Lenguajes de Programación: Computabilidad

Diana Marcela Cardona Román, M.Ing. Dr.Ing.
dcardona@unillanos.edu.co

Escuela de Ingeniería
Facultad de Ciencias Básicas e Ingeniería
Universidad de los Llanos

Resultado de aprendizaje

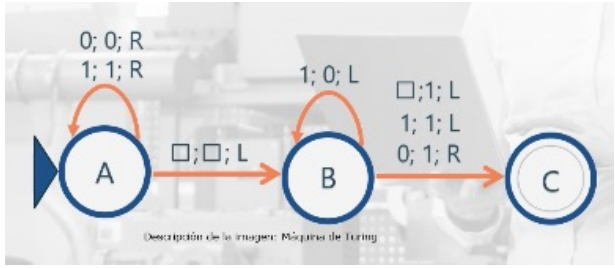
- RAC2 Analiza la aplicación, características, problemas, desempeño y computabilidad de modelos matemáticos o computacionales de autómatas, lenguajes, gramáticas y máquinas de Turing

Objetivo específico del curso

3. Identificar e implementar diferentes clases de autómatas y lenguajes, así como sus relaciones y propiedades.

Contenido

- Máquina de Turing
- Tesis de Church
- Gramática Dependiente del Contexto (GDC)
- Equivalencias entre gramáticas dependiente de contexto y máquina de Turing
- Lenguajes decidibles y aceptables
- Problemas indecidibles



¿Qué es Computabilidad?

- Consiste en ser capaz de encontrar la representación adecuada para la descripción de un problema o fenómeno.
- Para tal representación es necesario:
    - Un conjunto finito de símbolos.
    - Hacer asociaciones entre conceptos y elementos del lenguaje (de símbolos)
    - Encontrar las combinaciones adecuadas de símbolos para evitar ambigüedad.
    - Definir una manera de confirmar tal descripción para que terceros puedan reproducirla y llegar a los mismos resultados.

Noción de Modelo

- **Modelo**: es una especificación, generalmente en términos de un lenguaje matemático, de los pasos necesarios para reproducir un subconjunto determinado de la realidad. La representación del modelo surge siempre a partir de su descripción.

- ¿Es posible siempre pasar de la descripción de un modelo a su representación? ¿Todo lo que es descriptible puede ser representable?
    - Aparentemente la exactitud de la descripción hace depender a la exactitud de la representación.
    - Existen procesos que pueden ser descritos con gran exactitud, pero su representación o modelado no es posible.

Teoría de la Computabilidad

- La Teoría de la Computabilidad consiste en encontrar maneras de representar descripciones de procesos, de tal manera que se pueda asegurar si existe o no una representación.

- Se dice que un algoritmo es una manera formal y sistemática de representar la descripción de un proceso.

Noción de problema

Problema: es cualquier cuestión formulada en términos formales.

- Un problema es de decisión si su resolución implica la afirmación o negación de un predicado lógico. Existen otros tipos de problemas que no son de decisión (ej. Problemas de optimización y de búsqueda).

- Un problema de decisión es decidible si existe un algoritmo que lo resuelva (en caso contrario es indecidible).

- Un problema es tratable si existe un algoritmo rápido.

Máquina de Turing

Hopcroft

Contexto

- La Máquina de Turing (MT)* fue propuesta por Alan Turing en los años 30 como un modelo de máquina abstracta (modelo matemático), como una extensión de los autómatas finitos, que resultó en gran simplicidad y alto poder.

- Fue introducida en el artículo *On Computable Numbers*, donde Turing construyó un modelo formal de computador, denominado la Máquina de Turing, con esto resolvió el ***entscheidungsproblem*** (planteado por David Hilbert: ¿existe un algoritmo tal que pueda decidir si la proposición es cierta o por el contrario es falsa?) y en 1936 Alonzo Church y Alan Turing probaron, de forma independiente, la imposibilidad de la existencia de tal algoritmo, usando el cálculo lambda en el caso de Church y la máquina de Turing en el caso de Turing. Es decir, Turing demostró que había problemas tales que una máquina no podía resolver, esto es, que existen problemas irresolubles, inasequibles para cualquier MT.

* La "máquina" no se debe confundir con un aparato físico. Se trata más bien de una construcción matemática.
- *Entscheidungsproblem: determina la verdad o falsedad de cualquier proposición matemática*

Contexto

- La MT es un mecanismo que formaliza el concepto de algoritmo, que pretende ser lo suficientemente general como para resolver cualquier problema posible (siempre que sea computacionalmente abordable), fue introducida para demostrar la validez de los postulados de Gödel (teorema de incompletitud).

- Su importancia radica en que se considera la más poderosa de todas las máquinas abstractas conocidas.

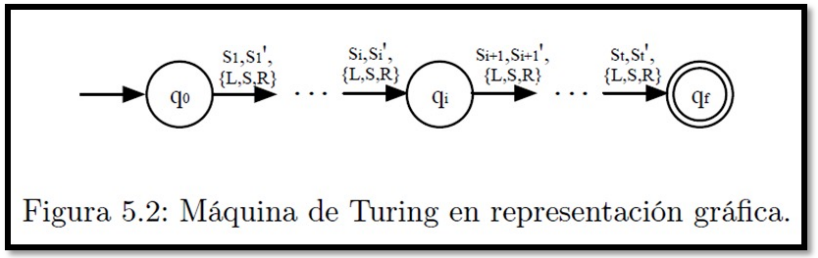


Figura 5.2: Máquina de Turing en representación gráfica.

Contexto – Tesis de Church

- La **Tesis Church-Turing** formula hipotéticamente la equivalencia entre los conceptos de función computable y máquina de Turing, así: “Cualquier forma general de computación nos permitirá calcular únicamente las funciones recursivas parciales”, que expresado en lenguaje corriente vendría a ser "**todo algoritmo es equivalente a una máquina de Turing**”.

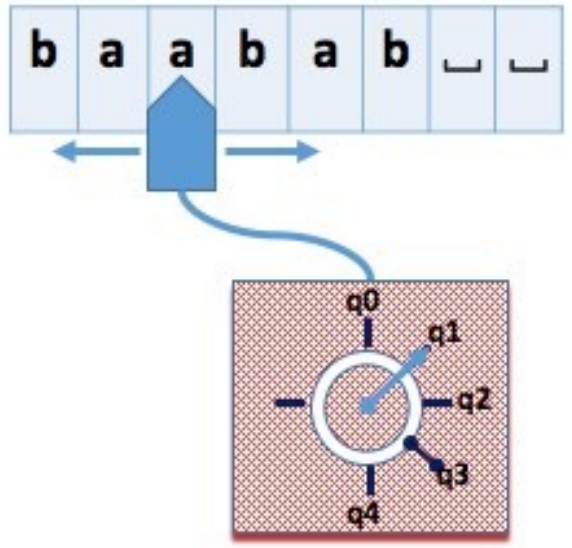
- La máquina de Turing es el primer modelo teórico de lo que luego sería un computador programable. Con el tiempo a este tipo de máquina se la conoció como máquina de estado finito, debido a que en cada etapa de un cálculo, la siguiente acción de la máquina se contrastaba con una lista finita de instrucciones de estado posibles.

Componentes de la máquina de Turing

Una máquina de Turing es un autómata finito que dispone de una única cinta de longitud infinita en la que se pueden leer y escribir datos.

Un movimiento o transición puede cambiar de estado (o quedarse en el estado actual), e escribir un símbolo (reemplazando el símbolo que exista o dejando el mismo) y mover la cabeza a la izquierda o derecha.

