de la secuencia tendrá que coincidir exactamente con el número de unos a la derecha. Para diseñar, y posteriormente simular, esta máquina de Turing utilizaremos JFLAP [2, 3]. A modo introductorio, sería recomendable visualizar los videotutoriales de la asignatura en los que se explica cómo utilizar JFLAP para diseñar y simular DFAs [5] y NFAs [6].

Para diseñar una máquina de Turing utilizaremos la opción `Turing machine` del menú principal de JFLAP. En la ventana de diseño crearemos la máquina de Turing que se muestra en la Figura 2. La idea central del diseño realizado es que por cada cero encontrado a la izquierda de la secuencia debemos encontrar un uno a la derecha. Para realizar este "emparejamiento" la máquina reemplazará el "0" más a la izquierda en la cinta por un símbolo X, a continuación se moverá hacia la derecha hasta el "1" situado más a la izquierda, reemplazándolo, en este caso, por un símbolo Y. A continuación, se moverá hacia la izquierda hasta encontrar la X situada más a la derecha. En ese momento se moverá una celda a la derecha para alcanzar el "0" situado más a la izquierda. A partir de ese momento, se volverá a repetir el ciclo. En este proceso se supone que cuando recorremos ceros sólo tendremos ceros y que cuando se encuentra el primer uno ya no tendremos ningún cero. Si la máquina encuentra un \$ cuando busca un "1", la máquina parará sin aceptar la cadena. Si después de cambiar un "1" por una Y la máquina no encuentra más ceros, entonces habrá que comprobar que tampoco hay más unos (que hayan quedado sin emparejar); aceptando la cadena si no los hay.

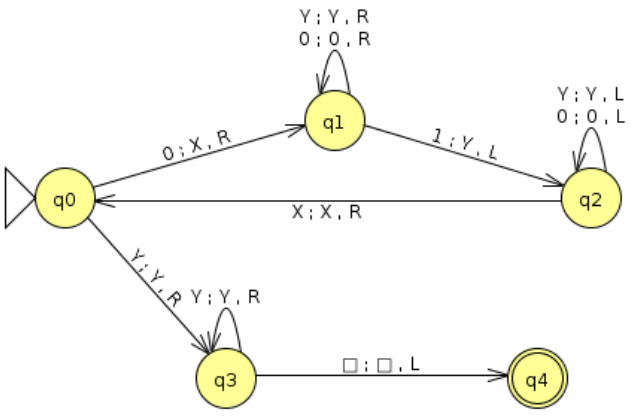

Figura 2: Máquina de Turing para $L = \{0^n 1^n \mid n \geq 1\}$

El alfabeto de entrada es $\Sigma = \{0, 1\}$, el alfabeto de la cinta es $\Gamma = \{0, 1, X, Y, \$\}$ y el símbolo blanco se ha representado por \$ aunque en JFLAP se representa con un símbolo $\square$. El conjunto de estados de la máquina será $Q = \{q_0, q_1, q_2, q_3, q_4\}$. El estado q_0 es el inicial y tenemos a q_4 como único estado de aceptación. La función de transición para esta máquina sería la siguiente:

δ	0	1	X	Y	\$
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)	$(q_4, \$, L)$
q_4	-	-	-	-	-

El funcionamiento general de la máquina de Turing diseñada se describe a continuación:

- Desde el estado q_0 se reemplazará por X un cero situado más a la izquierda.
- El estado q_1 se utiliza para buscar hacia la derecha ignorando ceros y símbolos Y, hasta encontrar el 1 situado más a la izquierda. Si la máquina halla un "1", lo cambia por Y pasando al estado q_2. Cuando la máquina busca hacia la derecha en el estado q_1 rechaza la entrada si encuentra un \$ o una X antes que un "1": en este caso o bien hay demasiados ceros o bien la entrada no es de la forma 0^+1^+.
- El estado q_2 se usa para buscar un símbolo X hacia la izquierda. Cuando lo encuentra, la máquina pasa al estado q_0 y se mueve a la derecha para colocarse sobre el "0" situado más a la izquierda.
- El estado q_0 juega también otro papel: si después de que en el estado q_2 se encuentre la X situada más a la derecha hay una Y situada inmediatamente a la derecha, entonces es que los ceros se han acabado. En este caso (estando en q_0 y viendo una Y) se transita al estado q_3.
- En el estado q_3 se leen todas las Y para comprobar que no queda ningún "1". Si los símbolos Y están seguidos por un \$ se pasa al estado de aceptación q_4. En otro caso, la cadena es rechazada pues significaría que el número de ceros no ha coincidido con la secuencia de unos que debería seguirle a continuación.

