Capítulo 13.

Máquinas de Turing

13.1. Conceptos generales

Definición. Descripciones instantáneas. Lenguaje reconocido por una MT. Función computada por una MT.

13.2. Otras definiciones

Equivalencia. Máquinas con alfabeto binario, Máquinas no deterministas. Máquinas con dos cintas.

13.3. MT para reconocer lenguajes

MT equivalente a una gramática de tipo 0. Gramática de tipo 0 equivalente a una MT.

13.4. MT para computar funciones

Funciones de un parámetro. Funciones de varios parámetros. Funciones complejas.

13.1. Conceptos generales

Definición

Una máquina de Turing es una séptupla $M= (\Gamma,\Sigma,\bullet,Q,q_0,f,F)$ donde :

1. Γ es el alfabeto de símbolos de la cinta
2. $\Sigma \subset \Gamma$ es el alfabeto de símbolos de entrada
3. $\bullet \in \Gamma$ es el símbolo blanco que no pertenece a Σ
4. Q es un conjunto finito de estados
5. $q_0 \in Q$ es el estado inicial
6. $F \subseteq Q$ es el conjunto de estados finales
7. f es una función de transición parcial

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

Intuitivamente:

Dispositivo capaz de adoptar un estado determinado (de Q) y que está conectado a una cabeza lectura/escritura que puede leer y escribir símbolos en una cinta infinita. En cada acción o movimiento:

1. la máquina lee el símbolo de la cinta en la posición donde se encuentra la cabeza de lectura/escritura
2. en función del símbolo leído y del estado actual la máquina:
   a. pasa a un nuevo estado (de forma determinista)
   b. imprime un símbolo en la cinta en la misma posición donde acaba de leer el símbolo actual
   c. mueve la cabeza de lectura/escritura una posición a la izquierda (L), o a la derecha (R).

Inicialmente, la cinta (infinita a ambos lados) contiene un número finito de símbolos de Σ, precedidos y seguidos por un número infinito de blancos. La cinta se comporta como un dispositivo de entrada/salida.

Ejemplo 1:

$M = (\{a,b,\bullet\},\{a,b\},\bullet,\{q_0,q_1\},q_0,f,\{q_1\})$

$f(q_0,a)=(q_0,a,R)$

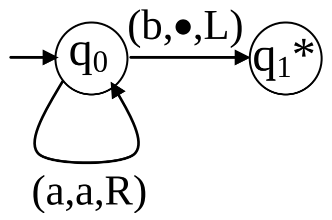
$f(q_0,b)=(q_1,\bullet,L)$

... • a ... a b • ... $\Rightarrow$ para en ... • a ... a • • ...
$\quad q_0 \qquad\qquad\qquad\qquad\qquad\qquad q_1$

... • a ... a • ... $\Rightarrow$ para en ... • a ... a • ...
$\quad q_0 \qquad\qquad\qquad\qquad\qquad\qquad q_0$

$\Rightarrow$ pasa todos los a's que haya y cuando encuentra una b lo cambia por • y para en el estado q_1

Representación gráfica:



- cada nodo corresponde a un estado de la MT
- cada transición $f(q_i,a)=(q_j,b,X)$ con $q_i,q_j \in Q$, $a,b \in \Gamma$ y $X \in \{L,R\}$ corresponde a un arco del nodo q_i al nodo q_j marcado con la etiqueta (a,b,X)
- el estado inicial se marca con $\rightarrow$ y los estado finales con *

<u>Descripciones instantáneas</u>

- contiene el contenido de la cinta (sólo escribiremos los blancos cuando sea necesario)
- se incluye el estado de la máquina en la posición delante del símbolo donde se encuentra la cabeza lectora/escritora

q_0aaaab o $aaaabq_1\bullet$

Movimiento:

$q_0aaab \vdash aq_0aab$ posible si y sólo si $f(q_0,a)=(q_0,a,R)$
$q_0aaab \vdash q_0\bullet baab$ posible si y sólo si $f(q_0,a)=(q_0,b,L)$
$aaaq_0b \vdash aaa\bullet q_1\bullet$ posible si y sólo si $f(q_0,b)=(q_1,\bullet,R)$

Cierre transitivo de $\vdash$ (serie de movimientos):

$q_0aaab \vdash^* aaaq_1\bullet$ si existe $q_0aaab \vdash \ldots \vdash aaaq_1\bullet$

Una MT comienza en un estado inicial con alguna información en la cinta, $x_1q_ix_2$,realiza una serie de movimientos y posiblemente:

1. Entre en un bucle infinito (**no pare nunca**):
   $x_1q_ix_2 \vdash^* \infty$
2. Llegue a una configuración en la que no es posible ningún movimiento:
   $x_1q_ix_2 \vdash^* y_1q_jay_2$ (para algún q_j y a, tal que $f(q_j,a)$ no esta definido)

Si, además, $q_j \in F$, decimos que la máquina para en un estado final. **(por convención no definimos transiciones $f(q_j,a)$ para ningún estado final q_j)**

Ejemplo 2:
$M= (\{a,b,\bullet\},\{a,b\},\bullet,\{q_0,q_1,q_2\},q_0,f,\{q_2\})$
$f(q_0,a)=(q_1,a,R)$ $f(q_1,b)=(q_0,b,L)$ $f(q_1,\bullet)=(q_0,\bullet,L)$
$f(q_0,b)=(q_2,b,R)$ $f(q_1,a)=(q_0,a,L)$

diferentes casos:
$q_0\bullet$ $\Rightarrow$ para en $q_0\bullet$
$q_0bx_1x_2...x_n \vdash bq_2x_1x_2...x_n$ $\Rightarrow$ para en q_2
$q_0ax_1x_2...x_n \vdash aq_1x_1x_2...x_n \vdash q_0ax_1x_2...x_n$ ... (bucle infinito)

Lenguaje reconocido por una MT

El contenido de la cinta al iniciar una máquina puede interpretarse como una palabra de un determinado lenguaje.