- Cinta de longitud infinita
- Control finito
- Cabeza lecto/escritora
- Movimiento izquierda/derecha



Componentes de la máquina de Turing

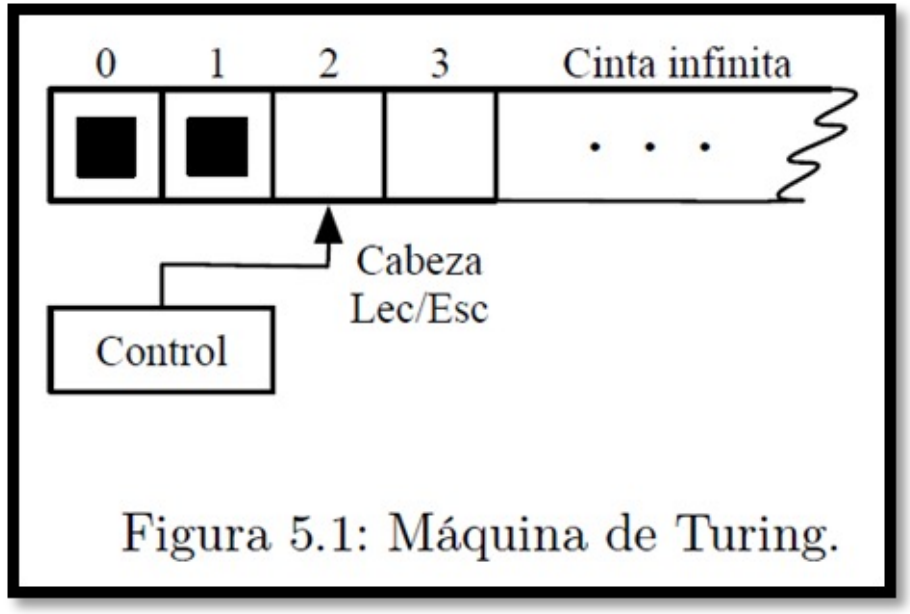


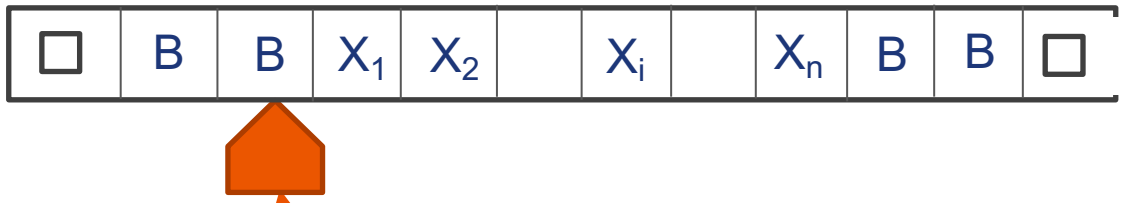
Figura 5.1: Máquina de Turing.

- **Una cinta de longitud infinita** dividida en celdas (cada celda puede tener solamente un símbolo tomado de un diccionario de símbolos predefinido).
- Un **control finito** que tiene la capacidad de examinar el algún símbolo de alguna celda y tomar una decisión que depende del símbolo observado y del estado en que se encuentre el control finito.
    - El control es finito porque puede estar solamente en alguno de los estados posibles, habiendo solamente un número finito de ellos.
- Se supone un **diccionario de símbolos finito**.

Descripción Máquina de Turing

- Una MT consiste en un **control finito** que puede estar en *cualquier estado* de un **conjunto finito de estados**. Se tiene una **cinta** dividida en celdas, cada celda con un símbolo.
- Inicialmente, la entrada (cadena finita de símbolos del alfabeto) se coloca en la cinta, el resto de las celdas tienen el símbolo especial vacío o espacio en blanco.

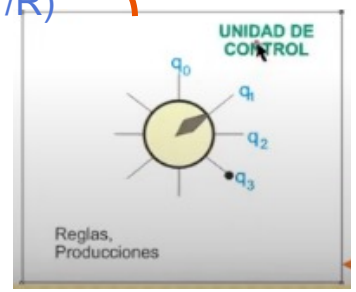
Cinta que almacena el alfabeto o la cadena de carateres



Cabezal

Lectura / Escritura

Se mueve bidireccionalmente (L /R)



Cinta de longitud infinita hacia la derecha ➔

Llenándose los espacios con el carácter blanco □

Definición Formal MT

Formalmente, una máquina de Turing se define mediante una séptupla:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$$

donde :

- Q: es un conjunto finito de estados de control
- Σ: es un conjunto finito de símbolos distinto del espacio en blanco, denominado alfabeto de máquina o de entrada
- Γ: es un conjunto finito de símbolos de cinta, denominado alfabeto de cinta ($\Sigma \subseteq \Gamma$)
- $q_0 \in Q$: el estado inicial
- $B \in \Gamma$: es el símbolo blanco (el símbolo B no puede hacer parte de ∑), y es el único símbolo que se puede repetir un número infinito de veces, aparece en todas las casillas excepto en aquellas que contienen los símbolos de entrada.
- $F \subseteq Q$: es el subconjunto de estados finales o de aceptación
- δ: es la función de transición $\delta = (Q \times \Gamma) \rightarrow (Q \times \Gamma \times \{L, R\})$; es una función parcial donde L, R = [Left, Right]

  $\delta(q, X) = (p, Y, Dir)$

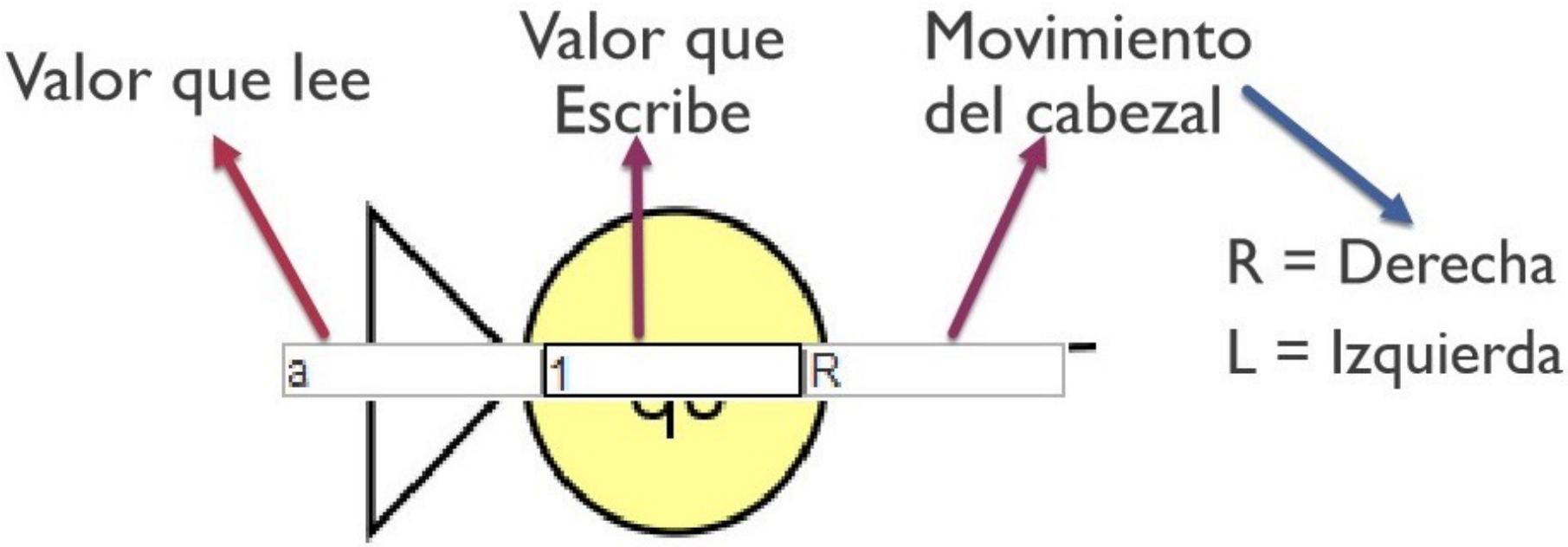
  q es un estado, X un símbolo de la cinta

  p es un nuevo estado, en Q;

  Y es un símbolo en Γ que substituye X;

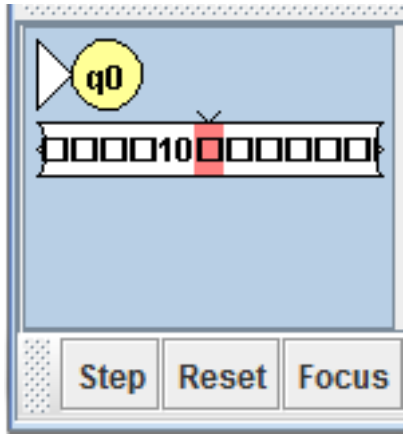
  Dir es el sentido o dirección en que la cabeza se mueve, puede ser: R (Derecha) o L (Izquierda),