Veamos un ejemplo de computación con esta máquina cuando la cadena de entrada es 0011:
$(q_0, \underline{0}011\$) \vdash (q_1, X\underline{0}11\$) \vdash (q_1, X0\underline{1}1\$) \vdash (q_2, X\underline{0}Y1\$) \vdash (q_2, \underline{X}0Y1\$) \vdash (q_0, X\underline{0}Y1\$) \vdash (q_1, XX\underline{Y}1\$) \vdash (q_1, XXY\underline{1}\$) \vdash (q_2, XX\underline{Y}Y\$) \vdash (q_2, X\underline{X}YY\$) \vdash (q_0, XX\underline{Y}Y\$) \vdash (q_3, XXY\underline{Y}\$) \vdash (q_3, XXYY\underline{\$}) \vdash (q_4, XXY\underline{Y}\$)$

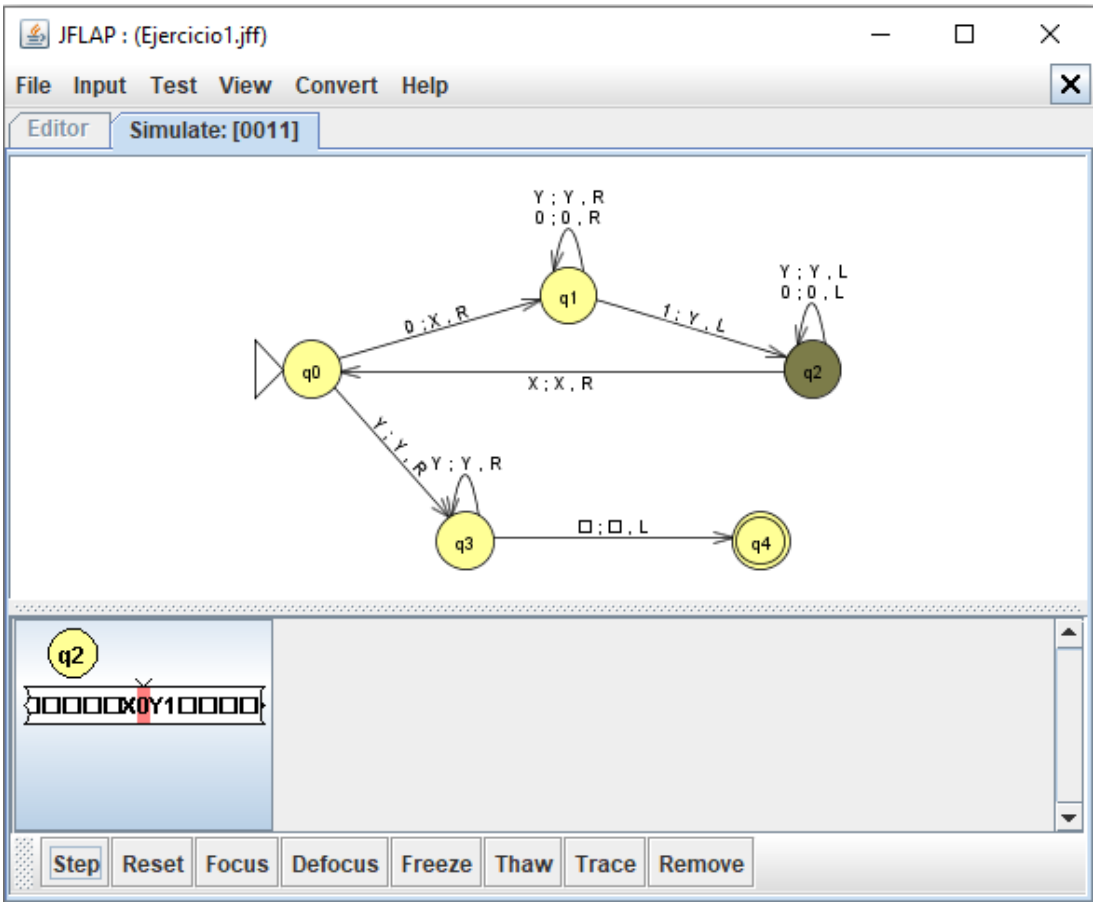

Figura 3: Funcionamiento para la cadena 0011

Para simular el comportamiento de la máquina diseñada utilizaremos la interfaz de JFLAP. Para ello, desde la ventana en la que hemos diseñado nuestra máquina de Turing utilizaremos la opción `Step` del menú `Input`. Esto nos permitirá simular el comportamiento de la máquina, paso a paso, con una determinada cadena de entrada. Por ejemplo, si fijamos como entrada la cadena 0011 podremos ir viendo los pasos de computación uno a uno, hasta que tras una primera pasada, la máquina cambiará el "0" de más a la izquierda por una X y el "1" de más a la izquierda por una Y, tal y como puede apreciarse en la Figura 3. Esta simulación nos permite visualizar de una forma clara e interactiva el comportamiento de la máquina en cada caso, ayudando a entender el funcionamiento interno de las máquinas de Turing.

Por otro lado, si quisiéramos simular la máquina de Turing con un conjunto de diferentes cadenas de entrada, podríamos utilizar la opción `Multiple Run` disponible también en el menú `Input`. Esta opción nos proporciona una interfaz en la que podremos definir un listado de diferentes entradas. Una vez definidas las cadenas de entrada podremos utilizar la opción `Run Inputs` para que se simule el comportamiento de la MT con cada una de las cadenas de entrada. Para cada cadena se especificará en la columna `Result` un *Accept* o un *Reject* en función de si la máquina ha aceptado o rechazado la entrada en cuestión. A modo de ejemplo de esta opción de ejecuciones múltiples se incluye la Figura 4.

