Máquinas de Turing

Teoría de la Computación

Grado en Ingeniería Informática

Contenidos

Contenidos

1 Definiciones básicas

Los lenguajes regulares son un subconjunto propio de los LIC,s. Por tanto, los AP,s son más potentes que los AF,s con respecto a la capacidad de aceptar lenguajes. Aún así, todavía existen muchos otros lenguajes que no son LIC,s (por ejemplo, $\{a^n b^n c^n \mid n \geq 0\}$ o $\{ww \mid w \in \Sigma^*\}$).

1 Definiciones básicas

Los lenguajes regulares son un subconjunto propio de los LIC,s. Por tanto, los AP,s son más potentes que los AF,s con respecto a la capacidad de aceptar lenguajes. Aún así, todavía existen muchos otros lenguajes que no son LIC,s (por ejemplo, $\{a^n b^n c^n \mid n \geq 0\}$ o $\{ww \mid w \in \Sigma^*\}$).

Por tanto, en este capítulo estudiaremos un nuevo dispositivo reconocedor de lenguajes, la **máquina de Turing**, más potente que los autómatas anteriores, aunque bastante similar a ellos con respecto a sus componentes y a las acciones que realiza.

1 Definiciones básicas

Los lenguajes regulares son un subconjunto propio de los LIC,s. Por tanto, los AP,s son más potentes que los AF,s con respecto a la capacidad de aceptar lenguajes. Aún así, todavía existen muchos otros lenguajes que no son LIC,s (por ejemplo, $\{a^n b^n c^n \mid n \geq 0\}$ o $\{ww \mid w \in \Sigma^*\}$).

Por tanto, en este capítulo estudiaremos un nuevo dispositivo reconocedor de lenguajes, la **máquina de Turing**, más potente que los autómatas anteriores, aunque bastante similar a ellos con respecto a sus componentes y a las acciones que realiza.

Cuando pasamos de los AF,s a los AP,s introdujimos la pila como mecanismo para *recordar* la información necesaria para el reconocimiento de los LIC,s. Sin embargo, el uso de la pila es muy restrictivo, ya que sólo tenemos acceso al elemento de la cima. Para reconocer, por ejemplo, $\{a^n b^n c^n \mid n \geq 0\}$, si apilamos las *a*es y las desapilamos a medida que llegan las *b*es, habremos perdido la información necesaria para poder contar las *c*es. Así pues, el problema subyacente no es la carencia de memoria, sino la forma en la que está organizada.

1 Definiciones básicas

Los lenguajes regulares son un subconjunto propio de los LIC,s. Por tanto, los AP,s son más potentes que los AF,s con respecto a la capacidad de aceptar lenguajes. Aún así, todavía existen muchos otros lenguajes que no son LIC,s (por ejemplo, $\{a^n b^n c^n \mid n \geq 0\}$ o $\{ww \mid w \in \Sigma^*\}$).

Por tanto, en este capítulo estudiaremos un nuevo dispositivo reconocedor de lenguajes, la **máquina de Turing**, más potente que los autómatas anteriores, aunque bastante similar a ellos con respecto a sus componentes y a las acciones que realiza.

Cuando pasamos de los AF,s a los AP,s introdujimos la pila como mecanismo para *recordar* la información necesaria para el reconocimiento de los LIC,s. Sin embargo, el uso de la pila es muy restrictivo, ya que sólo tenemos acceso al elemento de la cima. Para reconocer, por ejemplo, $\{a^n b^n c^n \mid n \geq 0\}$, si apilamos las $a$es y las desapilamos a medida que llegan las $b$es, habremos perdido la información necesaria para poder contar las $c$es. Así pues, el problema subyacente no es la carencia de memoria, sino la forma en la que está organizada.

La nueva organización de memoria introducida por las máquinas de Turing consiste en una colección de celdas que se extiende infinitamente en ambas direcciones. Es decir, esencialmente se trata de una **cinta**. Cada celda almacena un único símbolo. No existe primera ni última celda, por lo que la capacidad de almacenamiento es ilimitada. A los contenidos de las celdas se puede acceder en cualquier orden. Para ello, la máquina de Turing tiene asociada una **cabeza de lectura/escritura** que puede moverse sobre la cinta en cualquier dirección, y por cada movimiento leerá y escribirá un símbolo.

1 Definiciones básicas

Definición

Una **máquina de Turing** o **MT** M consta de siete elementos, $M = (Q, \Sigma, \Gamma, s, B, F, \delta)$, donde:

- Q es un conjunto finito de estados.
- Σ es el alfabeto de los símbolos de entrada.
- Γ es el alfabeto de los símbolos de la cinta.
- $s \in Q$ es el estado inicial de la máquina.
- $B \in \Gamma$ es el símbolo blanco (se asume que $B \notin \Sigma$, y se asume también que las celdas de la cinta que no contengan símbolos de la cadena a procesar, o símbolos resultado de las operaciones de la máquina, estarán inicializadas a este símbolo blanco).
- $F \subseteq Q$ es el conjunto de estados finales o de aceptación.
- $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ es la función de transición que, a partir del estado actual y del símbolo de la celda apuntada por la cabeza de L/E, determina el nuevo estado, el nuevo símbolo que se escribe en esa celda y el movimiento de la cabeza: L (*left*) o R (*right*).

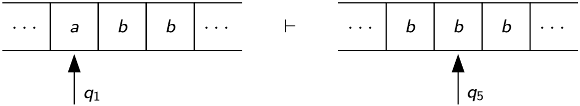
1 Definiciones básicas

Definición

Una **máquina de Turing** o **MT** M consta de siete elementos, $M = (Q, \Sigma, \Gamma, s, B, F, \delta)$, donde:

- Q es un conjunto finito de estados.
- Σ es el alfabeto de los símbolos de entrada.
- Γ es el alfabeto de los símbolos de la cinta.
- $s \in Q$ es el estado inicial de la máquina.
- $B \in \Gamma$ es el símbolo blanco (se asume que $B \notin \Sigma$, y se asume también que las celdas de la cinta que no contengan símbolos de la cadena a procesar, o símbolos resultado de las operaciones de la máquina, estarán inicializadas a este símbolo blanco).
- $F \subseteq Q$ es el conjunto de estados finales o de aceptación.
- $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ es la función de transición que, a partir del estado actual y del símbolo de la celda apuntada por la cabeza de L/E, determina el nuevo estado, el nuevo símbolo que se escribe en esa celda y el movimiento de la cabeza: L (*left*) o R (*right*).

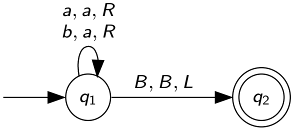
Es decir, si $\delta(q_1, a) = (q_5, b, R)$, la configuración de una MT podría evolucionar así:



Por ejemplo, consideremos la siguiente MT:

$$
\begin{array}{lll}
Q = \{q_1, q_2\} & s = q_1 & \delta(q_1, a) = (q_1, a, R) \\
\Sigma = \{a, b\} & F = \{q_2\} & \delta(q_1, b) = (q_1, a, R) \\
\Gamma = \{a, b, B\} & B = B & \delta(q_1, B) = (q_2, B, L)
\end{array}
$$

Esta máquina, al procesar una cadena de $\{a, b\}^*$, reemplaza todas las $b$es por $a$es y queda posicionada en el último símbolo de dicha cadena, y puede representarse también mediante el siguiente grafo:

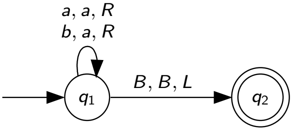


1 Definiciones básicas

Por ejemplo, consideremos la siguiente MT:

$$
\begin{array}{lll}
Q = \{q_1, q_2\} & s = q_1 & \delta(q_1, a) = (q_1, a, R) \\
\Sigma = \{a, b\} & F = \{q_2\} & \delta(q_1, b) = (q_1, a, R) \\
\Gamma = \{a, b, B\} & B = B & \delta(q_1, B) = (q_2, B, L)
\end{array}
$$

Esta máquina, al procesar una cadena de $\{a, b\}^*$, reemplaza todas las *b*es por *a*es y queda posicionada en el último símbolo de dicha cadena, y puede representarse también mediante el siguiente grafo:



Para representar las *configuraciones instantáneas* de una MT, tenemos dos posibilidades. La primera de ellas indica la posición de la cabeza de lectura/escritura mediante un subrayado:

$$(q_1, \underline{a}bba) \vdash (q_1, a\underline{b}ba) \vdash (q_1, aa\underline{b}a) \vdash (q_1, aaa\underline{a}) \vdash (q_1, aaaa\underline{B}) \vdash (q_2, aaa\underline{a})$$

La segunda lo hace insertando el estado actual antes del símbolo apuntado por la cabeza:

$$(q_1 abba) \vdash (aq_1 bba) \vdash (aaq_1 ba) \vdash (aaaq_1 a) \vdash (aaaaq_1 B) \vdash (aaaq_2 a)$$

Las notaciones $\overset{*}{\vdash}$ y $\overset{+}{\vdash}$ tienen el significado usual. Y si la MT fuera no determinista, las configuraciones instantáneas involucrarían a varios elementos como los arriba indicados.

1 Definiciones básicas

Cuando $\delta(q_i, \sigma)$ no está definido, la MT no puede cambiar de configuración, y se dice que está parada. Puede que $q_i \in F$ y puede que no. Por tanto, nos gustaría dotar de significado a las paradas en estados de F. De hecho, para simplificar, supondremos que la MT se parará siempre que llegue a un estado final.