Definición:
Sea M= $(\Gamma,\Sigma,\bullet,Q,q_0,f,F)$ una MT. El lenguaje aceptado por M es

$$L(M) = \{x \mid q_0x \vdash^* y_1q_iay_2, \text{ con } q_i\in F,\ x\in\Sigma^*, y_1,y_2\in\Gamma^*,\ a\in\Gamma \text{ y } f(q_i,a) \text{ no esta definido}\}$$

Convenciones:
-Inicialmente la cinta contiene una palabra x y la cabeza de lectura/escritura se encuentra en la posición del primer símbolo de x en el estado q_0: q_0x
-La máquina acepta x si se para en un estado final.
-El contenido de la cinta al parar es irrelevante.
-Se rechaza x si la máquina no se para nunca o se para en un estado que no es final.
-Si la palabra es λ, la configuración inicial es $q_0\bullet$.

Ejemplo 1: $L=\{x \mid x\in\{a,b\}^+$ y número de b en x $n_b(x)>0\}$
Ejemplo 2: $L=\{bx \mid x\in\{a,b\}^*\}$

Ejemplo 3:

$M= (\{a,b,\bullet,1\},\{a,b\},\bullet,\{q_0,q_1,q_2,q_3,q_4,q_5\},q_0,f,\{q_5\})$
$f(q_0,a)=(q_1,\bullet,R)$ — lee dos a's y eliminarlos
$f(q_1,a)=(q_2,\bullet,R)$
$f(q_2,b)=(q_3,\bullet,R)$ — lee una b y elimínala
$f(q_3,a)=(q_3,a,R)$ — pasa todas las a's y b's hasta el final

$f(q_3,b)=(q_3,b,R)$
$f(q_3,\bullet)=(q_4,1,L)$ — añade un 1 al final (si ya hay unos pásalos)
$f(q_3,1)=(q_3,1,R)$
$f(q_4,1)=(q_4,1,L)$ — vuelve atrás hasta un $\bullet$ (pasa todos los símbolos)
$f(q_4,a)=(q_4,a,L)$
$f(q_4,b)=(q_4,b,L)$
$f(q_4,\bullet)=(q_0,\bullet,R)$ — cuando llega a un blanco a la izquierda empieza de nuevo
$f(q_0,1)=(q_5,1,L)$ — cuando en q_0 se lee un 1, pasa al estado final y para

$\Rightarrow$ lee cadenas de $(aab)^n$ y al final deja el número n en la cinta
$L=\{x|\ x\in L((aab)+)\}$

Función computada por una MT

Las MTs pueden transformar entradas en salidas:

- La entrada son todos los símbolos no blancos en la cinta en el momento inicial
- El contenido de la cinta (los símbolos no blancos) al final de la computación (cuando la máquina se para en un estado final) se considera como salida

En otras palabras, se puede considerar una MT como la implementación de una función f: *y=f(x)* si para la configuración inicial q_0x la máquina M para en una configuración q_fy, donde q_f es un estado final de M: $q_0x \vdash_M^* q_fy$

Convenciones:

- Codificación de los números naturales en el sistema unario:

Se representa cada $x \in N$ por una palabra $w(x) \in \{1\}^+$ tal que $|w(x)|-1=x$

(0→"1"; 1→"11"; 2→"111"; 3→"1111" ...)

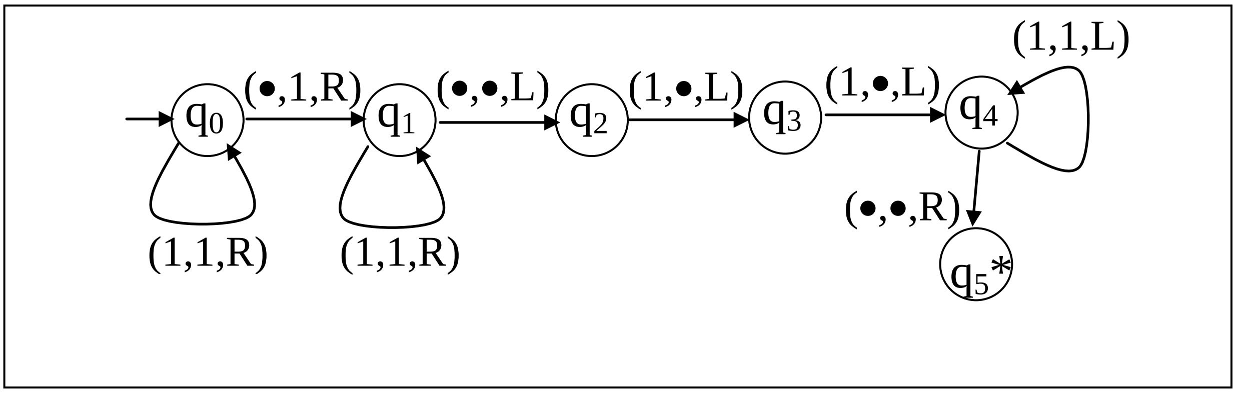
(se pueden usar otros tipos de codificaciones)

- Al inicio y al final de la ejecución la cabeza se encuentra sobre el primer 1 a la izquierda.
- No hay transiciones definidas para estados finales

Ejemplo:

$M= (\{1,\bullet\},\{1\},\bullet,\{q_0,q_1,q_2,q_3,q_4,q_5\},q_0,f,\{q_5\})$

$f(q_0,1)=(q_0,1,R)$	$f(q_2,1)=(q_3,\bullet,L)$
$f(q_0,\bullet)=(q_1,1,R)$	$f(q_3,1)=(q_4,\bullet,L)$
$f(q_1,1)=(q_1,1,R)$	$f(q_4,1)=(q_4,1,L)$
$f(q_1,\bullet)=(q_2,\bullet,L)$	$f(q_4,\bullet)=(q_5,\bullet,R)$



$\bullet q_0111\bullet11\bullet \vdash \bullet1q_011\bullet11\bullet \vdash \bullet11q_01\bullet11\bullet \vdash \bullet111q_0\bullet11\bullet \vdash$
$\bullet1111q_111\bullet \vdash \bullet11111q_11\bullet \vdash \bullet111111q_1\bullet \vdash \bullet11111q_21\bullet \vdash$
$\bullet1111q_31\bullet\bullet \vdash \bullet111q_41\bullet\bullet\bullet \vdash \bullet11q_411\bullet\bullet\bullet \vdash \bullet1q_4111\bullet\bullet\bullet \vdash$
$\bullet q_41111\bullet\bullet\bullet \vdash q_4\bullet1111\bullet\bullet\bullet \vdash \bullet q_51111\bullet\bullet\bullet$