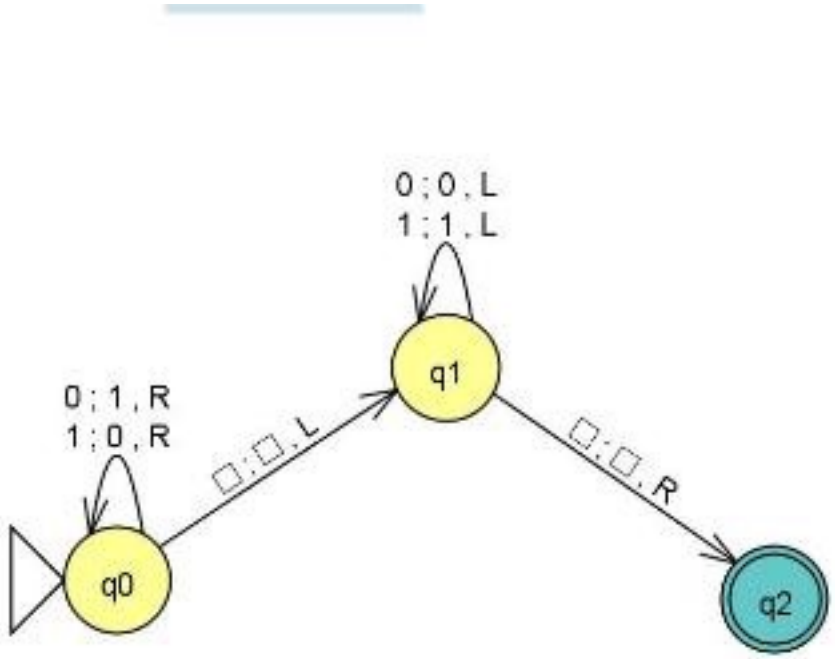
Representación gráfica de la MT – Valores que recibe



Ejemplo Transductor

- Cambia “0”s por “1”s y “1”s por “0”s). El mismo alfabeto se usa tanto para las cadenas de entrada como para la cinta.





Input	Output	Result
		Accept
11	00	Accept
00	11	Accept
111	000	Accept
010101	101010	Accept
1010	0101	Accept
11111111	00000000	Accept
00000000	11111111	Accept
10101010	01010101	Accept
1	0	Accept
0	1	Accept

Ejemplo Transductor

- Diseñe una MT que se comporte como transductor que reconozca el lenguaje L = {a}* (incluye la cadena λ). La transducción (salida) debe ser que por cada símbolo que entre, se duplique. Ej. para la cadena ‘aa’ la salida será ‘aaaa’. El alfabeto de la cinta debe ser diferente al alfabeto de entrada.

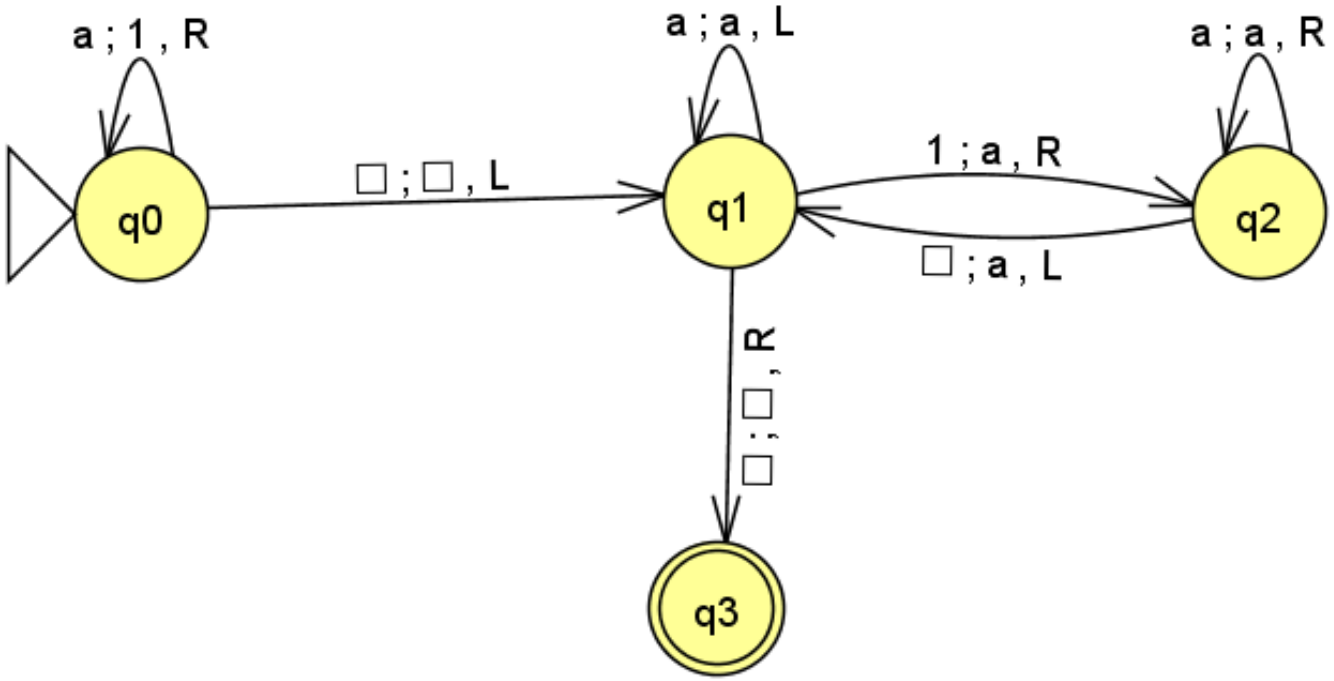


Table Text Size

Input	Output	
a	aa	Accept
aa	aaaa	Accept
aaa	aaaaaa	Accept
aaaa	aaaaaaaa	Accept
aaaaa	aaaaaaaaaa	Accept

Ejemplo Transductor – Definición formal

La máquina de Turing con una sola cinta puede definirse como una 7-tupla

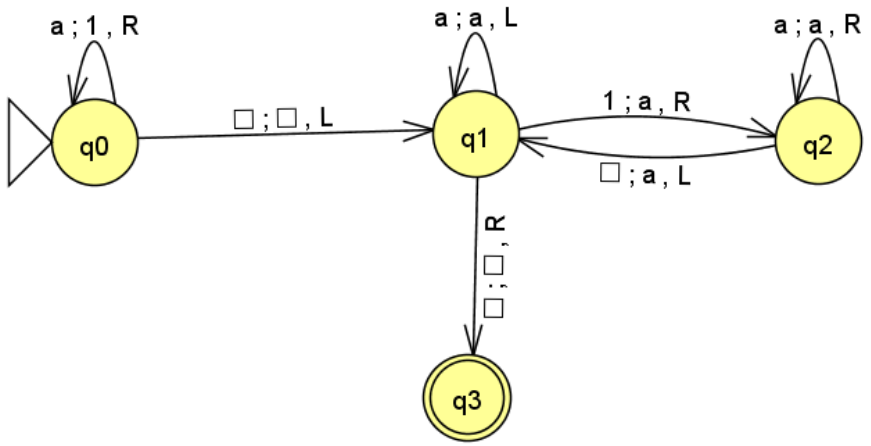
$$MT = (K, \sum, \Gamma, s, b, F, \delta) \quad \text{donde:}$$

- ✓ K = $\{q_0, q_1, q_2, q_3\}$ es el conjunto de estados, tal que h ∈ K.
- ✓ Σ = {a }, es el alfabeto de entrada, donde Ц ∉ Σ;
- ✓ Γ = {1 }, es el alfabeto de la cinta, donde Ц ∈ Γ y Σ ⊆ Γ ;
- ✓ s = $q_0 \in$ K es el estado inicial;
- ✓ F = $q_3 \subseteq$ K es el estado final;

b = □

δ(q, X) = (p,Y,Dir) => (q0, a) = (q0, 1, R)