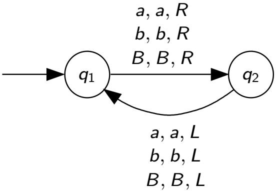
En cualquier caso, la secuencia de movimientos que conducen a una configuración de parada se denomina **computación**.

1 Definiciones básicas

Cuando $\delta(q_i, \sigma)$ no está definido, la MT no puede cambiar de configuración, y se dice que está parada. Puede que $q_i \in F$ y puede que no. Por tanto, nos gustaría dotar de significado a las paradas en estados de F. De hecho, para simplificar, supondremos que la MT se parará siempre que llegue a un estado final.

En cualquier caso, la secuencia de movimientos que conducen a una configuración de parada se denomina **computación**.

Sin embargo, consideremos la siguiente MT:



En esta MT, al procesar una cadena cualquiera, ocurrirá siempre lo siguiente:

$$q_1 ab \cdots \vdash aq_2 b \cdots \vdash q_1 ab \cdots \vdash aq_2 b \cdots \vdash q_1 ab \cdots \vdash \cdots$$

Es decir, se trata de una MT que nunca parará. Esta situación es muy importante en la teoría de máquinas de Turing y podemos representarla así:

$$q_1 ab \cdots \overset{*}{\vdash} \infty$$

Contenidos

2 Máquinas de Turing como aceptadoras de lenguajes

Una MT se puede comportar como un aceptador de lenguajes igual que los AF,s y AP,s. Colocaremos una cadena w en la cinta, la cabeza de lectura/escritura sobre el primer símbolo de la cadena, la MT en el estado inicial, y la pondremos en marcha. La cadena w es aceptada si después de una secuencia de movimientos la MT llega a un estado final (y por tanto se detiene), incluso aunque la cadena no haya sido procesada totalmente

Definición

El **lenguaje aceptado por una MT** M se define entonces como:

$$L(M) = \{w \in \Sigma^* \mid sw \overset{*}{\vdash} w_1 p w_2 \text{ donde } p \in F \text{ y } w_1, w_2 \in \Gamma^*\}$$

2 Máquinas de Turing como aceptadoras de lenguajes

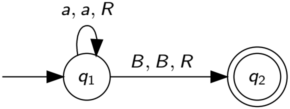
Una MT se puede comportar como un aceptador de lenguajes igual que los AF,s y AP,s. Colocaremos una cadena w en la cinta, la cabeza de lectura/escritura sobre el primer símbolo de la cadena, la MT en el estado inicial, y la pondremos en marcha. La cadena w es aceptada si después de una secuencia de movimientos la MT llega a un estado final (y por tanto se detiene), incluso aunque la cadena no haya sido procesada totalmente

Definición

El **lenguaje aceptado por una MT** M se define entonces como:

$$L(M) = \{w \in \Sigma^* \mid sw \overset{*}{\vdash} w_1 p w_2 \text{ donde } p \in F \text{ y } w_1, w_2 \in \Gamma^*\}$$

Por ejemplo, la siguiente MT acepta el lenguaje regular a^* sobre $\Sigma = \{a, b\}$. Esta MT para en q_2 sólo si se analiza una cadena de 0 o más $a$es y después encontramos un blanco. Si aparece alguna b, la MT no tendrá ningún movimiento definido y también se parará, pero en q_1, que no es un estado final.



Por tanto, para rechazar una cadena podemos:

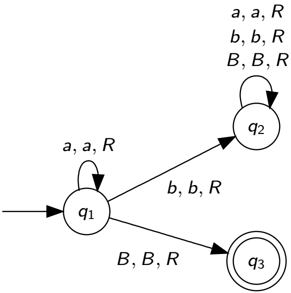
- Hacer que la MT se detenga en un estado no final.
- O bien, hacer que la MT entre en un bucle infinito que nunca alcanza un estado final (ya que seguimos suponiendo que no hay transiciones salientes definidas para los estados finales, y que cualquier MT que alcance un estado final automáticamente parará).

2 Máquinas de Turing como aceptadoras de lenguajes

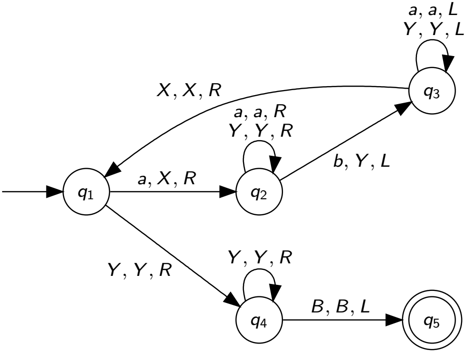
Por tanto, para rechazar una cadena podemos:

- Hacer que la MT se detenga en un estado no final.
- O bien, hacer que la MT entre en un bucle infinito que nunca alcanza un estado final (ya que seguimos suponiendo que no hay transiciones salientes definidas para los estados finales, y que cualquier MT que alcance un estado final automáticamente parará).

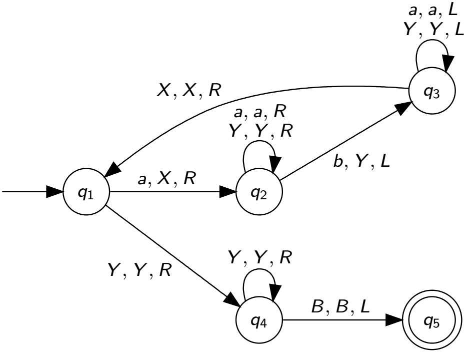
Por ejemplo, la siguiente figura muestra una MT que también acepta a^*, pero que rechaza cadenas no por parada en un estado no final, sino por entrada en un bucle infinito:



Otro ejemplo de MT aceptadora es la siguiente, que reconoce el LIC $\{a^n b^n \mid n \geq 1\}$:



Otro ejemplo de MT aceptadora es la siguiente, que reconoce el LIC $\{a^n b^n \mid n \geq 1\}$:

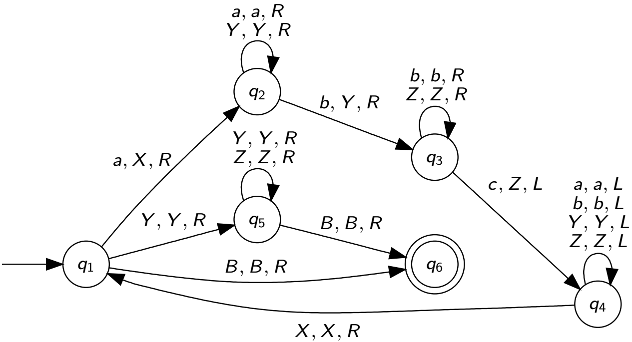


Para reconocer dicho lenguaje con una MT, haremos lo siguiente:

- Primero *marcaremos* la primera a con otro símbolo, por ejemplo, con una X.
- Entonces, nos desplazaremos hacia la derecha hasta encontrar la primera b, y la convertiremos, por ejemplo, en una Y.
- Seguidamente nos desplazaremos hasta encontrar la a más a la izquierda, y repetiremos el proceso hasta que no queden $a$es (en su lugar habrá símbolos X).
- Finalmente, comprobamos que todas las $b$es hayan sido convertidas en símbolos Y.

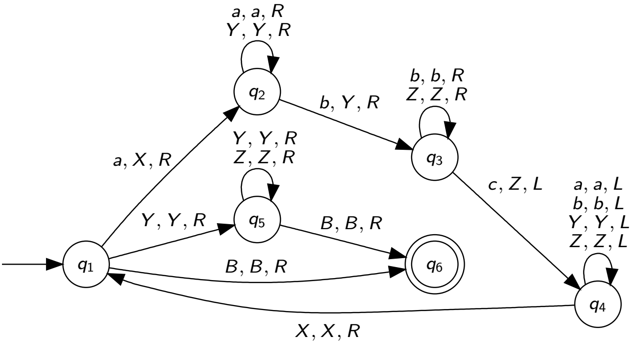
2 Máquinas de Turing como aceptadoras de lenguajes

Utilizando una técnica similar, vemos ahora una MT capaz de reconocer $\{a^n b^n c^n \mid n \geq 0\}$, que recordemos que no es un LIC:



2 Máquinas de Turing como aceptadoras de lenguajes

Utilizando una técnica similar, vemos ahora una MT capaz de reconocer $\{a^n b^n c^n \mid n \geq 0\}$, que recordemos que no es un LIC:



Esta MT trabaja del modo siguiente:

- Para cada a marcada con X, se marca una b con Y y una c con Z.
- Este proceso se repite hasta que todas las $a$es estén marcadas.
- Finalmente, se comprueba que todas las $b$es y $c$es hayan sido marcadas también.

2 Máquinas de Turing como aceptadoras de lenguajes

Definición

Un lenguaje aceptado por una MT se denomina **lenguaje recursivamente enumerable**, ya que todas sus cadenas pueden ser listadas o enumeradas por una MT. Este conjunto de lenguajes incluye de manera estricta a los LIC,s.

2 Máquinas de Turing como aceptadoras de lenguajes

Definición

Un lenguaje aceptado por una MT se denomina **lenguaje recursivamente enumerable**, ya que todas sus cadenas pueden ser listadas o enumeradas por una MT. Este conjunto de lenguajes incluye de manera estricta a los LIC,s.