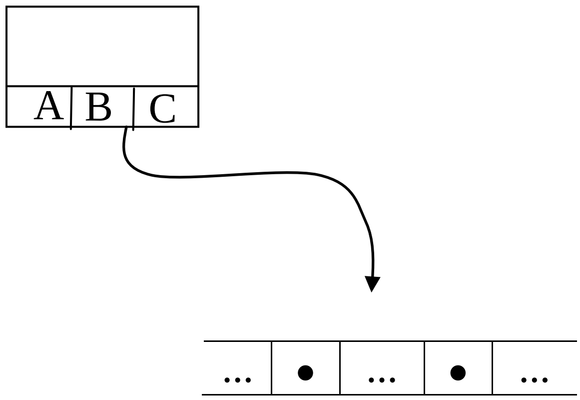
$\Rightarrow q_0111\bullet11 \vdash^* q_51111$

⇒ calcula la suma de dos enteros positivos x+y, donde x e y se representan mediante notación unaria: 1^n corresponde al número n-1 . Es decir, x=0 se representa con "1", x=1 se representa con "11".

13.2. Otras definiciones

Trucos: Almacenamiento en el estado

Una máquina M puede almacenar datos en su unidad de control (y realizar los movimientos en función de estos datos y del estado actual):



(no cambia la definición de la máquina de Turing)

Ejemplo:

$M= (\{0,1,\bullet\},\{1,0\},\bullet,Q,[q_0,\bullet],f,\{[q_2, \bullet]\})$

$Q=\{[q_0,\bullet], [q_1,1], [q_1,0], [q_2,\bullet]\} \subseteq\{q_0,q_1,q_2\}\times\{0,1, \bullet\}$

(los estados son tuplas [estado, dato])

$f([q_0,\bullet],0)= ([q_1,0], \bullet,R)$	-se almacena el primer símbolo
$f([q_0,\bullet],1)= ([q_1,1], \bullet,R)$	en la segunda posición del estado
$f([q_1,0],1)= ([q_1,0], \bullet,R)$	- se busca otro símbolo que es
$f([q_1,1],0)= ([q_1,1], \bullet,R)$	igual que el primero
$f([q_1,0],0)= ([q_2,\bullet],0,L)$	- si encuentra el mismo símbolo
$f([q_1,1],1)= ([q_2,\bullet],1,L)$	va al estado de parada

$[q_0,\bullet]01011 \vdash [q_1,0]1011 \vdash [q_1,0]011 \vdash [q_2,\bullet]\bullet 011$

Trucos: Separar la cinta en tramos o pistas:

Se puede considerar que la cinta de una máquina M tiene varias pistas o tramos (para almacenar distintos datos).

...	X		...		...		
...	Y		...		...		...

Idea: se considera los símbolos de la cinta como tuplas [X,Y]

Ejemplo:
usar el primer tramo para marcar celdas en el segundo tramo

...	●	●	*	●	●	●	...
...	●	a	B	...	b	●	...

- marcar la primera b (alfabeto de entrada es {a,b})

$f(q_0,[\bullet,a]) = (q_0,[\bullet,a],R)$
$f(q_0,[\bullet,b]) = (q_1,[*,b],R)$

●●●●●　　●●●●　　●●*
q_0a...ab... ⊢* a...aq_0b... ⊢ a...abq_1...

- buscar la celda marcada, cópiala al dato almacenado en el estado y quita la marca (alfabeto de entrada es {a,b})

$f([q_0,\bullet],[\bullet,a]) = ([q_0,\bullet],[\bullet,a],R)$
$f([q_0,\bullet],[\bullet,b]) = ([q_0,\bullet],[\bullet,b],R)$
$f([q_0,\bullet],[*,b]) = ([q_1,b],[\bullet,b],R)$
$f([q_0,\bullet],[*,a]) = ([q_1,a],[\bullet,a],R)$

●●*●　　●●　　*●　　●●●　　●
$[q_0,\bullet]$ aabb... ⊢* aa$[q_0,\bullet]$ bb... ⊢ aab$[q_1,b]$b...

Equivalencia de máquinas secuenciales

Sean M^1 y M^2 dos máquinas secuenciales. Se dice que M^1 y M^2 son *equivalentes* (se escribe $M^1 \equiv M^2$) si y sólo si realizan la misma función, es decir, para cada entrada posible w:

- M^2 para en un estado final si y solo si M^1 para en un estado final (y con la misma salida – contenido de cinta)
- si M^1 no para nunca o para en un estado que no es final también M^2 no para nunca o para en un estado no final (y al revés)

Dos clases de máquinas secuenciales CMS^1 y CMS^2 son *equivalentes*, si y solo si

1. para cada $M \in CMS^1$ existe $M' \in CMS^2$ tal que $M \equiv M'$ y
2. para cada $M \in CMS^2$ existe $M' \in CMS^2$ tal que $M \equiv M'$

Hay otros modelos de máquinas de Turing.

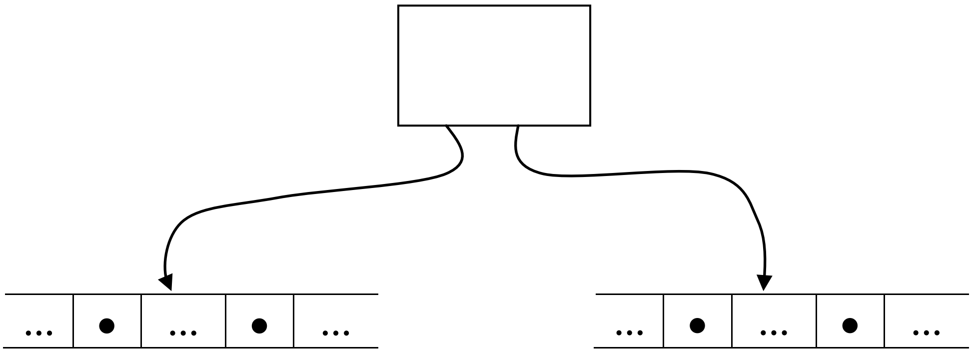
Todos ellos son equivalentes al modelo presentado (máquinas estándar), es decir: 1) cualquier máquina estándar puede ser *simulada* por una máquina de estos modelos, y 2) cualquier máquina de estos modelos puede ser simulada por una máquina estándar.