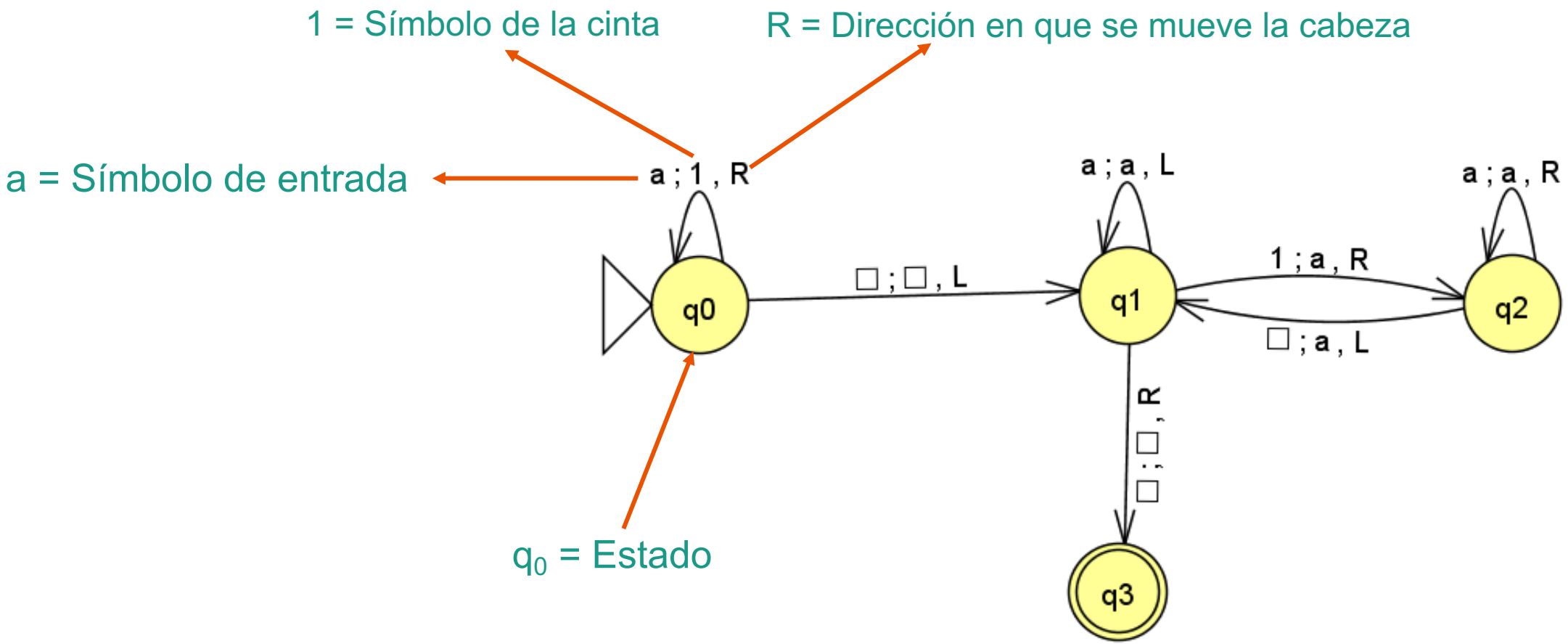
…



Ejemplo Transductor – Hallar las transiciones

$\delta(q, X) = (p,Y,S)$

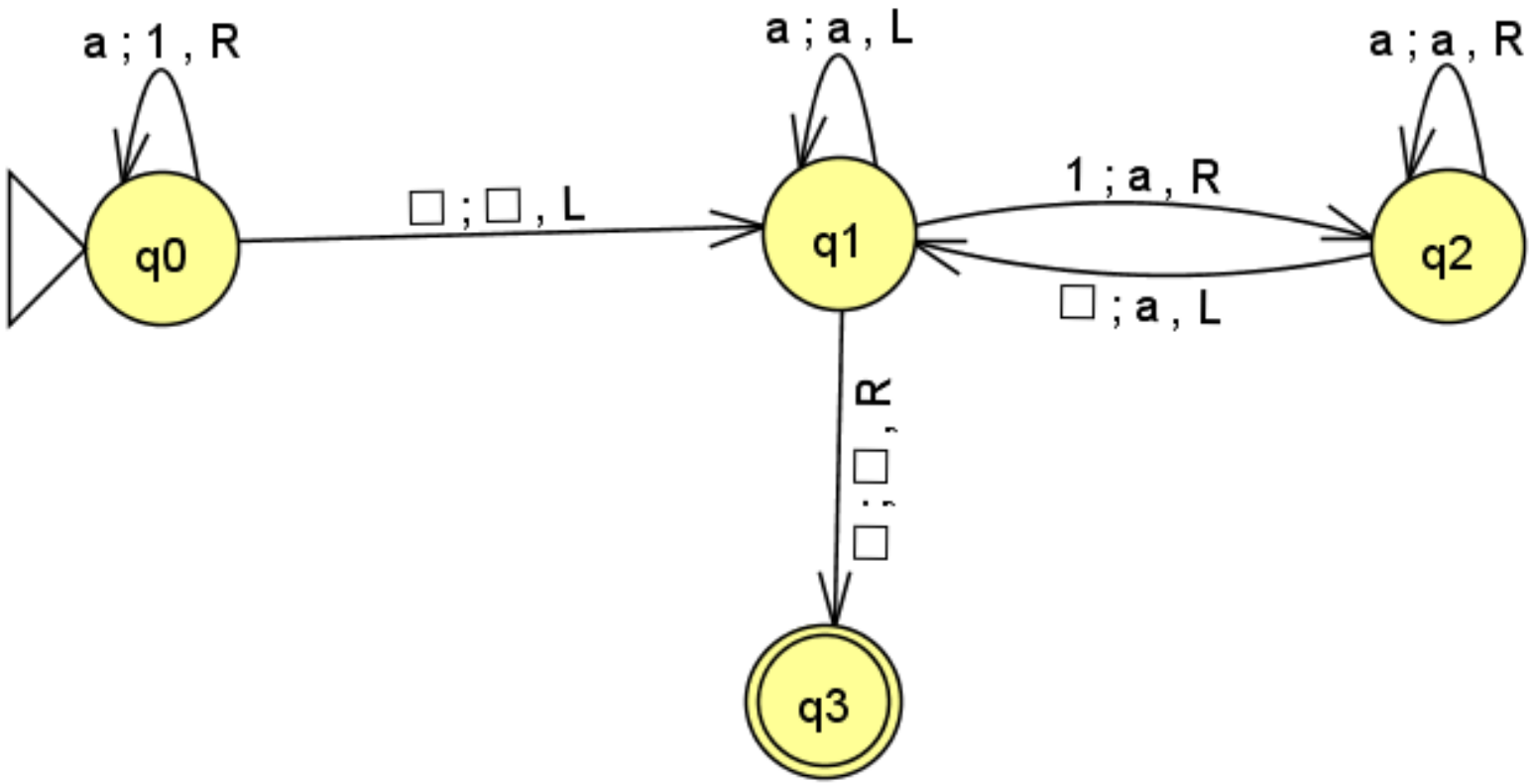
$\delta(q0, a) = (q0, 1, R)$



Ejemplo Transductor – Transiciones δ

✓ $\delta : (K - \{h\} \times \Gamma) \rightarrow K \times (\Gamma \cup \{L, R\})$ La función de transición donde L es un movimiento a la izquierda y R es el movimiento a la derecha.

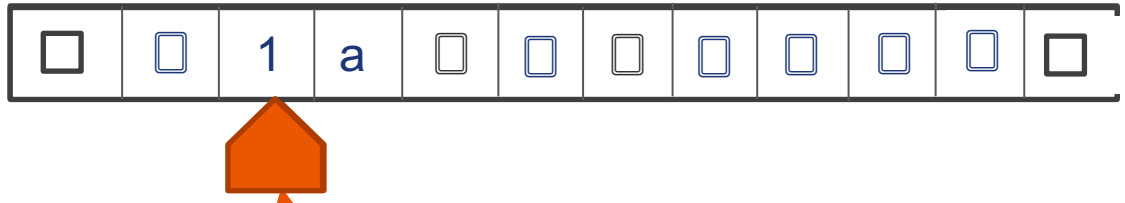
$\delta\ (q_0, a) = (q_0, 1, R)$
$\delta\ (q_0, Ц) = (q_1, Ц, L)$
$\delta\ (q_1, a) = (q_1, a, L)$
$\delta\ (q_1, 1) = (q_2, a, R)$
$\delta\ (q_2, a) = (q_2, a, R)$
$\delta\ (q_2, Ц) = (q_1, a, L)$
$\delta\ (q_1, Ц) = (q_3, Ц, R)$

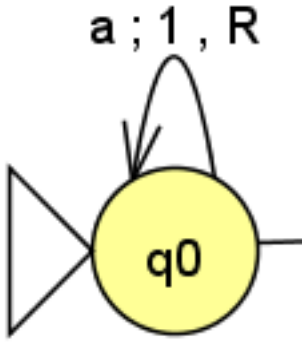


Recorrido de una cadena

- Se parte del estado inicial, y la cinta contiene símbolos de entrada.
- Va leyendo una celda de la cinta, luego borra el símbolo, para después escribir el nuevo símbolo perteneciente al alfabeto de salida y finalmente avanza a la izquierda o a la derecha (solo una celda a la vez, según la función de transición), repitiendo esto según se indique en la función de transición, para finalmente detenerse en un estado final o de aceptación, representando así la salida y la aceptación de la cadena. Si la cabeza lectora rebasa el extremo izquierdo de la cinta, la cadena es rechazada y el proceso termina (terminación anormal).
- Las operaciones que se pueden realizar en esta máquina se limitan a: avanzar el cabezal lector/escritor hacia la derecha. Avanzar el cabezal lector/escritor hacia la izquierda. El cómputo es determinado a partir de una tabla de estados de la forma: (estado, valor) (nuevo estado, nuevo valor, dirección).