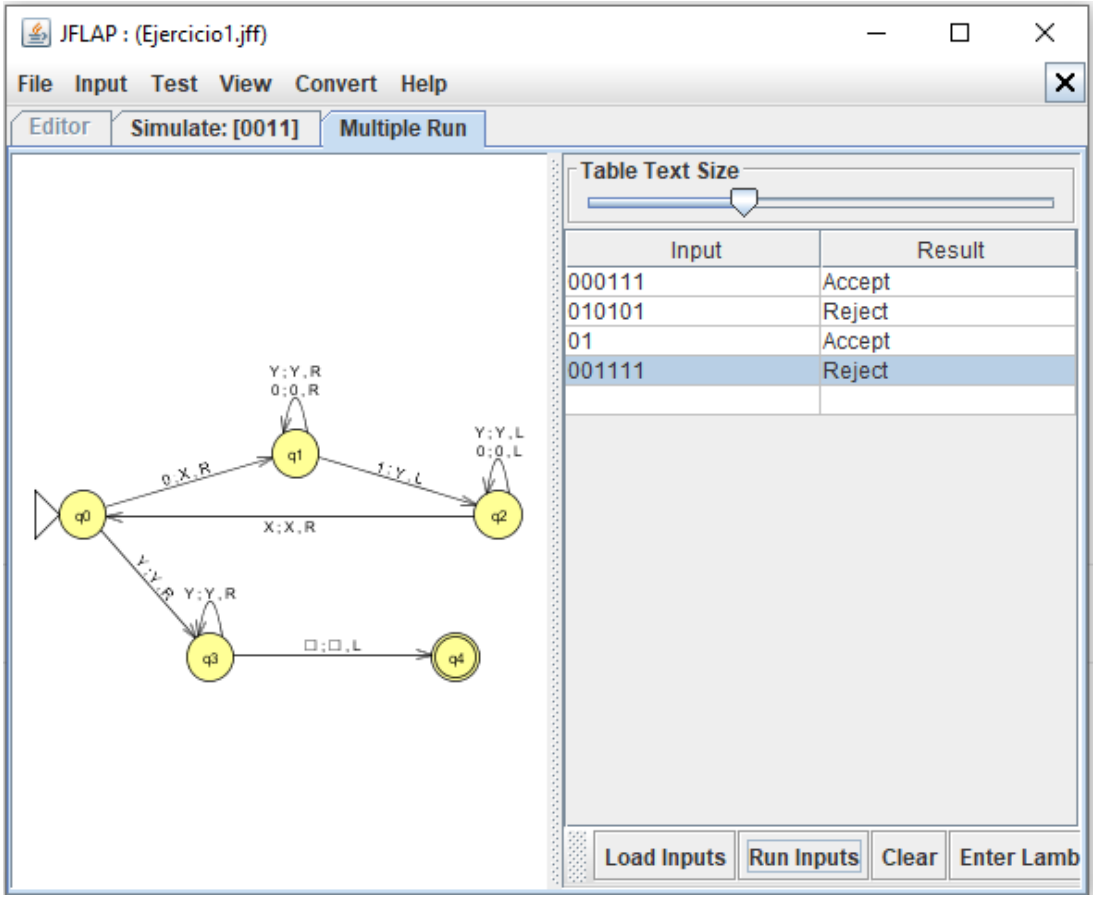

Figura 4: Funcionamiento para múltiples cadenas

Tal y como habremos podido comprobar, cada "emparejamiento" de un cero con un uno implica marcar el cero, desplazarse hacia la derecha para marcar el uno y regresar de nuevo hacia la izquierda para ir en busca del siguiente cero. Este diseño implica, por tanto, realizar múltiples recorridos sobre la cadena de entrada. Sin embargo, vamos a ver cómo podríamos utilizar una máquina de Turing multi-cinta para hacer la misma comprobación pero haciendo únicamente un recorrido sobre la cadena de entrada. Antes de continuar, se recomienda visualizar el videotutorial de la asignatura sobre diseño y simulación de máquinas de Turing en JFLAP [7]. Téngase en cuenta que al definir transiciones para una MT de dos cintas, en JFLAP nos aparecerá una tabla con dos filas de tal forma que en la primera fila definiremos el símbolo que se lee en la primera cinta, el símbolo que se escribe en esa primera cinta y, finalmente, el movimiento que se hará en dicha cinta. En la segunda fila de la tabla se indicará, por su parte, lo que se leerá, escribirá y el movimiento que se realizará en la segunda de las cintas. Este formato de especificación de transiciones puede apreciarse en la Figura 5.