→ ningún modelo es más potente que el modelo estándar

Veremos algunos modelos y analizaremos

1. Cómo se pueden simular máquinas estándar en estos modelos
2. Cómo se pueden simular sus máquinas por máquinas estándar

Máquinas con dos (o más) cintas infinitas



1. Cada máquina estándar es una máquina con dos cintas que no usa la segunda cinta

2. Simular la máquina M de dos cintas en una máquina N con una cinta:
- M transita en función del estado actual y de los dos símbolos leídos (f(q,a,b)=(p,e,c,L,R)) y escribe dos nuevos símbolos en las cintas y mueve las dos cabezas de lectura/escritura
- ¿Cómo se pueden simular los movimientos de la máquina M?

Idea:

Representar el contenido de dos (o más) cintas en la máquina N en una cinta con varios tramos o pistas.

...	•	i_0	i_1	i_2	i_3	i_4	i_5	i_6	•	...	Tramo1
...	•	1	•	•	•	•	•	•	•	...	
...	•	k_0	k_1	k_2	k_3	•	•	•	•	...	Tramo2
...	•	1	•	•	•	•	•	•	•	...	

Los estados de la máquina N tienen la forma [q, t_1,t_2,s_1,s_2] y almacenan:

- q: el estado actual de la máquina M

- t_1 y t_2: los símbolos leídos en las pistas 1 y 2
- s_1 y s_2: unos símbolos que indican en que dirección de la cabeza se encuentran las marcas para el tramo 1 y 2 (L – izquierda, R - derecha, • - en esta)

Inicialmente N arranca con la cinta dada arriba en el estado [q_0,•,•,•,•] (q_0 es el estado inicial de M) y sucesivamente hace los mismos movimientos que la máquina M.
Para simular un movimiento de M, N hace lo siguiente:

- en función de s_1 busca la marca para el tramo 1 y copia el símbolo marcado en t_1 (deja este símbolo y su marca)
- hace lo mismo para el tramo 2
- suponemos que N está en estado [r,a,b,?,?] entonces se simula el movimiento f(r,a,b)=(p,e,c,X,Y) de M
    - busca el símbolo marcado en el tramo 1, lo cambia por la e
    - quita la marca de este tramo y lo pone en la posición que indica X (a la derecha o a la izquierda)
    - hace lo mismo para el tramo 2 (cambiando b por c)
    - cambia el estado de r a p (si p es final -> para)

Ha realizado el movimiento y empieza de nuevo.

Importante: en todo momento tiene que actualizar s_1 y s_2

Máquinas no deterministas

Máquinas no deterministas: $M=(\Gamma,\Sigma,\bullet,Q,q_0,f,F)$
f es aplicación de $Q\times\Gamma$ en el conjunto de las partes $Q\times\Gamma\times\{I,D\}$

1. Cada máquina estándar es también una máquina no determinista (que no usa el concepto de no determinismo)

2. Simulación de una máquina no determinista M con una determinista N:

Idea: f(q,a)={(p,b,R), (r,c,L)}

1. para cada posibilidad distinta crea un nuevo par de tramos en la cinta

...	•	a	b	b	a	•	...	Camino1
...	•	q	•	•	•	•	...	
...	•	a	b	b	a	•	...	Camino2
...	•	q	•	•	•	•	...	

2. realiza en cada par de tramos un movimiento

...	•	b	b	b	a	•	...	Camino1
...	•	•	p	•	•	•	...	
...	•	c	b	b	a	•	...	Camino2
...	r	•	•	•	•	•	...	

3. Si uno de los estados alcanzados es final, entonces para.
4. En caso contrario repite los pasos 1 y 2 para todas las pares de tramos que haya

La máquina N simula una búsqueda en amplitud.

Obviamente, esta N tiene un "overhead" (que no se ha especificado).

13.3. MT para reconocer lenguajes

Ejemplo 1

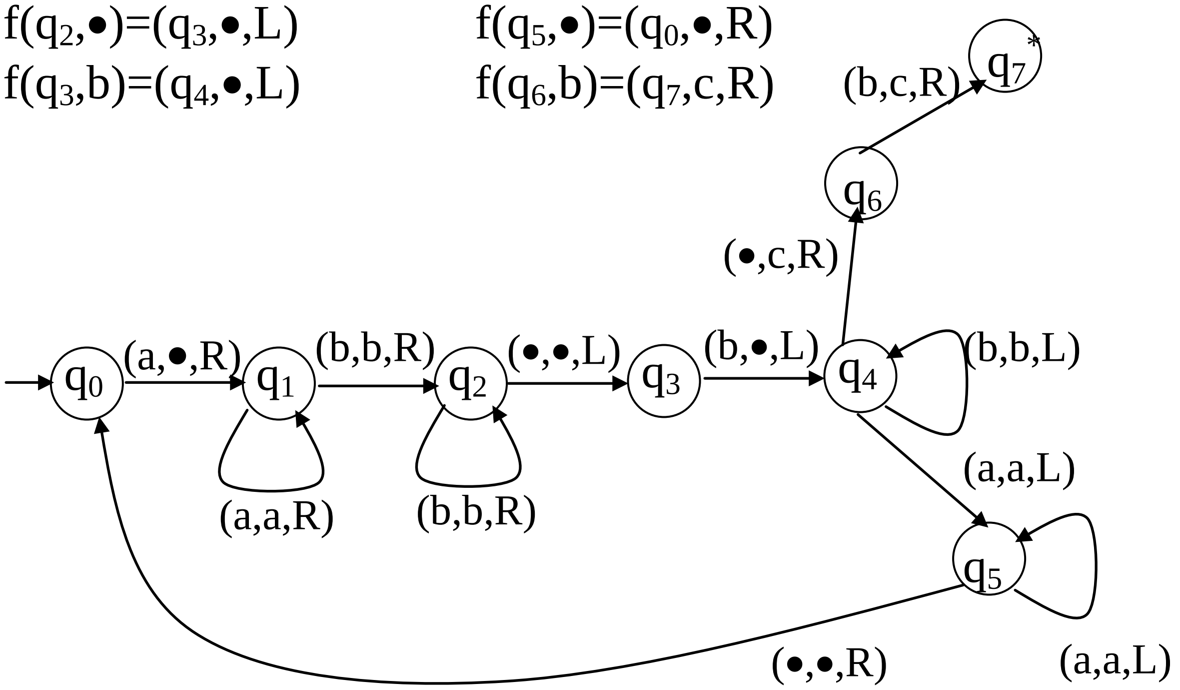
Alfabeto: $\Sigma=\{0,1\}$ Lenguaje sobre Σ: L(0*)
Construimos $M=(\{\bullet,1,0\},\{1,0\},\bullet,\{q_0,q_1\},q_0,f,\{q_1\})$, donde
$f(q_0,0)=(q_0,0,R)$
$f(q_0,\bullet)=(q_1,\bullet,R)$