Existen lenguajes recursivamente enumerables para los cuales ninguna MT que los acepta es capaz de parar con todas y cada una de las posibles cadenas de entrada. Obviamente, sí para con las del lenguaje, y además lo hace en un estado final. Pero con las cadenas fuera del lenguaje, podría parar en un estado no final, o podría no parar.

2 Máquinas de Turing como aceptadoras de lenguajes

Definición

Un lenguaje aceptado por una MT se denomina **lenguaje recursivamente enumerable**, ya que todas sus cadenas pueden ser listadas o enumeradas por una MT. Este conjunto de lenguajes incluye de manera estricta a los LIC,s.

Existen lenguajes recursivamente enumerables para los cuales ninguna MT que los acepta es capaz de parar con todas y cada una de las posibles cadenas de entrada. Obviamente, sí para con las del lenguaje, y además lo hace en un estado final. Pero con las cadenas fuera del lenguaje, podría parar en un estado no final, o podría no parar.

Definición

A la subclase de lenguajes recursivamente enumerables que son aceptados por al menos una MT que siempre para con cualquier cadena de entrada, independientemente de si se acepta o no, se le da el nombre de **lenguajes recursivos**.

Más adelante, retomaremos y ampliaremos todos estos conceptos.

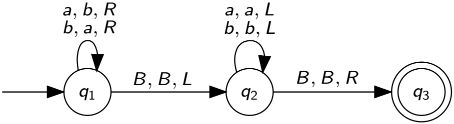
2 Máquinas de Turing como aceptadoras de lenguajes

Puesto que las MT,s pueden leer y escribir, podemos hablar entonces de la conversión de una **cadena de entrada** en una **cadena de salida**. Ambas están formadas por los símbolos de la cinta no blancos, una antes de la computación, y la otra después de la parada. Por tanto, la MT se considera como un **modelo abstracto de computación**.

2 Máquinas de Turing como aceptadoras de lenguajes

Puesto que las MT,s pueden leer y escribir, podemos hablar entonces de la conversión de una **cadena de entrada** en una **cadena de salida**. Ambas están formadas por los símbolos de la cinta no blancos, una antes de la computación, y la otra después de la parada. Por tanto, la MT se considera como un **modelo abstracto de computación**.

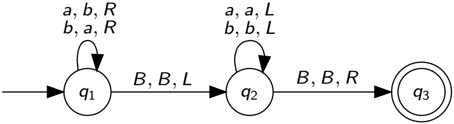
Por ejemplo, esta MT cambia *a*es por *b*es y viceversa, y se posiciona de nuevo al principio de la cadena:



2 Máquinas de Turing como aceptadoras de lenguajes

Puesto que las MT,s pueden leer y escribir, podemos hablar entonces de la conversión de una **cadena de entrada** en una **cadena de salida**. Ambas están formadas por los símbolos de la cinta no blancos, una antes de la computación, y la otra después de la parada. Por tanto, la MT se considera como un **modelo abstracto de computación**.

Por ejemplo, esta MT cambia *a*es por *b*es y viceversa, y se posiciona de nuevo al principio de la cadena:

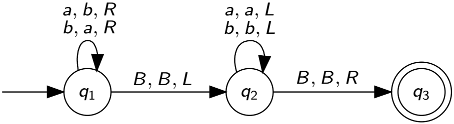


Así pues, cualquier MT se puede considerar también como una función sobre cadenas de la forma $f(w) = u$, cuando $sw \overset{*}{\vdash} pu$, donde s es el estado inicial, $p \in F$, $w \in \Sigma^*$ y $u \in \Gamma^*$.

2 Máquinas de Turing como aceptadoras de lenguajes

Puesto que las MT,s pueden leer y escribir, podemos hablar entonces de la conversión de una **cadena de entrada** en una **cadena de salida**. Ambas están formadas por los símbolos de la cinta no blancos, una antes de la computación, y la otra después de la parada. Por tanto, la MT se considera como un **modelo abstracto de computación**.

Por ejemplo, esta MT cambia *a*es por *b*es y viceversa, y se posiciona de nuevo al principio de la cadena:



Así pues, cualquier MT se puede considerar también como una función sobre cadenas de la forma $f(w) = u$, cuando $sw \overset{*}{\vdash} pu$, donde s es el estado inicial, $p \in F$, $w \in \Sigma^*$ y $u \in \Gamma^*$.

Definición

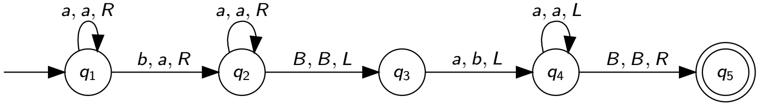
Y, en general, se dice que una función de cadena f es **Turing computable** si existe una MT para la cual $sw \overset{*}{\vdash} pu$, siempre que $f(w) = u$.

2 Máquinas de Turing como aceptadoras de lenguajes

Este concepto se puede extender fácilmente a las funciones que trabajan con números enteros. Por ejemplo, si sobre el alfabeto $\Sigma = \{a, b\}$ representamos el entero positivo n mediante a^n, la función de suma $f(n, m) = n + m$ podría implementarse mediante la transformación de $a^n b a^m$ en $a^{n+m} b$.

Este concepto se puede extender fácilmente a las funciones que trabajan con números enteros. Por ejemplo, si sobre el alfabeto $\Sigma = \{a, b\}$ representamos el entero positivo n mediante a^n, la función de suma $f(n, m) = n + m$ podría implementarse mediante la transformación de $a^n ba^m$ en $a^{n+m}b$.

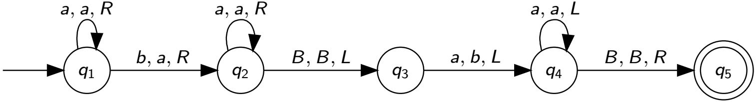
La MT que realiza esta tarea es la siguiente:



2 Máquinas de Turing como aceptadoras de lenguajes

Este concepto se puede extender fácilmente a las funciones que trabajan con números enteros. Por ejemplo, si sobre el alfabeto $\Sigma = \{a, b\}$ representamos el entero positivo n mediante a^n, la función de suma $f(n, m) = n + m$ podría implementarse mediante la transformación de $a^n b a^m$ en $a^{n+m} b$.

La MT que realiza esta tarea es la siguiente:



Ejercicio

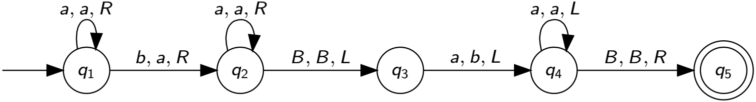
Construya una MT que calcule la función de paridad de los números naturales:

$$f(n) = \begin{cases} 0, \text{ si } n \text{ es par} \\ 1, \text{ si } n \text{ es impar} \end{cases}$$

2 Máquinas de Turing como aceptadoras de lenguajes

Este concepto se puede extender fácilmente a las funciones que trabajan con números enteros. Por ejemplo, si sobre el alfabeto $\Sigma = \{a, b\}$ representamos el entero positivo n mediante a^n, la función de suma $f(n, m) = n + m$ podría implementarse mediante la transformación de $a^n b a^m$ en $a^{n+m} b$.

La MT que realiza esta tarea es la siguiente:

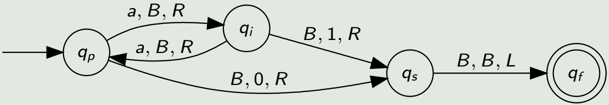


Ejercicio

Construya una MT que calcule la función de paridad de los números naturales:

$$f(n) = \begin{cases} 0, \text{ si } n \text{ es par} \\ 1, \text{ si } n \text{ es impar} \end{cases}$$

<u>Solución</u>:



Contenidos

3 Construcción de máquinas de Turing

Hemos visto que las MT,s pueden realizar muchas tareas, además del reconocimiento de lenguajes. Nos centraremos ahora en la forma de simplificar su construcción. La idea es construir una colección de MT,s sencillas y combinarlas para crear otras más complejas.

3 Construcción de máquinas de Turing

Hemos visto que las MT,s pueden realizar muchas tareas, además del reconocimiento de lenguajes. Nos centraremos ahora en la forma de simplificar su construcción. La idea es construir una colección de MT,s sencillas y combinarlas para crear otras más complejas.

Para ello permitiremos que dos o más MT,s compartan la misma cinta. Cuando una termina su ejecución, empieza la siguiente tomando como contenido de la cinta lo que dejó la otra y, como posición de inicio de la cabeza de L/E, la celda sobre la que terminó su computación dicha MT.

3 Construcción de máquinas de Turing

Hemos visto que las MT,s pueden realizar muchas tareas, además del reconocimiento de lenguajes. Nos centraremos ahora en la forma de simplificar su construcción. La idea es construir una colección de MT,s sencillas y combinarlas para crear otras más complejas.

Para ello permitiremos que dos o más MT,s compartan la misma cinta. Cuando una termina su ejecución, empieza la siguiente tomando como contenido de la cinta lo que dejó la otra y, como posición de inicio de la cabeza de L/E, la celda sobre la que terminó su computación dicha MT.