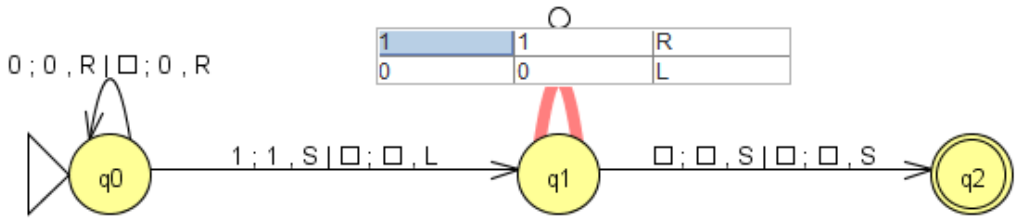

Figura 5: Definición de transiciones en JFLAP para una MT de dos cintas

En la Figura 5 se presenta una MT de dos cintas para aceptar las cadenas de $L = \{0^n1^n \mid n \geq 1\}$. Suponemos que en la primera de las cintas de la MT tendremos la cadena de entrada y que, inicialmente, la segunda cinta de la MT estará vacía. De esta forma, la idea general de esta máquina es que al recorrer la secuencia de ceros a la izquierda de la cadena se utilizará la segunda cinta para realizar una copia de estos ceros. Cuando se encuentra el primer uno de la secuencia la máquina hará un *stop* en la cinta 1 pero se moverá a la izquierda en la cinta 2 para empezar a realizar "emparejamientos" de unos de la primera cinta con ceros en la segunda cinta. Si los ceros copiados en la segunda cinta se acaban al mismo tiempo que los unos en la primera cinta, significará que la cadena de entrada tenía el mismo número de ceros que de unos. En la Figura 6 se muestra el comportamiento de la MT para múltiples cadenas de entrada.