$q_0000 \vdash 0q_000 \vdash 00q_00 \vdash 000q_0\bullet \vdash 000\bullet q_1\bullet$ (para en est. final)
$q_0\bullet \vdash q_1\bullet$ (para en est. final)
$q_001 \vdash 0q_01$ (para en estado que no es final)

Ejemplo 2

Lenguaje sobre $\Sigma=\{a,b,c\}$: $L=\{a^nb^m \mid m>n,\ n>0\}$
Construimos $M=(\{\bullet,a,b,c\},\{a,b,c\},\bullet,\{q_0,...,q_7\},q_0,f,\{q_7\})$, donde

$f(q_0,a)=(q_1,\bullet,R)$	$f(q_4,b)=(q_4,b,L)$
$f(q_1,a)=(q_1,a,R)$	$f(q_4,a)=(q_5,a,L)$
$f(q_1,b)=(q_2,b,R)$	$f(q_4,\bullet)=(q_6,\bullet,R)$
$f(q_2,b)=(q_2,b,R)$	$f(q_5,a)=(q_5,a,L)$
$f(q_2,\bullet)=(q_3,\bullet,L)$	$f(q_5,\bullet)=(q_0,\bullet,R)$
$f(q_3,b)=(q_4,\bullet,L)$	$f(q_6,b)=(q_7,c,R)$



Ejemplos: $a^n b^{n+2}$, aab y ab

$q_0$aa...ab...bbbb ├q_1a...ab...bbbb ├*a...aq_1b...bbbb ├a...abq_2...bbbb ├* a...ab...bbbbq_2● ├a...ab...bbbq_3b ├a...ab...bbq_4b ├*a...$q_4$ab...bbb ├ a...$q_5$aab...bbb ├*q_5●a... ab...bbb ├q_0a... ab...bbb ├ ....
├ $q_0$abbb ├ $q_1$bbb ├ b$q_2$bb ├* bbbq_2● ├ bbq_3b ├ bq_4b ├*q_4●bb ├c$q_6$bb ├ ccq_7b

$q_0$aab ├$q_1$ab ├aq_1b ├abq_2● ├aq_3b ├q_4a● ├q_5●a ├q_0a ├ q_1●

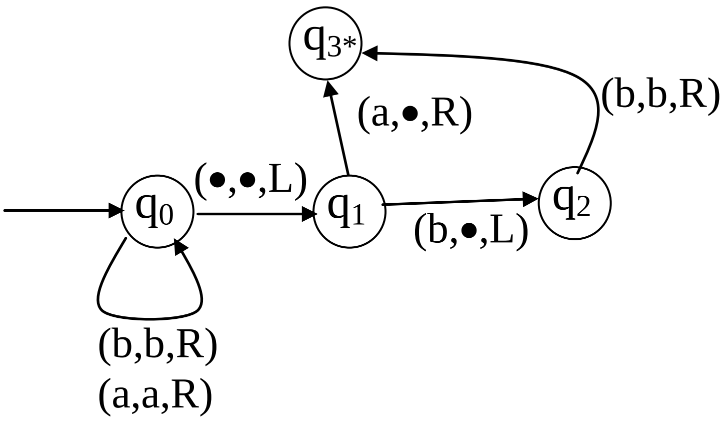
$q_0$ab ├q_1b ├bq_2● ├q_3b ├q_4● ├cq_6●

<u>Gramática equivalente a una MT</u>

Ejemplo:

M= ({a,b,●},{a,b},●,{q_0,q_1,q_2, q_3},q_0,f,{q_3})

f(q_0,b)=(q_0,b,R)	f(q_1,b)=(q_2,●,L)
f(q_0,a)=(q_0,a,R)	f(q_1,a)=(q_3,●,R)
f(q_0,●)=(q_1,●,L)	f(q_2,b)=(q_3,b,R)



Acepta el lenguaje descrito por (a+b)*(a+bb)

Se acepta una palabra w si existe: q_0w ├*xq_2y (x,y∈Γ*).

Sólo hay una secuencia de movimientos para aceptar w∈L(M):

Ejemplo:
$q_0babb \vdash bq_0abb \vdash baq_0bb \vdash babq_0b \vdash babbq_0\bullet \vdash babq_1b \vdash baq_2b \vdash babq_3$

Se puede simular esta secuencia con una gramática.

- Consideramos que solo existe un estado final q_k y no hay transiciones definidas para él
- Suponemos que tenemos una cadena "$w[q_0w]$" siendo w cualquier palabra del alfabeto de entrada.
- La idea es simular los movimientos de M en la segunda repetición de la palabra, con reglas de una gramática: $w[q_0w] \rightarrow^* w[xq_iy]$ siendo x, y $\in\Gamma^*$:
  - Si $q_i = q_k$ (estado final) deben existir derivaciones para:

  $w[xq_ky] \rightarrow w[xq_ky] \rightarrow^* w[xq_k] \rightarrow^* w[q_k] \rightarrow w$

  Se deja (genera) la palabra w, ya que sobre ella la máquina M llega a un estado de aceptación.