Cadena: aa





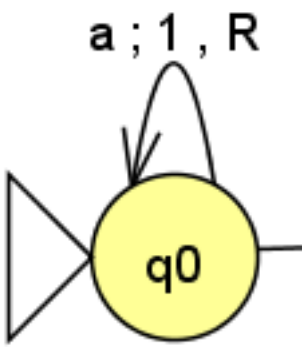
Cabezal
Lectura / Escritura

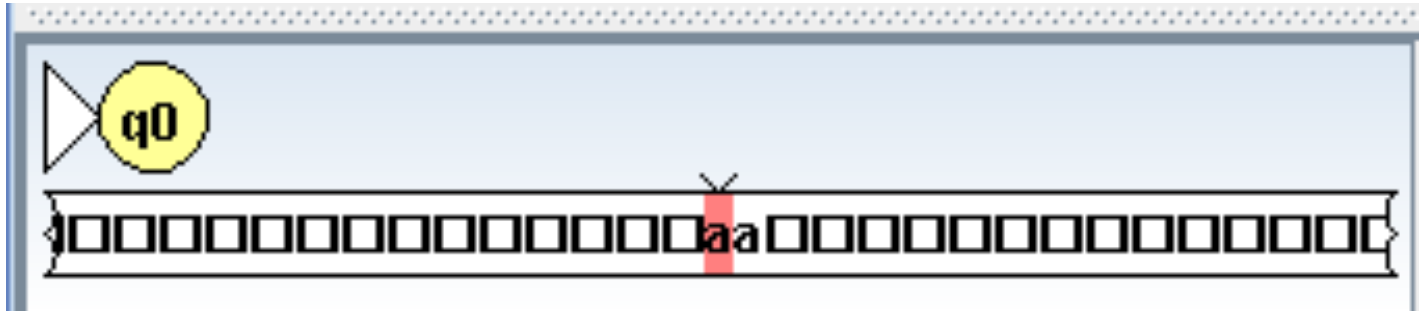
a es Símbolo que lee

1 es Símbolo que escribe

Avanza a la Derecha (R)

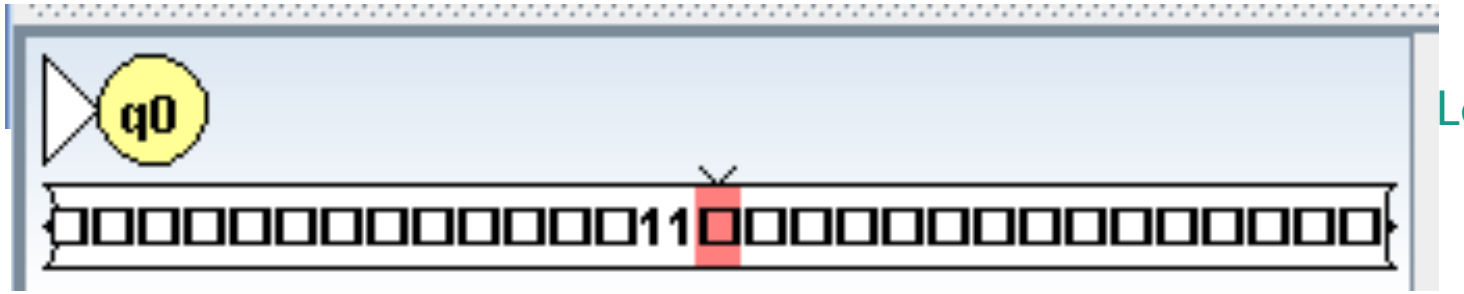
Recorrido de la cadena ‘aa’



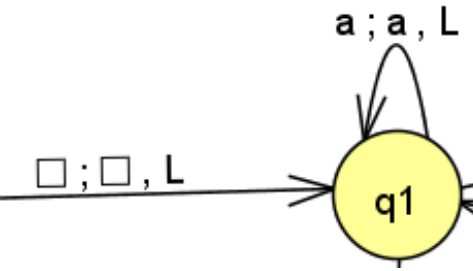


a es Símbolo que lee

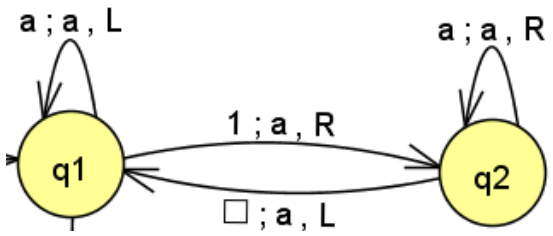
1 es Símbolo que escribe

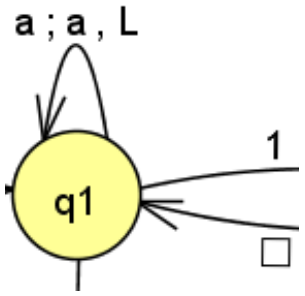


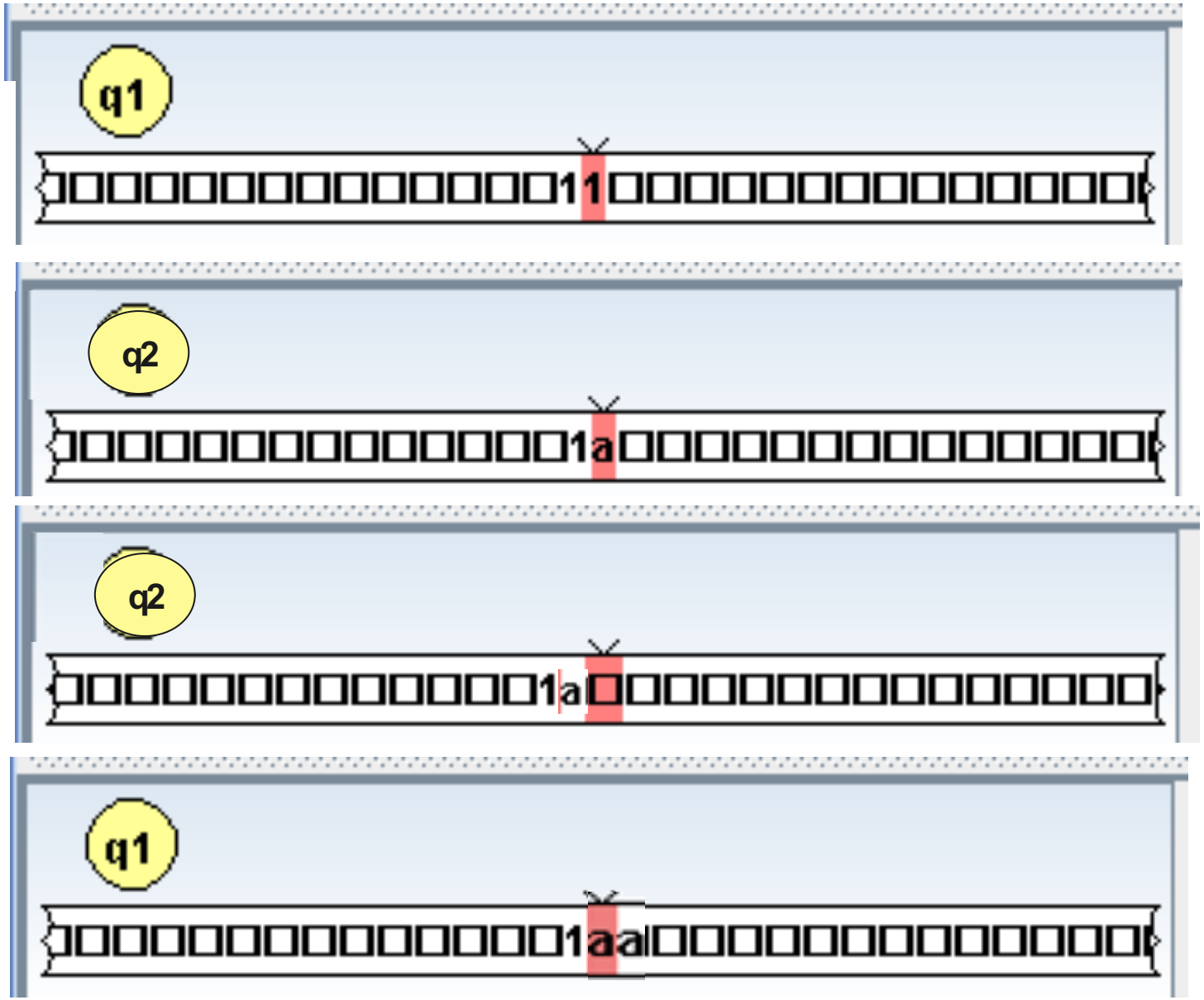
Lee blanco



Recorrido de la cadena ‘aa’