Definición

Formalmente, la **composición de MT,s** se define como sigue. Consideremos dos máquinas de Turing, $M_1 = (Q_1, \Sigma, \Gamma, s_1, B, F_1, \delta_1)$ y $M_2 = (Q_2, \Sigma, \Gamma, s_2, B, F_2, \delta_2)$, sobre el mismo alfabeto de entrada Σ y el mismo alfabeto de la cinta Γ. Suponemos que $Q_1 \cap Q_2 = \emptyset$. La composición de M_1 y M_2, que denotamos como $M_1 M_2$, es la máquina $M = (Q, \Sigma, \Gamma, s, B, F, \delta)$, donde:

$$Q = Q_1 \cup Q_2 \qquad s = s_1 \qquad F = F_2$$

$$\delta(q, \sigma) = \begin{cases} \delta_1(q, \sigma), & \text{si } q \in Q_1 \text{ y } \delta_1(q, \sigma) \neq (p, \tau, X), \ \forall p \in F_1 \\ \delta_2(q, \sigma), & \text{si } q \in Q_2 \\ (s_2, \tau, X), & \text{si } q \in Q_1 \text{ y } \delta_1(q, \sigma) = (p, \tau, X), \text{ para algún } p \in F_1 \end{cases}$$

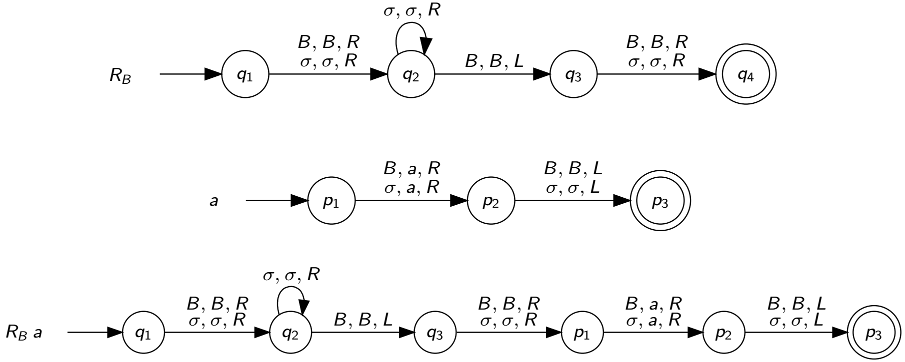
Es decir, la composición $M_1 M_2$ se comporta como M_1 hasta que M_1 llega a un estado final, en ese momento cambia al estado inicial de M_2, y se comporta como M_2 hasta que termina.

3 Construcción de máquinas de Turing

Por ejemplo, la primera de estas MT,s busca el primer blanco que haya a la derecha de su posición de comienzo. Nos referiremos a esta máquina como R_B.

La segunda MT escribe una a y para, sea cual sea el contenido de la celda actual, y permanece sobre ella. Nos referiremos a esta máquina simplemente como a.

Por último, al combinar estas dos máquinas, obtenemos el dispositivo $R_B\ a$ que primero busca el primer blanco hacia la derecha, después escribe una a, y queda posicionado en dicha celda.



$\forall \sigma \in \Gamma - \{B\}$

3 Construcción de máquinas de Turing

Por supuesto, podemos hacer composiciones sucesivas. Por ejemplo, $R_B R_B R_B\, a$ aplicada sobre:

$\cdots$	a	a	a	B	a	a	B	B	B	$\cdots$
	$\uparrow$									

transformaría esta cinta en:

$\cdots$	a	a	a	B	a	a	B	a	B	$\cdots$
								$\uparrow$		

3 Construcción de máquinas de Turing

Por supuesto, podemos hacer composiciones sucesivas. Por ejemplo, $R_B R_B R_B\, a$ aplicada sobre:

$\cdots$	a	a	a	B	a	a	B	B	B	$\cdots$
	$\uparrow$									

transformaría esta cinta en:

$\cdots$	a	a	a	B	a	a	B	a	B	$\cdots$
								$\uparrow$		

Otras máquinas básicas que consideraremos (y cuya construcción se plantea como ejercicio propuesto) son las siguientes:

- L_B: hace lo mismo que R_B pero hacia la izquierda, es decir, busca el primer blanco que haya a la izquierda de su posición de comienzo.
- $R_{\overline{B}}$: busca el primer símbolo no blanco que haya a la derecha de su posición de comienzo.
- $L_{\overline{B}}$: hace lo mismo que $R_{\overline{B}}$ pero hacia la izquierda.
- R: se mueve una posición a la derecha.
- L: se mueve una posición a la izquierda.

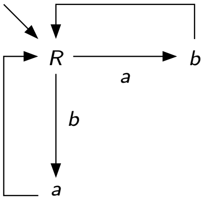
3 Construcción de máquinas de Turing

A veces, querremos hacer composiciones de tal manera que la máquina resultante haga una cosa u otra, en función del contenido de una celda.

Podemos expresar estas condiciones mediante diagramas de bifurcación, similares a los que utilizábamos para construir los autómatas finitos.

Por ejemplo, la siguiente MT cambia *a*es por *b*es y *b*es por *a*es.

Esta máquina se detiene cuando alcanza un símbolo blanco.



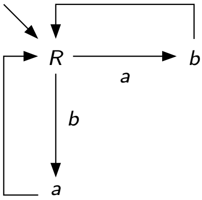
3 Construcción de máquinas de Turing

A veces, querremos hacer composiciones de tal manera que la máquina resultante haga una cosa u otra, en función del contenido de una celda.

Podemos expresar estas condiciones mediante diagramas de bifurcación, similares a los que utilizábamos para construir los autómatas finitos.

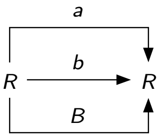
Por ejemplo, la siguiente MT cambia *a*es por *b*es y *b*es por *a*es.

Esta máquina se detiene cuando alcanza un símbolo blanco.



Otras cuestiones de notación relacionadas con este tipo de diagramas son estas.

Si $\Gamma = \{a, b, B\}$ y si tenemos, por ejemplo:



son equivalentes estas representaciones:

$$R \xrightarrow{a, b, B} R \qquad\qquad R \longrightarrow R$$

$$RR \qquad\qquad R^2$$

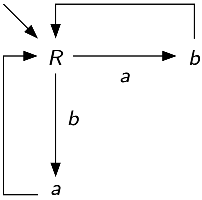
3 Construcción de máquinas de Turing

A veces, querremos hacer composiciones de tal manera que la máquina resultante haga una cosa u otra, en función del contenido de una celda.

Podemos expresar estas condiciones mediante diagramas de bifurcación, similares a los que utilizábamos para construir los autómatas finitos.

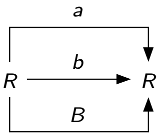
Por ejemplo, la siguiente MT cambia *a*es por *b*es y *b*es por *a*es.

Esta máquina se detiene cuando alcanza un símbolo blanco.



Otras cuestiones de notación relacionadas con este tipo de diagramas son estas.

Si $\Gamma = \{a, b, B\}$ y si tenemos, por ejemplo:

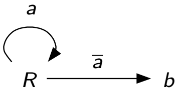


son equivalentes estas representaciones:

$R \xrightarrow{a,\, b,\, B} R \qquad R \longrightarrow R$

$RR \qquad R^2$

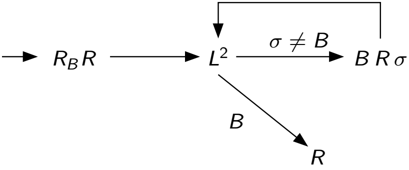
Y otro caso frecuente es el de una rama para un símbolo y otra para los demás:



3 Construcción de máquinas de Turing

A continuación vemos una MT que desplaza una cadena sobre la cinta una posición a la derecha, suponiendo la cadena a desplazar precedida y seguida por blancos. Es decir, esta máquina transforma $\underline{B}wB$ en $B\underline{B}w$.

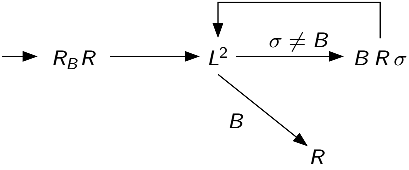
En este ejemplo, es interesante destacar que podemos indicar que la MT *recuerda* ciertos símbolos mediante la utilización de variables. En este caso, es suficiente con una, que hemos llamado σ. De esta manera, evitamos construir un determinado camino de procesamiento para cada uno de los símbolos del alfabeto de la cinta.



3 Construcción de máquinas de Turing

A continuación vemos una MT que desplaza una cadena sobre la cinta una posición a la derecha, suponiendo la cadena a desplazar precedida y seguida por blancos. Es decir, esta máquina transforma $\underline{B}wB$ en $B\underline{B}w$.

En este ejemplo, es interesante destacar que podemos indicar que la MT *recuerda* ciertos símbolos mediante la utilización de variables. En este caso, es suficiente con una, que hemos llamado σ. De esta manera, evitamos construir un determinado camino de procesamiento para cada uno de los símbolos del alfabeto de la cinta.



Esta MT pasa a formar parte de nuestra colección de máquinas básicas con el nombre de S_R.

Y tiene su máquina básica simétrica S_L (que transforma $\underline{\sigma_1}\sigma_2 wB$ en $\underline{\sigma_2}wBB$ y cuya construcción se plantea como ejercicio propuesto).

3 Construcción de máquinas de Turing

Estas máquinas abstractas que estamos construyendo pueden también hacer referencia no sólo a las situaciones de aceptación, sino también a las de rechazo.

Por ejemplo, la siguiente MT reconoce $\{w\, w^I \mid w \in \Sigma^*\}$.