Creamos las reglas:
1. para simular el funcionamiento de M:

diferentes movimientos:		Reglas:
f(q,b)=(p,c,R)	qb... $\vdash$ cp...	qb::=cp
f(q,b)=(p,c,L)	xqb... $\vdash$ pxc...	xqb::=pxc (para todo $x\in\Gamma$)

se añaden:
$[q_i$::=$[\bullet q_i$ $\quad$ $q_i]$::= $q_i\bullet]$ para todo estado q_i de M que no es final

2. si se llega al estado final q_k hay que dejar solo w:
q_kx::= q_k (para todo $x\in\Gamma$)
xq_k::= q_k (para todo $x\in\Gamma$)
$[q_k]$::=λ

Ejemplo:

M= ({a,b,•},{a,b},•,{q_0,q_1,q_2, q_3},q_0,f,{q_3})

Reglas de simulación:

f(q_0,b)=(q_0,b,R) q_0b::= bq_0
f(q_0,a)=(q_0,a,R) q_0a::= aq_0
f(q_0,•)=(q_1,•,L) •q_0•::= q_1•• aq_0•::= q_1a• bq_0•::= q_1b•
f(q_1,b)=(q_2,•,L) •q_1b::= q_2•• aq_1b::= q_2a• bq_1b::= q_2b•
f(q_1,a)=(q_3,•,R) q_1a::= •q_3
f(q_2,b)=(q_3,b,R) q_2b::= bq_3

[q_0::= [•q_0 q_0]::= q_0•] [q_1::= [•q_1 q_1]::= q_1•]
[q_2::= [•q_2 q_2]::= q_2•]

(sólo hay como mucho una regla para cada par q_ix y ninguna para el estado final)

Reglas de eliminación de q_3

q_3a::= q_3 q_3b::= q_3 q_3•::= q_3
aq_3::= q_3 bq_3::= q_3 •q_3::= q_3
[q_3]::=λ

Ejemplo:

1) $q_0$abb ├* abq_3
abb[$q_0$abb]$\rightarrow$* abb[abq_3••]$\rightarrow$* abb

2) $q_0$bab ├ b$q_0$ab ├ baq_0b ├ babq_0• ├ baq_1b ├ bq_2a
bab[$q_0$bab]$\rightarrow$* bab[bq_2a••]

Resumen:

- Si w es una palabra aceptada por la máquina M, entonces estas reglas, aplicadas a w[q_0w], generan w.
- En caso contrario no se genera nunca una palabra (las cadenas siempre contienen un símbolo que corresponde al estado de M y no pertenece al alfabeto)

Para terminar la gramática, solo falta la parte que genere $w[q_0w]$ para cualquier palabra w posible sobre el alfabeto de entrada de la máquina de Turing:

$\Sigma_T=\{a,b\}$ $\Sigma_N=\{S,A,B,q_0,[,],!\}$ Axioma: S
S::=a!a] | b!b] | $[q_0]$
!::=a!A | b!B
Aa::=aA Ba ::=aB Ab::=bA Bb::=bB
A]::=a] B]::=b]
!::=$[q_0$
Esta gramática genera cadenas $x[q_0x]$ siendo x cualquier posible palabra sobre el alfabeto {a,b}

S→a!a] →aa!Aa] →aab!BAa] genera dos palabras invertidas
→aab!BaA]→aab!Baa]→aab!aBa]→aab!aaB]→aab!aab]
→aab$[q_0$aab]

- se puede generar $w[q_0w]$ para cualquier $w\in\{a,b\}^*$
  - aplicando a estas palabras las reglas de antes se genera w si y solo si w es aceptado por la máquina M
- también se pueden generar cadenas como:
  - S→* abba!abba] : en este caso las reglas de antes no hacen nada (dejan las variables ! y])
  - S→* abb$[q_0$BBa] o S→* abb$[q_0$aBb]: en este caso las reglas de antes no pueden eliminar nunca la variable B

⇒ Se obtiene la gramática $G=(\Sigma_T, \Sigma_N,S,P)$ correspondiente a la máquina de Turing M= $(\{a,b,\bullet\},\{a,b\},\bullet,\{q_0,q_1,q_2,q_3\},q_0,f,\{q_3\})$ juntando los dos conjuntos de reglas:

$\Sigma_T=\Sigma=\{a,b\}$ $\Sigma_N=\{A,B,S,\bullet, [,],!,q_0,q_1,q_2,q_3\}$

Algoritmo para obtener la gramática que genera el lenguaje aceptado por una MT

Sea la máquina de Turing M= $(\Gamma,\Sigma,\bullet,Q,q_0,f,F)$ que acepta el lenguaje L(M) y sea $F=\{q_f\}$ (si hay más estados finales la convertimos en una máquina equivalente con un estado final). A partir de ella construimos la gramática $G=(\Sigma_T, \Sigma_N, S, P)$ donde

- $\Sigma_T=\Sigma$
- $\Sigma_N=\{S,\bullet,[,],!\} \cup Q \cup R$
  - R=conjunto de variables correspondientes a Σ
    - para cada $x\in\Sigma$ existe una variable única X en R
  - se supone que $S,[,],!\notin\Gamma$ y que $R\cap\Gamma=\emptyset$

El conjunto de producciones P es :
(para generar $w[q_0w]$):

- $S{:}{=}[q_0]$
- $S::=x!x]$ para cada $x\in\Sigma$
- $!::=x!X$ para cada $x\in\Sigma$ y su variable $X\in R$ correspondiente
- $Xy::=yX$ para cada $y\in\Sigma$ y $X\in R$
- $X]::=x]$ para cada $x\in\Sigma$ y su variable $X\in R$ correspondiente
- $!::=[q_0$

(para simular la máquina M):

- para cada transición $f(q_i,b)=(q_j,c,R)$: $q_ib::=cq_j$
- para cada transición $f(q_i,b)=(q_j,c,L)$: $xq_ib::= q_jxc$ (para todo $x\in\Gamma$)
- $[q_i::=[\bullet q_i$ para todo $q_i\in Q$ y $q_i\neq q_f$
- $q_i]::= q_i\bullet]$ para todo $q_i\in Q$ y $q_i\neq q_f$

(para dejar la palabra final):

- $q_fx::= q_f$ para todo $x\in\Gamma$
- $xq_f::= q_f$ para todo $x\in\Gamma$
- $[q_f]::=\lambda$

$\Rightarrow$G es del tipo 0 y se puede demostrar que L(G)=L(M).

MT equivalente a una gramática de tipo 0

Sea la gramática $G=(\Sigma_T, \Sigma_N, S, P)$.
Construiremos una máquina de Turing $M=(\Gamma,\Sigma,\bullet,Q,q_0,f,F)$ que acepta $L(G)$.

La demostración no es muy complicada, pero muy larga.