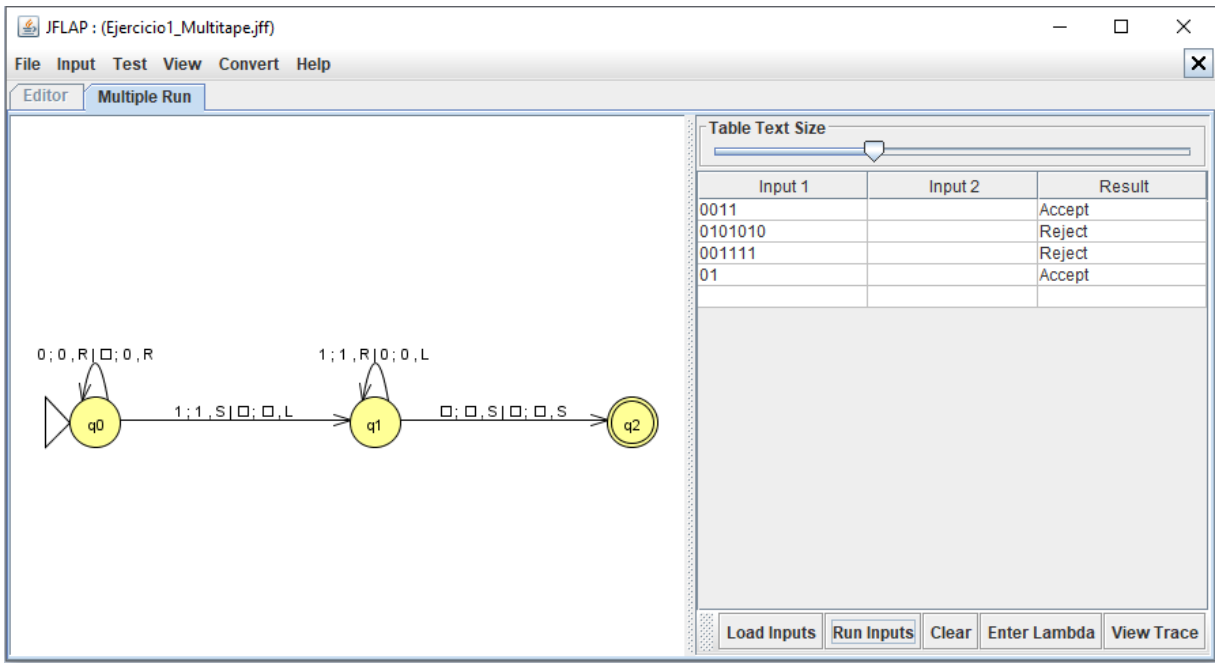

Figura 6: Simulación de MT multi-cinta para $L = \{0^n1^n \mid n \geq 1\}$

4. Ejercicios

Con el objetivo de profundizar en el diseño y la simulación de máquinas de Turing se proponen los siguientes ejercicios. La idea es que se realice, para cada ejercicio, un diseño de máquina de Turing (de una cinta y/o de múltiples cintas) en JFLAP y se realicen las simulaciones oportunas para comprobar su correcto funcionamiento. Para verificar el comportamiento de las máquinas de Turing diseñadas, se debería comprobar el comportamiento de las mismas para al menos una cadena que sea aceptada y para una que no sea aceptada, aunque lo ideal sería utilizar en cada caso una relación más exhaustiva de cadenas de prueba.

1. Diseñar y simular en JFLAP una máquina de Turing que acepte el lenguaje $L = \{a^n b^{2n} \mid n \geq 0\}$.
2. Diseñar y simular en JFLAP una máquina de Turing que acepte el lenguaje $L = \{w \mid$ la longitud de w es par$\}$.
3. Diseñar y simular en JFLAP una máquina de Turing que acepte el lenguaje $L = \{w \mid w = w^{-1}\}$ sobre el alfabeto $\Sigma = \{a, b\}$.
4. Diseñar y simular en JFLAP una máquina de Turing que enumere sobre su cinta todos los números enteros en binario, en orden numérico ascendente cuando comience con la configuración $(q_0, \text{ƀ0ƀ})$. Es decir, la máquina se ejecutaría de la forma siguiente (obsérvese que la máquina nunca para):

$$(q_0, \text{ƀ0ƀ}) \vdash (q_1, \text{ƀ1ƀ}) \vdash (q_0, \text{ƀ10ƀ}) \vdash (q_1, \text{ƀ11ƀ}) \ldots$$

Referencias

[1] Transparencias del Tema 4 de la asignatura: Máquinas de Turing, https://campusingenieriaytecnologia2122.ull.es/mod/resource/view.php?id=4244

[2] JFLAP: Java Formal Language and Automata Package, https://www.jflap.org/

[3] Instalación de JFLAP, https://campusingenieriaytecnologia2122.ull.es/mod/page/view.php?id=4232

[4] Susan H. Rodger and Thomas W. Finley: *An Interactive Formal Languages and Automata Package*. Jones & Bartlett Publishers, Sudbury, MA, 2006. ISBN 0763738344. https://www.jflap.org/jflapbook/jflapbook2006.pdf

[5] Ejemplo de DFA en JFLAP, https://www.youtube.com/watch?v=EzWxFMfqeFs

[6] Ejemplo de NFA en JFLAP, https://www.youtube.com/watch?v=audjcmz7ons

[7] Ejemplo Máquina de Turing Multi-Cinta en JFLAP, https://www.youtube.com/watch?v=fnnxpn9xLLE