Comienza a trabajar con $\underline{B}uB$ en la cinta, e intenta descubrir si $u = w\, w^I$, para lo cual utiliza dos variables, σ y τ.

Y para realizar este reconocimiento, compara los símbolos de los extremos derecho e izquierdo:

- Si coinciden, los cambia por blancos y repite el proceso, de forma que la cadena será aceptada cuando todos los símbolos hayan sido reemplazados por blancos.
- Si no coinciden, la cadena se rechaza. Y recordemos que la situación de rechazo podría venir dada por la entrada en un estado no final o en un bucle que, por ejemplo, escriba el símbolo B indefinidamente.

$$\begin{tikzcd}
{} \arrow[r] & R \arrow[r, "\sigma \neq B"] \arrow[d, "B"] & BR_B L \arrow[r, "\tau \neq B"] \arrow[dr, "B"'] & B \arrow[r, "\sigma = \tau"] \arrow[d, "\sigma \neq \tau"] & L_B \arrow[lll, bend right] \\
 & \begin{array}{c} B \\ \text{Aceptar} \end{array} & & \begin{array}{c} B \\ \text{Rechazar} \end{array} &
\end{tikzcd}$$

Ejercicio

¿Qué efecto tiene S_R sobre $\underline{B}aaBbbB$? ¿Qué efecto tiene S_R^2 sobre $\underline{B}aaBbbBccB$?

Ejercicio

¿Qué efecto tiene S_R sobre $\underline{B}aaBbbB$? ¿Qué efecto tiene S_R^2 sobre $\underline{B}aaBbbBccB$?

Solución:

En el primer caso, $B\underline{B}aabbB$.

En el segundo caso, primero $B\underline{B}aabbBccB$ y después $BB\underline{B}aabbccB$.

Ejercicio

¿Qué efecto tiene S_R sobre $\underline{B}aaBbbB$? ¿Qué efecto tiene S_R^2 sobre $\underline{B}aaBbbBccB$?

<u>Solución</u>:

En el primer caso, $B\underline{B}aabbB$.

En el segundo caso, primero $B\underline{B}aabbBccB$ y después $BB\underline{B}aabbccB$.

Ejercicio

Mediante composición de máquinas básicas, construya C (la MT de copia) que transforma $\underline{B}wB$ en $\underline{B}wBwB$.

Ejercicio

¿Qué efecto tiene S_R sobre $\underline{B}aaBbbB$? ¿Qué efecto tiene S_R^2 sobre $\underline{B}aaBbbBccB$?

Solución:

En el primer caso, $B\underline{B}aabbB$.

En el segundo caso, primero $B\underline{B}aabbBccB$ y después $BB\underline{B}aabbccB$.

Ejercicio

Mediante composición de máquinas básicas, construya C (la MT de copia) que transforma $\underline{B}wB$ en $\underline{B}wBwB$.

Solución:

$$\begin{tikzcd}
{} \arrow[r] & R \arrow[rr, "\sigma \neq B"] \arrow[d, "B"] & & B\, R_B^2\, \sigma\, L_B^2\, \sigma \arrow[ll, bend right] \\
& L_B & &
\end{tikzcd}$$

Ejercicio

Mediante composición de máquinas básicas, construya una MT que acepte $\{w \mid w = w^I\}$, suponiendo que la situación inicial es $\underline{B}wB$.

Ejercicio

Mediante composición de máquinas básicas, construya una MT que acepte $\{w \mid w = w^I\}$, suponiendo que la situación inicial es $\underline{B}wB$.

Solución:

$$
\begin{tikzcd}
{} \arrow[r] & R \arrow[r, "\sigma \neq B"] \arrow[d, "B"] & R_B\, L \arrow[r, "\sigma"] \arrow[d, "\overline{\sigma}"] & B\, L_B\, R\, B \arrow[ll, bend right] \\
& \begin{array}{c} B \\ \text{Aceptar} \end{array} & \begin{array}{c} B \\ \text{Rechazar} \end{array} &
\end{tikzcd}
$$

3 Construcción de máquinas de Turing

Ejercicio

Mediante composición de máquinas básicas, construya una MT que acepte $\{w \mid w = w^I\}$, suponiendo que la situación inicial es $\underline{B}wB$.

<u>Solución</u>:

$$\begin{tikzcd}
{} \arrow[r] & R \arrow[r, "\sigma \neq B"] \arrow[d, "B"] & R_B\, L \arrow[r, "\sigma"] \arrow[d, "\overline{\sigma}"] & B\, L_B\, R\, B \arrow[ll, bend right] \\
& B & B & \\
& \text{Aceptar} & \text{Rechazar} &
\end{tikzcd}$$

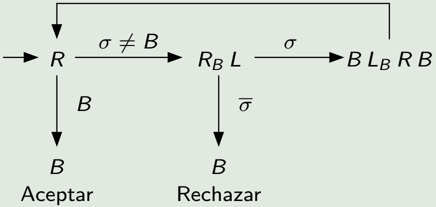
Ejercicio

Mediante composición de máquinas básicas, construya una MT que transforme $\underline{B}wB$ en $\underline{B}w^IB$.

Ejercicio

Mediante composición de máquinas básicas, construya una MT que acepte $\{w \mid w = w^I\}$, suponiendo que la situación inicial es $\underline{B}wB$.

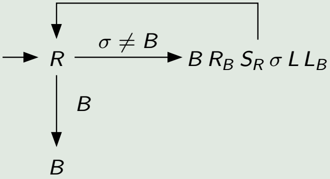
Solución:



Ejercicio

Mediante composición de máquinas básicas, construya una MT que transforme $\underline{B}wB$ en $\underline{B}w^IB$.

Solución:



Ejercicio

La resta de números naturales no está definida, ya que la diferencia puede ser negativa. Por tanto, definimos la resta modificada como:

$$n \dot{-} m = \begin{cases} n - m, & \text{si } n \geq m \\ 0, & \text{si } n < m \end{cases}$$

Mediante composición de máquinas básicas, construya entonces una MT que transforme $\underline{B}a^nBa^mB$ en $\underline{B}a^{n \dot{-} m}B$ (es decir, en $\underline{B}a^{n-m}B$ si $n \geq m$, o en $\underline{B}B$ si $n < m$).

Ejercicio

La resta de números naturales no está definida, ya que la diferencia puede ser negativa. Por tanto, definimos la resta modificada como:

$$n \mathbin{\dot{-}} m = \begin{cases} n - m, & \text{si } n \geq m \\ 0, & \text{si } n < m \end{cases}$$

Mediante composición de máquinas básicas, construya entonces una MT que transforme $\underline{B}a^nBa^mB$ en $\underline{B}a^{n \dot{-} m}B$ (es decir, en $\underline{B}a^{n-m}B$ si $n \geq m$, o en $\underline{B}B$ si $n < m$).

<u>Solución</u>:

$$\begin{tikzcd}
{} \arrow[r] & R \arrow[r, "a"] \arrow[d, "B"'] & B\,R_B^2\,L \arrow[r, "a"] \arrow[dr, "B"'] & B\,L_B^2 \arrow[ll, bend right] \\
& R \arrow[r, "a"] \arrow[d, "B"'] & B \arrow[l, bend left] & L_B\,a\,L \\
& L & &
\end{tikzcd}$$

Contenidos

4 Modificaciones de las máquinas de Turing

Existen otras definiciones alternativas de la MT. Todas ellas son equivalentes a la definición que hemos visto. Es decir, aunque alguno de estos nuevos modelos es mucho más complejo, todos ellos tienen la misma potencia computacional. Por tanto, lo que aportan es una mayor flexibilidad a la hora de diseñar MT,s que resuelvan ciertos problemas particulares.

4 Modificaciones de las máquinas de Turing

Existen otras definiciones alternativas de la MT. Todas ellas son equivalentes a la definición que hemos visto. Es decir, aunque alguno de estos nuevos modelos es mucho más complejo, todos ellos tienen la misma potencia computacional. Por tanto, lo que aportan es una mayor flexibilidad a la hora de diseñar MT,s que resuelvan ciertos problemas particulares.

Algunos ejemplos de estos modelos alternativos son los siguientes.

MT que escribe en la cinta y permanece en la misma celda

Si ampliamos la función de transición a:

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

donde S (*stay*) significa permanecer en la misma celda, es decir, no mover la cabeza de lectura/escritura, entonces podemos definir más cómodamente algunas operaciones de búsqueda, tales como R_B o L_B, y evitar así algunos movimientos.

4 Modificaciones de las máquinas de Turing

Existen otras definiciones alternativas de la MT. Todas ellas son equivalentes a la definición que hemos visto. Es decir, aunque alguno de estos nuevos modelos es mucho más complejo, todos ellos tienen la misma potencia computacional. Por tanto, lo que aportan es una mayor flexibilidad a la hora de diseñar MT,s que resuelvan ciertos problemas particulares.

Algunos ejemplos de estos modelos alternativos son los siguientes.

MT que escribe en la cinta y permanece en la misma celda

Si ampliamos la función de transición a:

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

donde S (*stay*) significa permanecer en la misma celda, es decir, no mover la cabeza de lectura/escritura, entonces podemos definir más cómodamente algunas operaciones de búsqueda, tales como R_B o L_B, y evitar así algunos movimientos.

Como hemos dicho, esto no amplía la potencia de cálculo. Esta MT se puede simular mediante una MT original con algunos estados y movimientos extra.

