Compiladores y lenguajes formales

Máquinas de Turing

Contenido

1. Concepto de computabilidad

Una vez se han realizado en el compilador las tres fases de análisis (léxico, sintáctico y semántico) empiezan las fases de síntesis, que se concretan en la generación de código para una máquina objetivo. Para llegar a generar el código objetivo, que es el resultado final de todo el proceso de compilación, deberemos haber optimizado ese código a generar y esto se ha realizado a partir de un código abstracto que sirve para cualquier máquina y que se denomina código intermedio.

Un problema se dice que es computable si se puede resolver mediante un programa de ordenador (mediante un cómputo).

Para profundizar sobre el concepto, conviene definir primero algunos elementos.

Por ejemplo, resolver un problema es convertir unos datos de entrada, en unos resultados de salida. De hecho, cualquier problema se puede modelar mediante este enfoque. A veces, los datos de entrada serán números y otras no, pero siempre se pueden codificar de manera numérica. Por ejemplo, para resolver el problema "¿vacaciones en la playa o en la montaña?", una familia decidirá qué características son favorables y desfavorables (datos de entrada) a cada opción y qué mecanismo elegirá para decidir al final dónde ir (datos de salida). Estas decisiones – qué tener en cuenta, quien valora y con qué reglas – pueden no ser explícitas en la decisión (es difícil de imaginar una familia tan racional), pero aunque sea de forma oculta, realmente lo que se decidirá es el valor de la función "ir de vacaciones". Esta función se puede codificar como una función donde cada elemento de entrada sea un número (positivo, a favor, negativo, en contra de cada opción de salida) y la salida indique el destino final (playa o montaña).

Existen muchos mecanismos que resuelven problemas. Algunos de estos mecanismos resuelven un tipo de problemas, pero no consiguen resolver los otros tipos. De hecho, los problemas se pueden clasificar según el tipo de mecanismo que puede resolverlos. Pero ¿existen problemas que no se puedan resolver con ningún mecanismo? Si existe un problema de este tipo, un problema irresoluble, este problema no será tampoco computable, es decir, no podremos realizar un programa informático que lo resuelva mediante un cómputo.

Para responder a esta inquietante pregunta (ningún informático querría empezar un programa para un problema irresoluble, porque fracasaría), conviene profundizar en la esencia de los problemas.

Todos los problemas se pueden reformular como un problema de lenguajes. Es decir, si construyo una gramática que genera el lenguaje formado por todas las soluciones a un problema, el problema queda resuelto. Y viceversa: si un problema se puede resolver, puede construir una solución que consista en la construcción del lenguaje de sus soluciones. Así, una aproximación es pensar en la generación de lenguajes como una proyección formal de la resolución de problemas.

Desde este enfoque, los tipos de problemas se pueden clasificar según los tipos de gramáticas que generan el "lenguaje que resuelve" el problema. Por ejemplo, Noah Chomsky propone una jerarquía de gramáticas, llamada *jerarquía de Chomsky* (figura 1), que sirve para clasificar los lenguajes en varios tipos, según el tipo de gramática que los genera.