Idea:

- Crear una máquina con $\Sigma_T \cup \Sigma_N \cup \{\lambda\} \subseteq \Gamma$, $\Sigma=\Sigma_T$, $F=\{q_1\}$.

Para aceptar una palabra $x \in L(G)$, esta máquina debe parar en q_1

- Usar una máquina M con varios tramos en la cinta:
  - uno para la palabra inicial
  - en el resto de los tramos se exploran las posibles derivaciones de palabras con G
- Inicialmente, la M tiene un tramo de exploración que contiene el axioma.
- En cada paso, la máquina hace lo siguiente:
  Repetir {
    Para cada tramo de exploración:
    - si se pueden aplicar varias reglas de producción, para cada una de ellas menos la primera:
      - copia el contenido del tramo a un tramo nuevo
      - aplica la regla al contenido del nuevo tramo
    - para la primera regla aplicable (o si solo se puede aplicar una regla) la aplica en el tramo actual

  }
  Después de aplicar una regla siempre comprueba si el tramo contiene la palabra del tramo 1
    - Si es así, para en q_1, en caso contrario sigue

Ejemplo:

S::=AB1|A0AB

...	•	a	b	b	a	•	...
...	•	S	•	•	•	•	...

Tramo1: la palabra x
Tramo de exploración: S

⇓

...	•	a	b	b	a	•	...
...	•	A	B	1	•	•	...
...	•	A	0	A	B	•	...

Tramo1: la palabra x
Tramo de exploración 1
Tramo de exploración 2

⇓

Ejemplo:

G=(Σ_T, Σ_N, S, P) con

P={S::=AB1|A0AB; B::=00|λ; A001::= λ; A0::=00; A1::=11}

Inicio: tramo1=0001
tramo2=S

Pasos:

1. tramo2.1= S→AB1 AB1≠0001→ sigue
   tramo2.2= S→A0AB A0AB≠0001→ sigue

2. tramo2.1.1= AB1→A001 A001≠0001 → sigue
   tramo2.1.2= AB1→A1 A1≠0001 → sigue
   tramo2.2.1= A0AB→A0A00 A0A00≠0001 → sigue
   tramo2.2.2= A0AB→A0A A0A ≠0001 → sigue
   tramo2.2.3= A0AB→00AB 00AB ≠0001 → sigue

3. tramo2.1.1.1= A001→λ λ ≠0001 → sigue
   tramo2.1.1.2= A001→0001 0001=0001→ pasa a q_1 y para

13.4. MT para computar funciones

Funciones de un parámetro

Ejemplo:

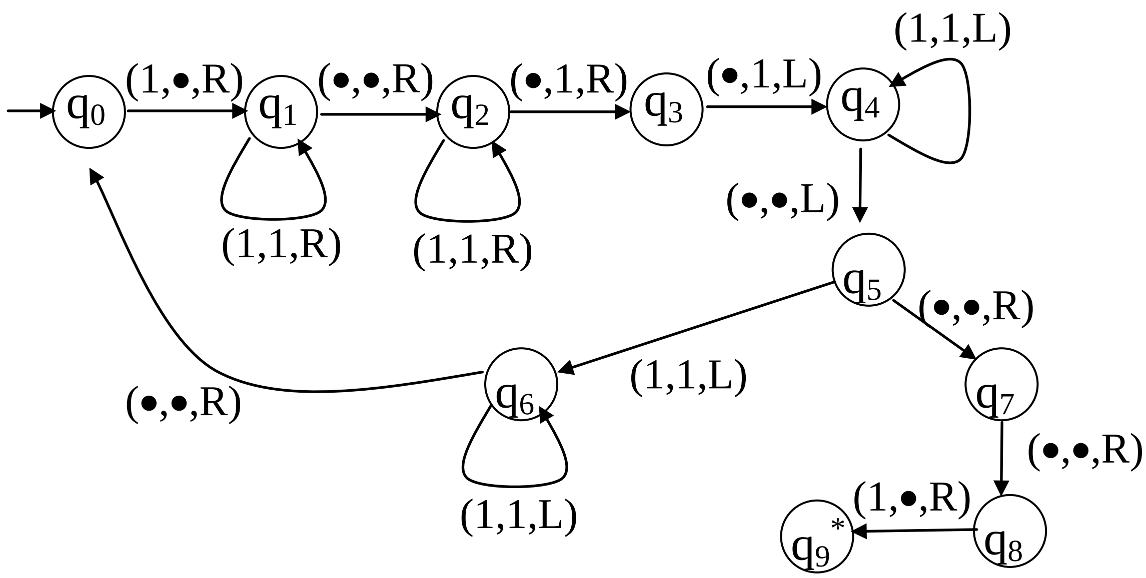
Realiza mediante una MT la función y=f(x)=x*2.

Es decir, construye una MT con $q_0x \vdash^* q_fy$:

- donde q_0 es el estado inicial y q_f un estado final
- $x,y \in \{1\}^+$ e |y|-1=2(|x|-1) , es decir, |y|=2|x|-1

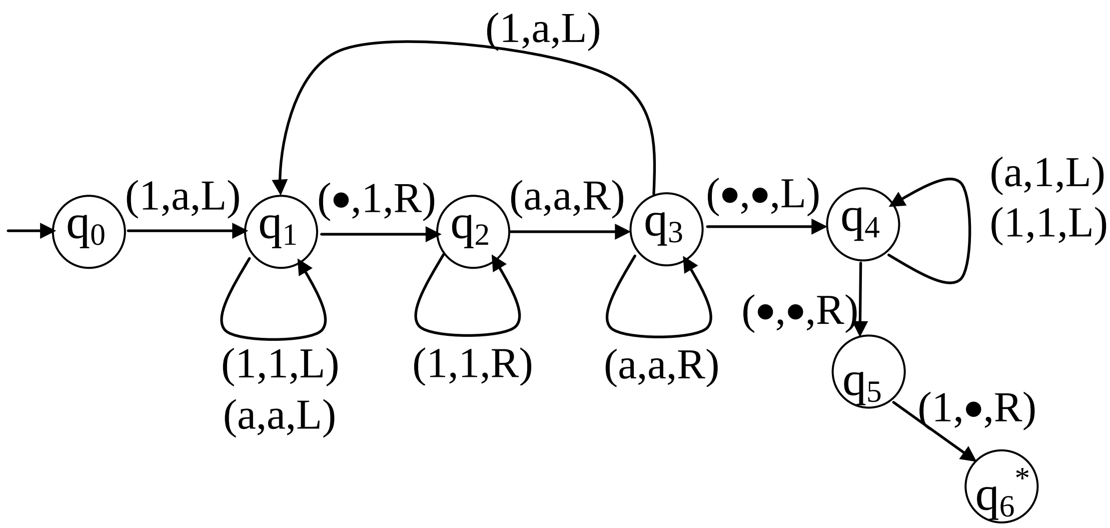
Idea 1:

- Para cada 1 que encontramos escribimos dos 1’s nuevos detrás del primer • y el 1 lo eliminamos.
- Al final quitamos un 1



Idea 2:

- Para cada 1 escribimos otro 1 a la izquierda de la cadena. Para eso tenemos que modificar temporalmente los 1's leídos (por a's).
- Al final cambiamos las a's por 1's y quitamos un 1



Funciones de varios parámetros

Convenciones:

- Las cadenas de 1's correspondientes a los parámetros están separadas por un único símbolo 0
- Al inicio y al final la cabeza se encuentra sobre el primer 1 del primer bloque de 1's (desde la izquierda).

Ejemplo:

Realiza mediante una MT la función $f(x,y) = \begin{cases} 0, \text{ si x} \leq \text{y} \\ 1, \text{ si x} > \text{y} \end{cases}$.

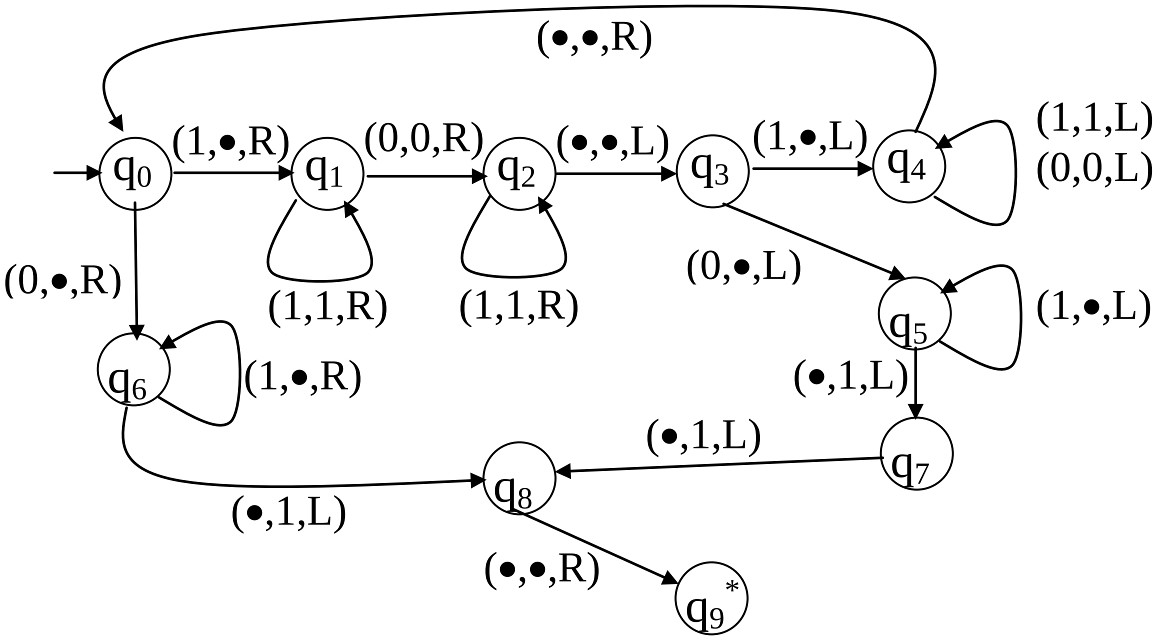
Es decir, construye una MT con

$q_0x0y \vdash^* q_f1$, si $x \leq y$

$q_0x0y \vdash^* q_k11$, si $x > y$

donde q_0 es el estado inicial y q_f, q_k son estados finales.

Idea: Comparamos cada 1 de x (al principio) con cada 1 de y (al final). Los 1's ya comparados los quitamos.



Funciones complejas

Idea: combinar varias máquinas de Turing

Ejemplo:

"if x≤y then A else B"

Sean A y B unas MTs que calculan las funciones correspondientes a A y B.

$A=(\Gamma_A,\Sigma_A,\bullet,Q_A,q_{0A},f_A,F_A)$

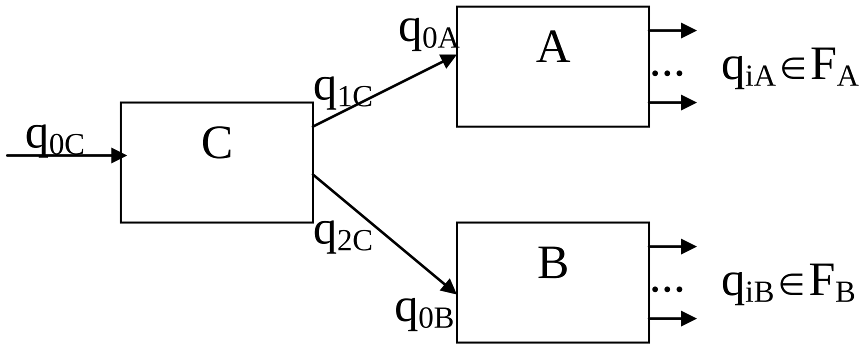
$B=(\Gamma_B,\Sigma_B,\bullet,Q_B,q_{0B},f_B,F_B)$

Y sea $C=(\Gamma_C,\Sigma_C,\bullet,Q_C,q_{0C},f_C,\{q_{1C},q_{2C}\})$ una máquina que realiza lo siguiente:

$q_{0C}x\bullet y \vdash^* q_{1C}x\bullet y$ si $x\leq y$

$q_{0C}x\bullet y \vdash^* q_{2C}x\bullet y$ en caso contrario

Podemos combinar las máquinas C, A y B obteniendo una máquina que calcula “if x≤y then A else B”.



En vez de terminar en sus estados finales, C termina en los estados iniciales de A y B.