Cada vez que $\delta(q, \sigma) = (p, \sigma', S)$:

- Hacemos $\delta(q, \sigma) = (p', \sigma', R)$ y $\delta(p', \tau) = (p, \tau, L)$, $\forall \tau \in \Gamma$.
- O bien $\delta(q, \sigma) = (p', \sigma', L)$ y $\delta(p', \tau) = (p, \tau, R)$, $\forall \tau \in \Gamma$.

MT multipista

En esta definición de MT, cada celda de la cinta se divide en subceldas, de foma que cada subcelda puede contener un símbolo.

Por ejemplo, la cinta:

$\cdots$	B	b	b	$\cdots$
$\cdots$	a	a	b	$\cdots$
$\cdots$	a	B	B	$\cdots$
		$\uparrow$		

tiene cada celda dividida en tres subceldas. Por tanto, el contenido de cada celda se podría representar con n-tuplas. En este caso, $n = 3$ y las tuplas serían $(B, a, a)\underline{(b, a, B)}(b, b, B)$.

4 Modificaciones de las máquinas de Turing

MT multipista

En esta definición de MT, cada celda de la cinta se divide en subceldas, de foma que cada subcelda puede contener un símbolo.

Por ejemplo, la cinta:

$\cdots$	B	b	b	$\cdots$
$\cdots$	a	a	b	$\cdots$
$\cdots$	a	B	B	$\cdots$
		$\uparrow$		

tiene cada celda dividida en tres subceldas. Por tanto, el contenido de cada celda se podría representar con n-tuplas. En este caso, $n = 3$ y las tuplas serían $(B, a, a)\underline{(b, a, B)}(b, b, B)$.

Los movimientos de la MT dependerán de su estado actual y de la n-tupla que represente el contenido de la celda actual. Por tanto, si en esta MT tenemos que $\Gamma = \{a, b, B\}$, la podríamos simular con una MT original cuyo alfabeto de la cinta sea $\Gamma \times \Gamma \times \Gamma$.

4 Modificaciones de las máquinas de Turing

MT multipista

En esta definición de MT, cada celda de la cinta se divide en subceldas, de foma que cada subcelda puede contener un símbolo.

Por ejemplo, la cinta:

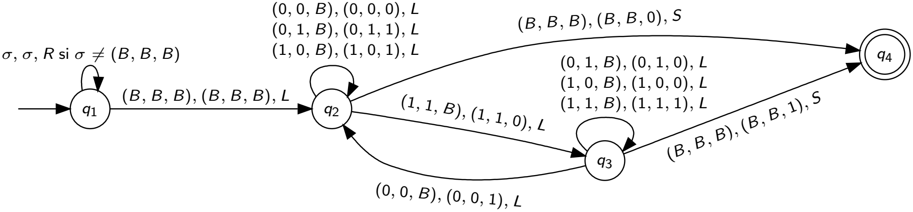
$\cdots$	B	b	b	$\cdots$
$\cdots$	a	a	b	$\cdots$
$\cdots$	a	B	B	$\cdots$
		$\uparrow$		

tiene cada celda dividida en tres subceldas. Por tanto, el contenido de cada celda se podría representar con n-tuplas. En este caso, $n = 3$ y las tuplas serían $(B, a, a)\underline{(b, a, B)}(b, b, B)$.

Los movimientos de la MT dependerán de su estado actual y de la n-tupla que represente el contenido de la celda actual. Por tanto, si en esta MT tenemos que $\Gamma = \{a, b, B\}$, la podríamos simular con una MT original cuyo alfabeto de la cinta sea $\Gamma \times \Gamma \times \Gamma$.

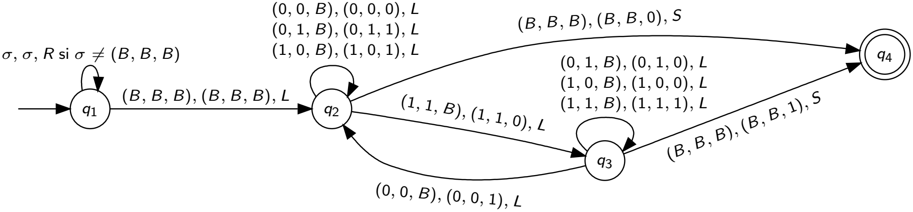
¿Qué tipo de operaciones facilita una MT multipista? Supongamos que queremos sumar dos números binarios. Podemos construir una MT de 3 pistas. Los dos sumandos ocuparían las pistas 1ª y 2ª, con sus dígitos alineados por la derecha. Si son de distinta longitud, rellenamos con ceros las subceldas que sean necesarias. La 3ª pista albergaría el resultado.

La MT que realiza esta tarea podría ser la siguiente:



4 Modificaciones de las máquinas de Turing

La MT que realiza esta tarea podría ser la siguiente:



Por ejemplo, al sumar 111 y 10, las configuraciones inicial y final de la cinta serían estas:

$\cdots$	B	1	1	1	B	$\cdots$
$\cdots$	B	0	1	0	B	$\cdots$
$\cdots$	B	B	B	B	B	$\cdots$
		$\uparrow$				

$\cdots$	B	1	1	1	B	$\cdots$
$\cdots$	B	0	1	0	B	$\cdots$
$\cdots$	1	0	0	1	B	$\cdots$
	$\uparrow$					

4 Modificaciones de las máquinas de Turing

MT con la cinta infinita en un solo sentido

En esta variante, la cinta se extiende de manera infinita, generalmente, sólo hacia la derecha. Por tanto, no se permiten movimientos hacia la izquierda desde la celda del extremo izquierdo.

MT con la cinta infinita en un solo sentido

En esta variante, la cinta se extiende de manera infinita, generalmente, sólo hacia la derecha. Por tanto, no se permiten movimientos hacia la izquierda desde la celda del extremo izquierdo.

Por supuesto, cualquier MT así puede ser simulada por una MT original, con sólo marcar una celda cualquiera y no permitir moverse a la izquierda de ella. Y viceversa, cualquier MT con la cinta infinita en un solo sentido puede simular a una infinita en ambos, con sólo permitir que haya dos pistas en ese sentido: la superior contiene la información de la parte derecha, y la inferior contiene (en orden inverso) la información de la parte izquierda. Por ejemplo, si la cinta original contenía:

$\cdots$	B	a	b	b	b	a	b	$\cdots$

donde el punto de referencia para "doblar" la cinta se ha elegido después del primer símbolo b, la cinta infinita en un solo sentido, es decir, la cinta "doblada" (y, por tanto, de dos pistas) contendrá:

$*$	b	b	a	b	$\cdots$
$*$	b	a	B	B	$\cdots$

MT con la cinta infinita en un solo sentido

En esta variante, la cinta se extiende de manera infinita, generalmente, sólo hacia la derecha. Por tanto, no se permiten movimientos hacia la izquierda desde la celda del extremo izquierdo.

Por supuesto, cualquier MT así puede ser simulada por una MT original, con sólo marcar una celda cualquiera y no permitir moverse a la izquierda de ella. Y viceversa, cualquier MT con la cinta infinita en un solo sentido puede simular a una infinita en ambos, con sólo permitir que haya dos pistas en ese sentido: la superior contiene la información de la parte derecha, y la inferior contiene (en orden inverso) la información de la parte izquierda. Por ejemplo, si la cinta original contenía:

$\cdots$	B	a	b	b	b	a	b	$\cdots$

donde el punto de referencia para "doblar" la cinta se ha elegido después del primer símbolo b, la cinta infinita en un solo sentido, es decir, la cinta "doblada" (y, por tanto, de dos pistas) contendrá:

$*$	b	b	a	b	$\cdots$
$*$	b	a	B	B	$\cdots$

Cuando la MT original trabaja sobre la parte derecha del punto de referencia, ésta trabajará sobre la pista de arriba. Y cuando lo haga sobre la parte izquierda, ésta trabajará sobre la pista de abajo. Así pues, lo único que faltaría sería introducir los estados y movimientos necesarios para que, cuando se alcancen los dos símbolos $*$, esta MT cambie de pista y de dirección. Pero, en resumen, ambas definiciones son también equivalentes e igualmente potentes.

4 Modificaciones de las máquinas de Turing

MT multicinta

En esta definición, la MT tiene varias cintas, cada una con su propia cabeza de L/E.

4 Modificaciones de las máquinas de Turing

MT multicinta

En esta definición, la MT tiene varias cintas, cada una con su propia cabeza de L/E.

En un solo movimiento, esta MT:

1. Cambia de estado dependiendo del estado actual y del contenido de las celdas actuales de todas las cintas.
2. Escribe un nuevo símbolo en cada una de esas celdas, que no tiene por qué ser el mismo para todas ellas.
3. Mueve cada cabeza de L/E a la derecha, a la izquierda, o la hace permanecer en el mismo sitio. Una vez más, cada cabeza puede hacer una cosa diferente al resto de las cabezas.

Por tanto, siendo n el número de cintas, la función de transición de esta MT es:

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

MT multicinta

En esta definición, la MT tiene varias cintas, cada una con su propia cabeza de L/E.