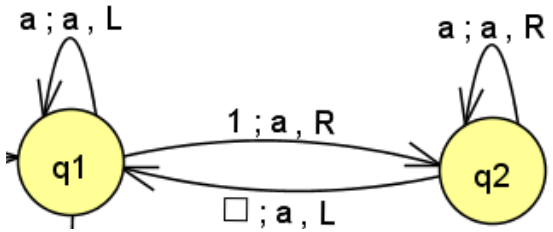
1 es Símbolo que lee

a es Símbolo que escribe

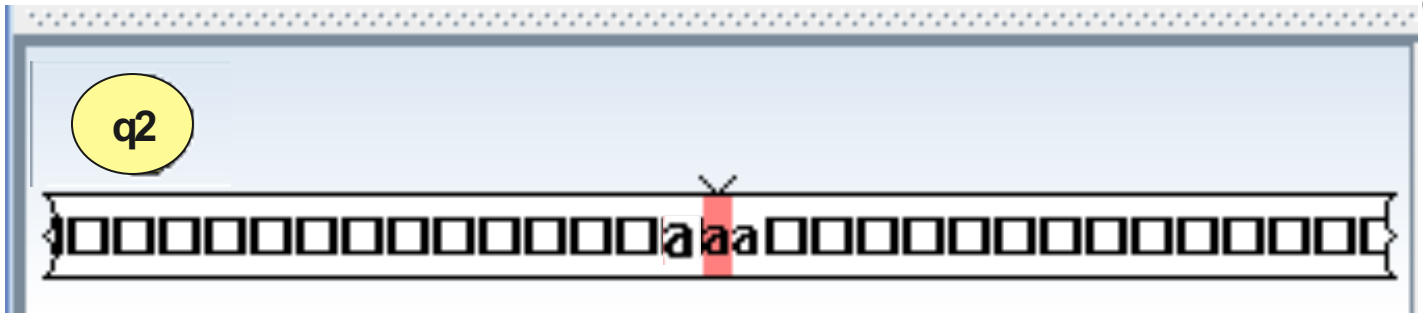
Lee blanco

a es Símbolo que lee y escribe

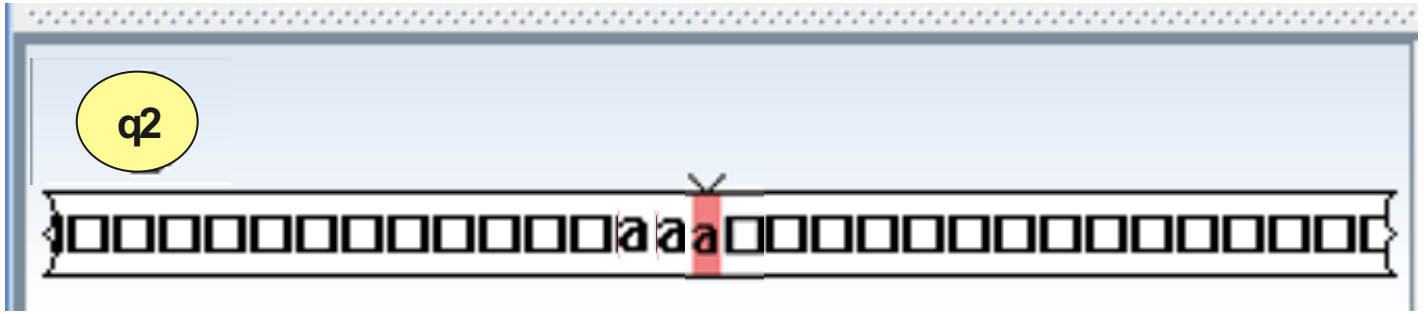
Recorrido de la cadena ‘aa’

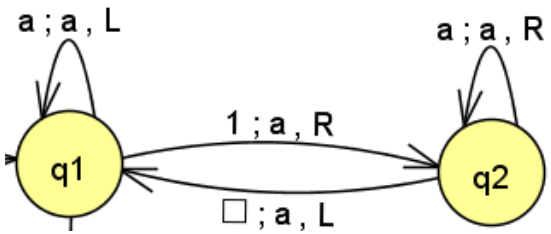


1 es Símbolo que lee
a es Símbolo que escribe

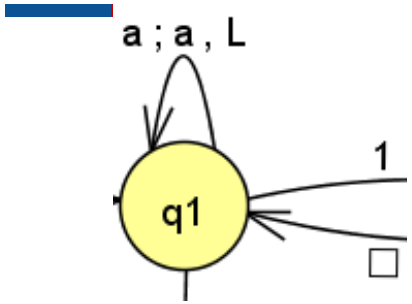


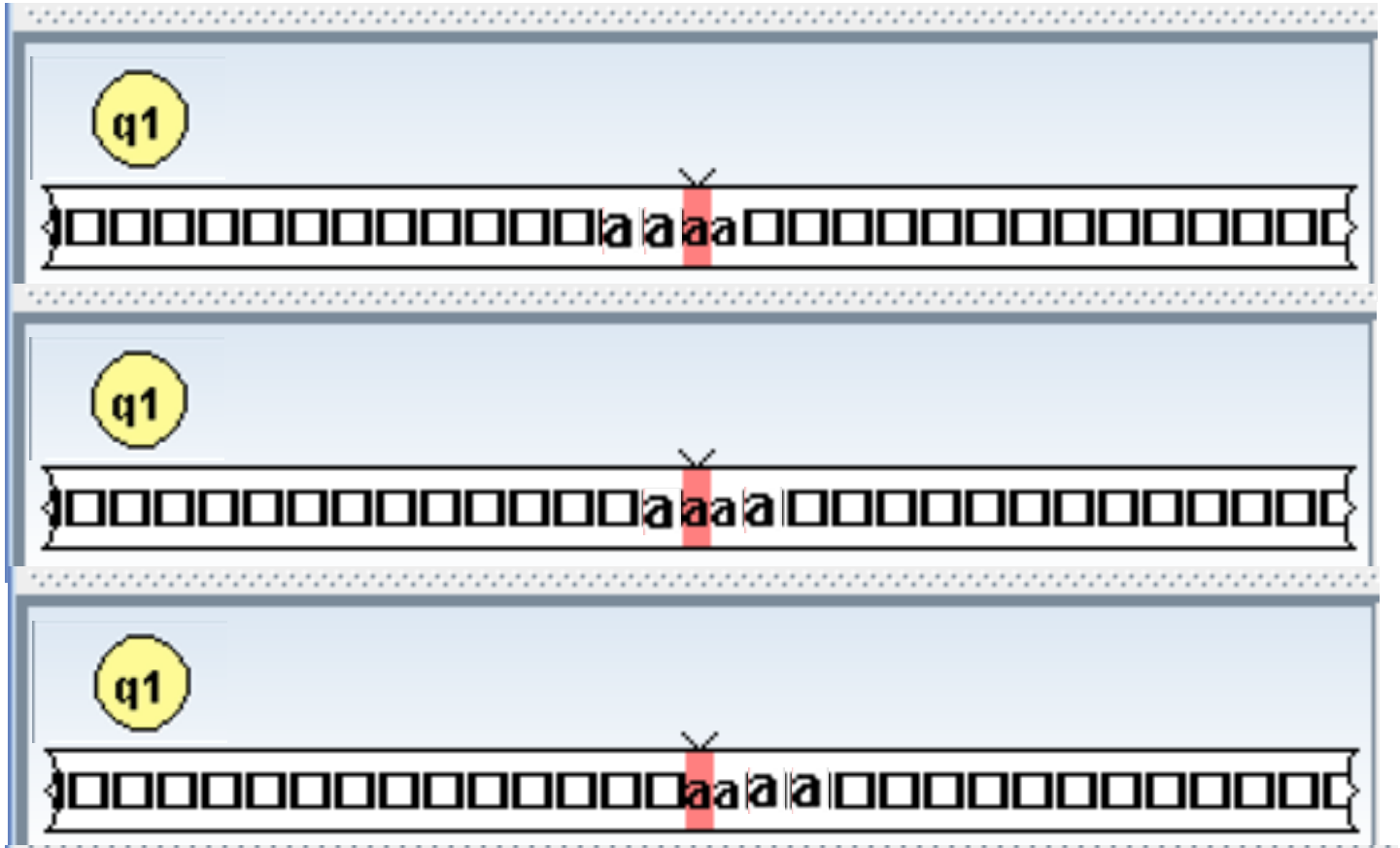
a es Símbolo que lee y escribe



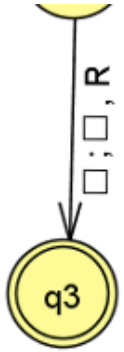


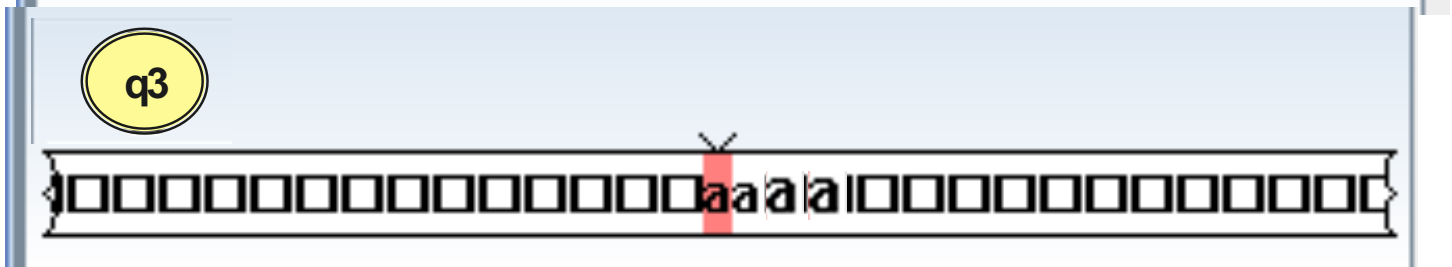
Lee blanco
Escribe a





Lee a y escribe a





Lee blanco y escribe blanco

Ejemplo

- Esta M.T. obtiene el **complemento binario** de un número binario almacenado en la cinta.

Q={q0,q1} F ={q1} Σ={0,1} Γ={0,1,b} La función de transición se define en la siguiente tabla:

	0	**1**	**b**
q_0	$q_0 1 D$	$q_0 0 D$	$q_1 b I$

Ejemplo

- Esta M.T. **calcula la paridad** del número de 1's que hay en la cadena binaria de entrada. Al final del proceso, se añade a la derecha de la cadena un 0 para indicar que hay un número par de 1's y un 1 para indicar que el número de 1's es impar.
- $h(n)\begin{cases} 0, sí\ n\ es\ par \\ 1, si\ n\ es\ impar \end{cases}$

Q = {q_0,q_1,q_2} F = {q_2} Σ = {0,1} Γ = {0,1,b} La función de transición se define en la siguiente tabla:

	0	**1**	**b**
q_0	$q_0 0 D$	$q_1 1 D$	$q_2 0 I$
q_1	$q_1 0 D$	$q_0 1 D$	$q_2 1 I$

Ejercicio

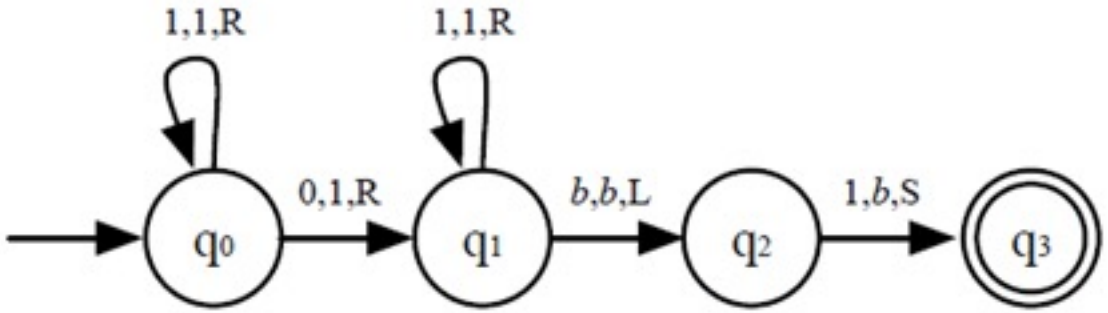


Figura 5.5: Máquina de Turing que obtiene la suma de dos números naturales.

1. Comprobar que la MT suma dos números naturales (en su extensión binaria).
2. Diseñe una MT que acepte el Lenguaje L = $\{0^n1^n \mid n \geq 1\}$. Pista: *Sobre la cinta se escribe una secuencia finita de ceros y unos, precedida y seguida por un número infinito de espacios en blanco. Asegúrese que sea de la forma 0*1*. Alternativamente, la MT cambiará primero un 0 por una X y se moverá hacia la derecha y luego un 1 por una Y y se mueve hacia la izquierda, hasta que se hayan cambiado todos los ceros y unos.* $\Sigma = \{0,1\}$ *y* $\Gamma = \{0,1,X,Y,B\}$

Lenguajes aceptados para una Máquina de Turing

Lenguaje Tipo 0

Sea M = (Q, Σ, Γ, δ, q0, B, F) una Máquina de Turing

Lenguaje aceptado se define:

$$L(M) = \{ w \mid w \text{ en } \Sigma^* , q_0 w \vdash^* \alpha_1 q \alpha_2, \alpha_1 \text{ y } \alpha_2 \text{ en } \Gamma^* \text{ y } q \text{ en } F\}$$

L(M) es un lenguaje recursivamente enumerable.

Lenguaje Recursivo (decidible = algoritmo) Existe una MT para ello

Lenguajes aceptados para una Máquina de Turing

- Una cadena de entrada w es aceptada por una MT M si el cómputo que se indica la configuración inicial q0w termina en una configuración instantánea:

w_1pw_2 *p* es un estado de aceptación, en la cual M se detiene completamente. El lenguaje L(M) aceptado por una MT M se define como:

$$L(M) = \{w \in \Sigma : q_0w \vdash^* w_1pw_2, p \in T\}$$

$L(M) := \{w \in \Sigma^* : q_0w \vdash * w_1pw_2, p \in F, w_1, w_2 \in \Gamma^*,$ M se detiene en la configuración $w_1pw_2\}$.

M se para en:
w1 p w2 , Si la cadena de entrada en una máquina M pertenece a L(M), la maquina M siempre se detiene.

Aceptan lenguajes formales que pueden ser generados por una gramática de tipo 0: recursivamente innumerable (r.e) Las maquinas de Turing son los reconocedores de lenguaje más poderosos que existen.

usaremos $\vdash$ para indicar los movimientos. Como es habitual, utilizaremos $\overset{*}{\underset{M}{\vdash}}$, o sólo $\vdash^*$, para indicar cero, uno o más movimientos de la MT M.

Lenguajes recursivamente enumerables

Lenguaje recursivamente enumerable: si existe una máquina de Turing M tal que L(M) = L.
Define la clase Lr.e.