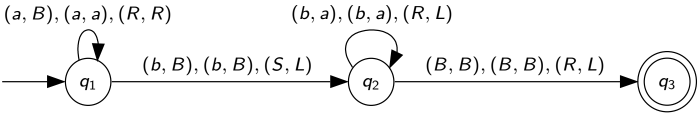
En un solo movimiento, esta MT:

1. Cambia de estado dependiendo del estado actual y del contenido de las celdas actuales de todas las cintas.
2. Escribe un nuevo símbolo en cada una de esas celdas, que no tiene por qué ser el mismo para todas ellas.
3. Mueve cada cabeza de L/E a la derecha, a la izquierda, o la hace permanecer en el mismo sitio. Una vez más, cada cabeza puede hacer una cosa diferente al resto de las cabezas.

Por tanto, siendo n el número de cintas, la función de transición de esta MT es:

$$\delta : Q \times \Gamma^n \to Q \times \Gamma^n \times \{L, R, S\}^n$$

Con esta nueva definición, sería particularmente sencillo construir una MT para reconocer el lenguaje $\{a^n b^n \mid n \geq 1\}$. Dicho lenguaje es un LIC y por tanto es aceptado también por un AP. Así pues, para aceptar L sería suficiente con una MT multicinta de dos cintas: la primera almacenaría la cadena a procesar, y la segunda simularía el comportamiento de la pila.



Y una MT multicinta de 3 cintas que reconozca $\{a^{n^2} \mid n \geq 1\}$ podría construirse como sigue:

- La 1ª cinta almacenará el número n mediante una secuencia de n símbolos X. Al principio habrá sólo una X y, si fuera necesario, se añadirá una nueva X en cada paso adicional.

X	X	X	B	$\cdots$

- La 2ª cinta almacenará en todo momento, también mediante secuencias de símbolos X, el cuadrado del número almacenado en la 1ª cinta, es decir, n^2.

X	X	X	X	X	X	X	X	X	B	$\cdots$

- La 3ª cinta almacenará la secuencia de $a$es que constituye la cadena de entrada.

a	a	a	a	a	a	B	$\cdots$

4 Modificaciones de las máquinas de Turing

Y una MT multicinta de 3 cintas que reconozca $\{a^{n^2} \mid n \geq 1\}$ podría construirse como sigue:

- La 1ª cinta almacenará el número n mediante una secuencia de n símbolos X. Al principio habrá sólo una X y, si fuera necesario, se añadirá una nueva X en cada paso adicional.

X	X	X	B	$\cdots$

- La 2ª cinta almacenará en todo momento, también mediante secuencias de símbolos X, el cuadrado del número almacenado en la 1ª cinta, es decir, n^2.

X	X	X	X	X	X	X	X	X	B	$\cdots$

- La 3ª cinta almacenará la secuencia de $a$es que constituye la cadena de entrada.

a	a	a	a	a	a	B	$\cdots$

Esta MT comprueba continuamente los contenidos de las cintas 2ª y 3ª, para ver si el número de Xs es igual al número de $a$es, pudiendo ocurrir lo siguiente:

- Si llega a ver dos símbolos B, es que ambos números son iguales y la cadena se acepta.
- Si llega a ver una X y un símbolo B, es que hay más Xs que $a$es y la cadena se rechaza.
- Si llega a ver un símbolo B y una a, calcula el siguiente cuadrado perfecto en la 2ª cinta, añade una nueva X en la 1ª cinta, y repite el proceso.
  Dado que $(n+1)^2 = n^2 + 2n + 1$, el siguiente cuadrado perfecto puede generarse añadiendo en la 2ª cinta dos veces el contenido de la 1ª más una nueva X.

4 Modificaciones de las máquinas de Turing

Una MT M_1 con k cintas puede ser simulada por una MT M_2 con una única cinta dividida en $2k+1$ pistas.

Cada una de las cintas de M_1 corresponde a dos pistas de M_2: una pista almacena el contenido de la cinta de M_1 correspondiente, mientras que la otra se utiliza para guardar un marcador que indica la posición de la cabeza de lectura/escritura de dicha cinta.

La última pista de M_2 se utiliza para almacenar las posiciones de dos de las cabezas (o marcadores): la que está más a la izquierda y la que está más a la derecha.

4 Modificaciones de las máquinas de Turing

Una MT M_1 con k cintas puede ser simulada por una MT M_2 con una única cinta dividida en $2k + 1$ pistas.

Cada una de las cintas de M_1 corresponde a dos pistas de M_2: una pista almacena el contenido de la cinta de M_1 correspondiente, mientras que la otra se utiliza para guardar un marcador que indica la posición de la cabeza de lectura/escritura de dicha cinta.

La última pista de M_2 se utiliza para almacenar las posiciones de dos de las cabezas (o marcadores): la que está más a la izquierda y la que está más a la derecha.

Cada movimiento de M_1 es simulado por M_2 mediante un barrido de izquierda a derecha y otro de derecha a izquierda:

- El primer barrido comienza en el marcador más a la izquierda y termina en el marcador más a la derecha. De esta manera, M_2 adquiere toda la información relativa a los símbolos de las cintas y a las posiciones de las cabezas de M_1, y puede entonces determinar un movimiento.
- El segundo barrido va hacia la izquierda actualizando el contenido de la cinta y ajustando la posición de todos los marcadores, incluidos los marcadores especiales de los extremos.

Por último, M_2 pasa al estado correspondiente al estado siguiente de M_1.

MT no determinista

La última definición alternativa que consideraremos será aquélla en la cual la función de transición es de la forma:

$$\Delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}}$$

MT no determinista

La última definición alternativa que consideraremos será aquélla en la cual la función de transición es de la forma:

$$\Delta : Q \times \Gamma \to 2^{Q \times \Gamma \times \{L,R\}}$$

Las MT,s no deterministas incluyen ya a las deterministas. Para la simulación inversa, consideraremos M_1 una MT no determinista que acepta algún lenguaje, y construiremos una MT determinista M_2 de tres cintas que simule a M_1.

MT no determinista

La última definición alternativa que consideraremos será aquélla en la cual la función de transición es de la forma:

$$\Delta : Q \times \Gamma \to 2^{Q \times \Gamma \times \{L,R\}}$$

Las MT,s no deterministas incluyen ya a las deterministas. Para la simulación inversa, consideraremos M_1 una MT no determinista que acepta algún lenguaje, y construiremos una MT determinista M_2 de tres cintas que simule a M_1.

En M_1, para cualquier estado y símbolo actuales, hay un número finito de posibles movimientos. Numeramos dichos movimientos con $1, 2, \ldots, k$. Y además, como Q_1 y Γ_1 son finitos, podemos obtener el par (q, σ) para el cual $\Delta(q, \sigma)$ tiene el mayor número de movimientos posibles a realizar a continuación. Llamemos n a este número. Entonces, cualquier computación de M_1 se puede representar mediante secuencias finitas de números entre 1 y n. Por supuesto, no toda secuencia así será válida, porque habrá situaciones con menos de n movimientos posibles.

MT no determinista

La última definición alternativa que consideraremos será aquélla en la cual la función de transición es de la forma:

$$\Delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}}$$

Las MT,s no deterministas incluyen ya a las deterministas. Para la simulación inversa, consideraremos M_1 una MT no determinista que acepta algún lenguaje, y construiremos una MT determinista M_2 de tres cintas que simule a M_1.

En M_1, para cualquier estado y símbolo actuales, hay un número finito de posibles movimientos. Numeramos dichos movimientos con $1, 2, \ldots, k$. Y además, como Q_1 y Γ_1 son finitos, podemos obtener el par (q, σ) para el cual $\Delta(q, \sigma)$ tiene el mayor número de movimientos posibles a realizar a continuación. Llamemos n a este número. Entonces, cualquier computación de M_1 se puede representar mediante secuencias finitas de números entre 1 y n. Por supuesto, no toda secuencia así será válida, porque habrá situaciones con menos de n movimientos posibles.

En M_2, la primera cinta almacena la cadena de entrada. En la segunda cinta se generan sistemáticamente secuencias de números entre 1 y n. Para cada secuencia, M_2 copia la cadena de entrada en la tercera cinta y simula M_1 sobre esta cinta, usando la secuencia de operaciones de la segunda cinta como guía. Si una de las secuencias hace que M_1 acepte la cadena, entonces M_2 también aceptará. Si ninguna de las secuencias provoca la aceptación por parte de M_1, tampoco M_2 aceptará.

4 Modificaciones de las máquinas de Turing

MT no determinista

La última definición alternativa que consideraremos será aquélla en la cual la función de transición es de la forma:

$$\Delta : Q \times \Gamma \to 2^{Q \times \Gamma \times \{L,R\}}$$

Las MT,s no deterministas incluyen ya a las deterministas. Para la simulación inversa, consideraremos M_1 una MT no determinista que acepta algún lenguaje, y construiremos una MT determinista M_2 de tres cintas que simule a M_1.

En M_1, para cualquier estado y símbolo actuales, hay un número finito de posibles movimientos. Numeramos dichos movimientos con $1, 2, \ldots, k$. Y además, como Q_1 y Γ_1 son finitos, podemos obtener el par (q, σ) para el cual $\Delta(q, \sigma)$ tiene el mayor número de movimientos posibles a realizar a continuación. Llamemos n a este número. Entonces, cualquier computación de M_1 se puede representar mediante secuencias finitas de números entre 1 y n. Por supuesto, no toda secuencia así será válida, porque habrá situaciones con menos de n movimientos posibles.