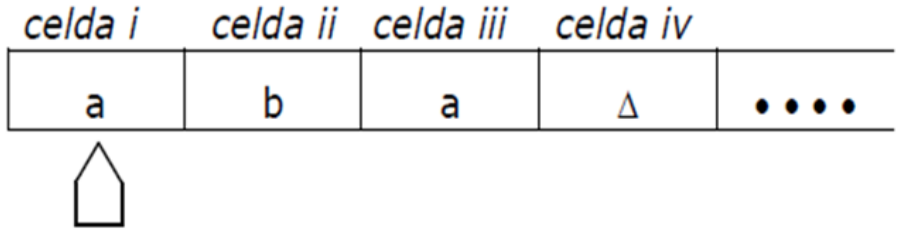
Lenguaje recursivo: si existe una máquina de Turing M tal que L(M)=L y M se detiene ante cualquier entrada. Define la clase Lrec

Lrec es subconjunto de Lr.e

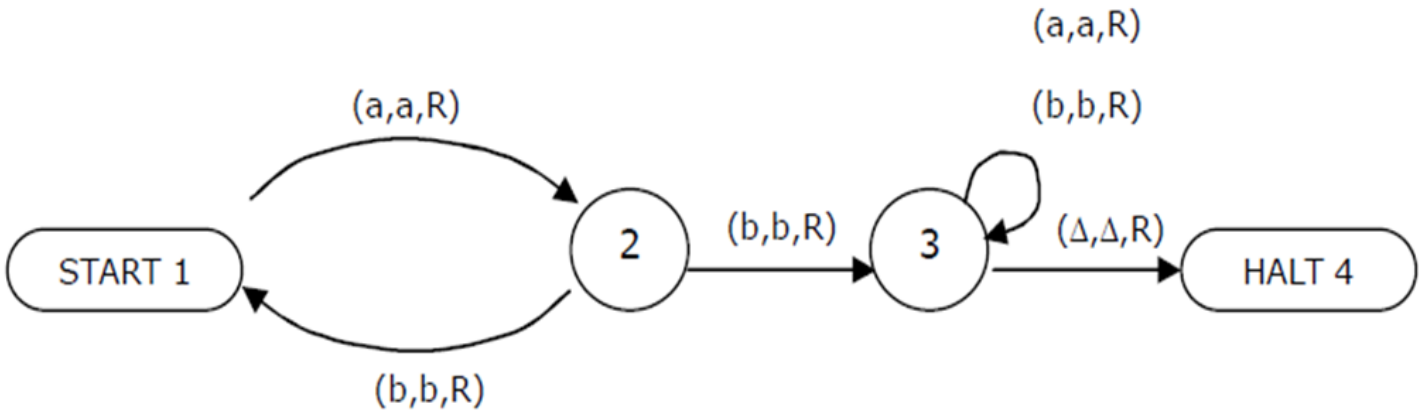
Una función es computable si puede ser calculada por una máquina de Turing.

Reconocimiento de un Lenguaje Regular

La siguiente es la CINTA de una máquina de Turing que correrá sobre la entrada **aba**.



El programa para esta MT es dado como un grafo dirigido con aristas etiquetadas como se muestra a continuación:



Este es un lenguaje regular porque puede ser definido también por la expresión regular: (a+b)b(a+b)*

Unidad de control estacionaria

Para simplificar la descripción del modelo estándar, se ha exigido que la unidad de control se desplace hacia la izquierda o hacia la derecha en cada transición. No obstante, se puede permitir que la unidad de control no se mueva en un determinado paso computacional, es decir, que no realice ningún desplazamiento. Este tipo de transición lo escribimos en la forma:

- $\delta(q,a) = (p,b,-)$
- donde $a,b \in \Gamma$; $p,q \in Q$ y $-$ significa que la unidad de control no se mueve. Tal transición se puede simular por medio de un movimiento a la derecha seguido de un retorno a la izquierda. Así, para simular la transición $\delta(q, a) = (p, b, -)$ utilizamos un estado auxiliar nuevo q' y las transiciones:
- $\delta(q, a) = (q', b, \rightarrow)$,
- $\delta(q', s) = (p,s,\leftarrow)$, para todo símbolo $s \in \Gamma$.

Propiedades

Propiedad	LR	LLC	LRE
U (unión)	S	S	S
∩ (intersección)	S	N	S
complemento	S	N	N
concatenación	S	S	S
∩ LR	S	S	S

- LR: Lenguaje Regular
- LLC: Lenguaje Libre de Contexto
- LRE: Lenguaje Recursivamente Enumerable

Propiedades

Propiedad	LR	LLC	LRE
Kleene-clausura	S	S	S
Reflejo	S	S	S
Morfismo	S	S	S
Morfismo^{-1}	S	S	S
Diferencia	S	N	S

- LR: Lenguaje Regular
- LLC: Lenguaje Libre de Contexto
- LRE: Lenguaje Recursivamente Enumerable

Jerarquía de Chomsky

Tipo	Lenguaje	Máquina	Gramática G = (V,T,P,S)
0	**Recursivamente enumerable**	**Máquina de Turing**	**Gramática sin restricciones** **α → β** (α, β en (V U T)*, α contiene una variables)
1	**Dependiente del Contexto**	**Autómata linealmente acotado**	***Gramática sensible al contexto*** **α → β** (α, β en (V U T)*, α contiene una variables, |β| ≥ |α|)
2	**Independiente del Contexto**	**Autómata de Pila**	***Gramática libre de contexto*** ***A → α*** (A en V y α en (V U T)*)
3	**Lenguaje Regular**	**Autómata finito**	***Gramática Regular*** ***A → aB*** ***A → a*** (A,B en V y a en T)

Lenguajes sensibles al contexto

Gramáticas sensibles al contexto

- Una gramática G = (V,T,P,S) es una gramática sensible de contexto si todas las producciones son de la forma

$$\alpha \rightarrow \beta$$

- donde α, β en $(V+T)^*$ y $|\alpha| \leq |\beta|$, α contiene al menos una variable V

Ejemplo: $L = \{ a^n b^n c^n \mid n > 0 \}$

$S \rightarrow A_0BCS_1 \mid A_0BC$

$S_1 \rightarrow ABCS_1 \mid ABC$

$BA \rightarrow AB$

$CA \rightarrow AC$

$CB \rightarrow BC$

$A_0 \rightarrow a$

$aA \rightarrow aa$

$aB \rightarrow ab$

$bB \rightarrow bb$

$bC \rightarrow bc$

$cC \rightarrow cc$

Gramáticas sin restricciones (irrestricta)

- Una gramática G = (V,T,P,S) es una gramática de tipo 0 (irrestricta) si todas las producciones son de la forma

$$\alpha \rightarrow \beta$$

- donde α en $(V+T)^+$, β en $(V+T)^*$

Teorema: L es un lenguaje recursivamente enumerable si L=L(G) donde G es de tipo 0.

Ejemplo: L = { ww | w en (a + b)* }

S → FM
Aa → aA
Ba → aB
AM → Ma
F → λ

F → FaA | FbB
Ab → bA
Bb → bB
BM → Mb
M → λ

Aplicaciones de la MT

- Teoría de la computación:
    - Rama de la matemática y la computación que centra su interés en las limitaciones y capacidades fundamentales de las computadoras
- Generadora de lenguajes:
    - En la cinta de salida se llevan a cabo operaciones de escritura qu van llenando la cinta con las palabras del lenguaje
- Codificación:
    - Proceso en donde el emisor converte las ideas que quiere transmitir en signaos que puedan ser recibidos fácilmente por el receptor
- Decodificación:
    - El receptor transforma el código utilizado por el emisor para interpretar los signos empleados

Síntesis

- La MT es una máquina de computación
- La MT realiza movimientos basados en su estado actual y en el símbolo de cinta de la casilla a la que apunta la cabeza de la cinta. En un movimiento, cambia el estado, sobrescribe la celda señalada por la cabeza con algún símbolo de cinta y mueve la cabeza una celda hacia la izquierda o hacia la derecha.
- Las MT tienen diferentes usos, uno es el validar problemas indecibibles, que no se intentan resolver; los problemas intratables, que requiere de más tiempo y del análisis del diseñador para determinar que se quiere hacer.

    - Suponemos que una máquina de Turing siempre se detiene cuando está en un estado de aceptación. Lamentablemente, no siempre es posible exigir que una MT se pare si no acepta. Los lenguajes reconocidos por máquinas de Turing que no se detienen, independientemente de si aceptan o no, se denominan recursivos. Las máquinas de Turing que siempre se paran, independientemente si aceptan o no, son un buen modelo de “algoritmo”. Si existe un algoritmo que permite resolver un determinado problema, entonces decimos que el problema es “decidible”, por lo que las máquinas de Turing que siempre se paran desempeñan un papel importante en la teoría de la decidibilidad.

¡GRACIAS!

¿Preguntas?