En M_2, la primera cinta almacena la cadena de entrada. En la segunda cinta se generan sistemáticamente secuencias de números entre 1 y n. Para cada secuencia, M_2 copia la cadena de entrada en la tercera cinta y simula M_1 sobre esta cinta, usando la secuencia de operaciones de la segunda cinta como guía. Si una de las secuencias hace que M_1 acepte la cadena, entonces M_2 también aceptará. Si ninguna de las secuencias provoca la aceptación por parte de M_1, tampoco M_2 aceptará.

Hemos definido la equivalencia en términos de la aceptación de lenguajes, pero se podría extender este razonamiento para simular una MT que realice cualquier otra operación.

4 Modificaciones de las máquinas de Turing

Existen otras muchas modificaciones, pero todas ellas son equivalentes a la original.

Como ya hemos dicho, lo único que aportan estas definiciones alternativas es más facilidad a la hora de especificar ciertas tareas.

Por poner algún ejemplo más, indicaremos que:

- En general, las MT,s con múltiples pistas sirven para realizar reconocimiento de cadenas de forma no destructiva.
- Mientras que las MT,s multicabeza sobre la misma cinta simulan los modelos computacionales de paralelismo y concurrencia.

Contenidos

5 Máquina de Turing universal

Definición

La **máquina de Turing universal** es aquélla que, dada una descripción adecuada de cualquier MT M y de cualquier cadena de entrada w, simula el comportamiento de M sobre w.

Describiremos ahora esta MT universal, que será necesaria para estudiar la jerarquía de los lenguajes y la resolubilidad, y fortalecerá la conexión existente entre las MT,s y los ordenadores.

Definición

La **máquina de Turing universal** es aquélla que, dada una descripción adecuada de cualquier MT M y de cualquier cadena de entrada w, simula el comportamiento de M sobre w.

Describiremos ahora esta MT universal, que será necesaria para estudiar la jerarquía de los lenguajes y la resolubilidad, y fortalecerá la conexión existente entre las MT,s y los ordenadores.

Lo primero que necesitamos es definir la **codificación de una MT arbitraria**:

- En primer lugar, se requiere que la MT tenga un único estado final p. Si hubiera más, el estado p será un estado extra al que se pasa desde todos los estados finales originales q, dejando la cabeza de L/E en la misma posición que estaba cuando llegó a q.

5 Máquina de Turing universal

Definición

La **máquina de Turing universal** es aquélla que, dada una descripción adecuada de cualquier MT M y de cualquier cadena de entrada w, simula el comportamiento de M sobre w.

Describiremos ahora esta MT universal, que será necesaria para estudiar la jerarquía de los lenguajes y la resolubilidad, y fortalecerá la conexión existente entre las MT,s y los ordenadores.

Lo primero que necesitamos es definir la **codificación de una MT arbitraria**:

- En primer lugar, se requiere que la MT tenga un único estado final p. Si hubiera más, el estado p será un estado extra al que se pasa desde todos los estados finales originales q, dejando la cabeza de L/E en la misma posición que estaba cuando llegó a q.
- Supondremos entonces que $Q = \{q_1, q_2, \ldots, q_n\}$, donde q_1 es el estado inicial y q_2 el único estado final, y que $\Gamma = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$, donde σ_1 es el blanco. De esta forma, cualquier MT quedará completamente especificada por medio de su función de transición.

Definición

La **máquina de Turing universal** es aquélla que, dada una descripción adecuada de cualquier MT M y de cualquier cadena de entrada w, simula el comportamiento de M sobre w.

Describiremos ahora esta MT universal, que será necesaria para estudiar la jerarquía de los lenguajes y la resolubilidad, y fortalecerá la conexión existente entre las MT,s y los ordenadores.

Lo primero que necesitamos es definir la **codificación de una MT arbitraria**:

- En primer lugar, se requiere que la MT tenga un único estado final p. Si hubiera más, el estado p será un estado extra al que se pasa desde todos los estados finales originales q, dejando la cabeza de L/E en la misma posición que estaba cuando llegó a q.
- Supondremos entonces que $Q = \{q_1, q_2, \ldots, q_n\}$, donde q_1 es el estado inicial y q_2 el único estado final, y que $\Gamma = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$, donde σ_1 es el blanco. De esta forma, cualquier MT quedará completamente especificada por medio de su función de transición.
- Por tanto, para codificar la MT, es suficiente con codificar δ. Para ello, representamos q_i mediante i unos. De igual forma, representamos σ_i también mediante i unos. Finalmente, para los movimientos de la cabeza, representamos L con 1 y R con 11. Entonces, si utilizamos el símbolo 0 como separador, una transición como $\delta(q_3, \sigma_1) = (q_4, \sigma_3, L)$ podría representarse como 011101011110111010.

5 Máquina de Turing universal

Definición

La **máquina de Turing universal** es aquélla que, dada una descripción adecuada de cualquier MT M y de cualquier cadena de entrada w, simula el comportamiento de M sobre w.

Describiremos ahora esta MT universal, que será necesaria para estudiar la jerarquía de los lenguajes y la resolubilidad, y fortalecerá la conexión existente entre las MT,s y los ordenadores.

Lo primero que necesitamos es definir la **codificación de una MT arbitraria**:

- En primer lugar, se requiere que la MT tenga un único estado final p. Si hubiera más, el estado p será un estado extra al que se pasa desde todos los estados finales originales q, dejando la cabeza de L/E en la misma posición que estaba cuando llegó a q.
- Supondremos entonces que $Q = \{q_1, q_2, \ldots, q_n\}$, donde q_1 es el estado inicial y q_2 el único estado final, y que $\Gamma = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$, donde σ_1 es el blanco. De esta forma, cualquier MT quedará completamente especificada por medio de su función de transición.
- Por tanto, para codificar la MT, es suficiente con codificar δ. Para ello, representamos q_i mediante i unos. De igual forma, representamos σ_i también mediante i unos. Finalmente, para los movimientos de la cabeza, representamos L con 1 y R con 11. Entonces, si utilizamos el símbolo 0 como separador, una transición como $\delta(q_3, \sigma_1) = (q_4, \sigma_3, L)$ podría representarse como 01110101111011101 0.

Es decir, cualquier MT se puede codificar mediante la secuencia de ceros y unos de todos sus arcos. Y dada una de estas codificaciones, podemos también decodificar la MT en cuestión.

5 Máquina de Turing universal

Para la MT universal M_U, cuyo alfabeto estará formado por ceros y unos, se puede entonces realizar la siguiente **implementación mediante una MT de tres cintas**:

- La 1ª cinta contendrá la codificación de una MT. La cabeza de L/E de esta cinta estará sobre el 0 inicial.
- La 2ª cinta contendrá la codificación de la cinta de dicha MT, es decir, los símbolos que estarían en la cinta de la MT que se está simulando, pero codificados y separados por ceros. La cabeza de L/E de esta cinta estará sobre el 0 que precede al símbolo actual.
- La 3ª cinta contendrá en todo momento la codificación del estado actual. Inicialmente, contendrá un 1 rodeado de blancos, con la cabeza de L/E posicionada en él.

5 Máquina de Turing universal

Para la MT universal M_U, cuyo alfabeto estará formado por ceros y unos, se puede entonces realizar la siguiente **implementación mediante una MT de tres cintas**:

- La 1ª cinta contendrá la codificación de una MT. La cabeza de L/E de esta cinta estará sobre el 0 inicial.
- La 2ª cinta contendrá la codificación de la cinta de dicha MT, es decir, los símbolos que estarían en la cinta de la MT que se está simulando, pero codificados y separados por ceros. La cabeza de L/E de esta cinta estará sobre el 0 que precede al símbolo actual.
- La 3ª cinta contendrá en todo momento la codificación del estado actual. Inicialmente, contendrá un 1 rodeado de blancos, con la cabeza de L/E posicionada en él.

M_U analiza y compara los contenidos de la 2ª y de la 3ª cinta con el contenido de la primera, en busca de una transición correspondiente a la codificación del estado y del símbolo actuales:

- Si la encuentra, M_U se comporta como la MT original, actualizando el estado y la cinta.
- Si no la encuentra, M_U se para, como lo haría también la MT original.

5 Máquina de Turing universal

Para la MT universal M_U, cuyo alfabeto estará formado por ceros y unos, se puede entonces realizar la siguiente **implementación mediante una MT de tres cintas**:

- La 1ª cinta contendrá la codificación de una MT. La cabeza de L/E de esta cinta estará sobre el 0 inicial.
- La 2ª cinta contendrá la codificación de la cinta de dicha MT, es decir, los símbolos que estarían en la cinta de la MT que se está simulando, pero codificados y separados por ceros. La cabeza de L/E de esta cinta estará sobre el 0 que precede al símbolo actual.
- La 3ª cinta contendrá en todo momento la codificación del estado actual. Inicialmente, contendrá un 1 rodeado de blancos, con la cabeza de L/E posicionada en él.

M_U analiza y compara los contenidos de la 2ª y de la 3ª cinta con el contenido de la primera, en busca de una transición correspondiente a la codificación del estado y del símbolo actuales:

- Si la encuentra, M_U se comporta como la MT original, actualizando el estado y la cinta.
- Si no la encuentra, M_U se para, como lo haría también la MT original.

En definitiva, si la MT original para con w como cadena de entrada, M_U también parará con las codificaciones de la MT y de w como entradas. Y además:

- La 3ª cinta contendrá la codificación del último estado alcanzado.
- Y la 2ª contendrá la codificación de la cinta tal y como la habría dejado la MT original.

Fin del capítulo

"Máquinas de Turing"