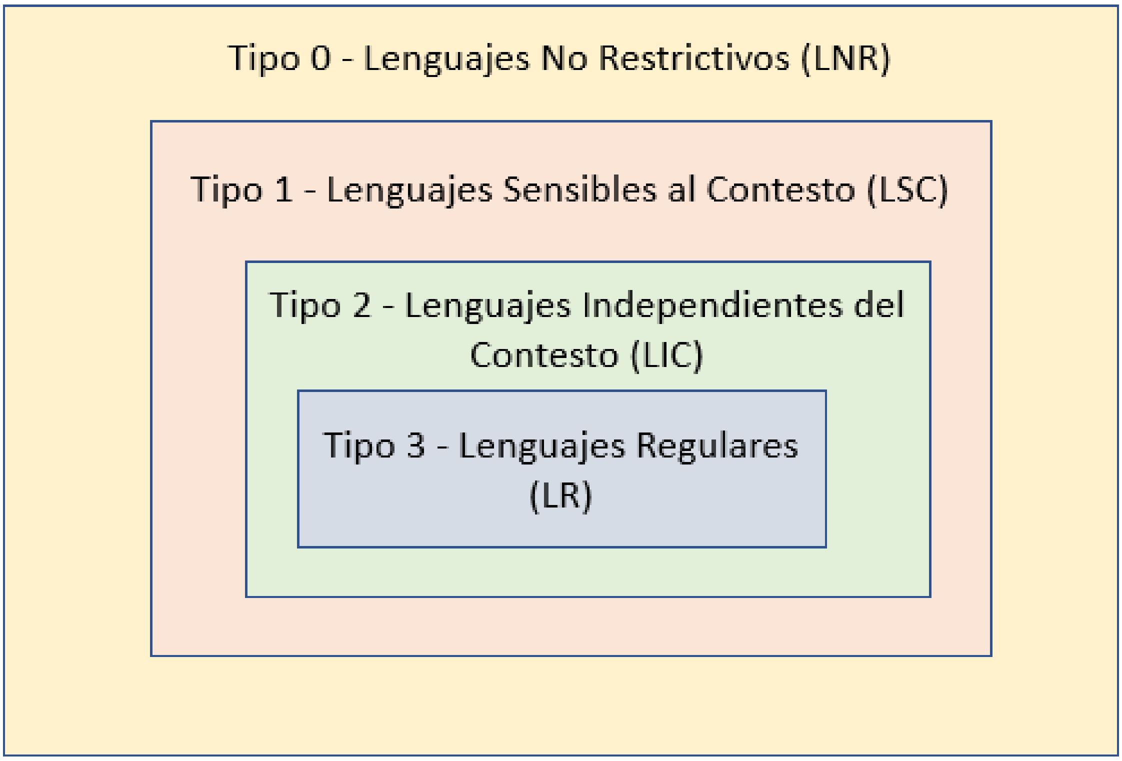


Figura1. Tipos de lenguajes en la jerarquía de Chomsky

En esta jerarquía, cada clase de lenguajes es generado por un tipo de gramática, es decir, por una gramática cuyas reglas de producción cumplen unas determinadas restricciones. Como se aprecia en la figura, cada clase de lenguajes incluye propiamente a la clase que le precede, de manera que, por ejemplo, todos los lenguajes LR (tipo 3) son también LIC (tipo 2), pero existen lenguajes LIC que no son LR.

Se sugiere al lector que busque las restricciones asociadas a cada tipo de lenguaje para comprender un poco mejor los tipos de lenguajes, y por lo tanto, los tipos de problemas.

Las máquinas de Turing son, como se verá, un mecanismo formal que sirve para reconocer una inmensa cantidad de lenguajes, o lo que es lo mismo, resolver muchos problemas.

La "mala notica" es que existen problemas no resolubles con máquinas de Turing y, según la "tesis de Turing", también existen problemas que no son resolubles con ningún mecanismo, es decir, no son computables, no podremos hacer un programa informático en ningún lenguaje de programación capaz de resolverlo.

De esto trata este documento.

2. Definición de la Máquina de Turing

La Máquina de Turing (MT) es una máquina teórica formada por:

1. Una cinta infinita (de ahí su carácter teórico) formada por un conjunto celdas. Cada celda puede almacenar un símbolo.

2. Una cabeza lectora-escritora que apunta a una celda, y puede moverse a derecha y a izquierda y acceder a las celdas. En cada movimiento, lee y escribe un símbolo.

3. Una unidad de control que, atendiendo a la información recibida desde la cabeza lectora-escritora y a la configuración actual de la MT, produce un "movimiento de cómputo". Este movimiento incluye qué escribir en la cinta, hacia dónde mover la cabeza lectora—escritora y cómo reconfigurar la máquina para el siguiente movimiento.

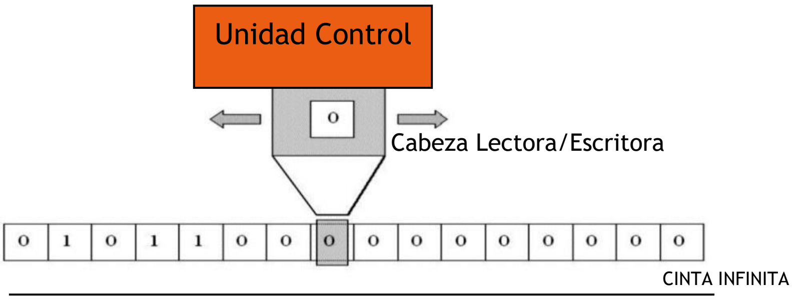


Figura 2. Representación gráfica de una MT genérica.

Formalmente, una MT M, se puede definir como M= ($Q,\Sigma,\Gamma,q0$, #, F,δ), donde:

Q: Conjunto de estados

Σ: Alfabeto de entrada

Γ: Alfabeto de la cinta $\Sigma\subseteq\Gamma$- {b}

q_0: Estado inicial

#: carácter "blanco" $\#\in\Gamma$, $\# \notin \Sigma$

F: Estados finales $F\subseteq Q$

δ: Función de transición, δ: $Q x\Gamma \rightarrow Q x\Gamma x\{L,R\}$, donde L y R representan un desplazamiento de la cinta a la izquierda (L) o derecha (R)

3. Función de transición

δ: Q x Γ → Q x Γ x {L,R}

Ejemplo: $\delta(q_0, a) = (q_1, b, R)$

Cada transición como la del ejemplo representa:

Si la máquina está en el estado q0 y el contenido de la celda a la que apunta actualmente la cabeza es una “a”, hacer:

1. Escribe una b en la cinta
2. Desplazar la cabeza a la derecha (R)
3. Transitar la máquina al estado q1

4. Diagrama de transición

Para representar una MT y facilitar la comprensión de su funcionamiento, a menudo se utilizan los **diagramas de transición**.

Un diagrama de transición es un grafo donde:

- El conjunto de nodos del grafo se corresponde a los estados de la MT.
- En un arco que vaya del estado p al q, aparecerán una o varias etiquetas de la forma X/YS, donde X e Y son símbolos de cinta, y S indica un movimiento de la cabeza lectora/escritora, es decir, puede ser L o R.
- A menudo, el sentido S, se representa visualmente utilizando "→" para la derecha y "←" para la izq.
- El símbolo que se utiliza para denotar el espacio en blanco es B si no se especifica lo contrario.
- Cuando no se puede aplicar ninguna transición, se dice que la MT está parada.
- La secuencia de todos los movimientos que conducen a una configuración de parada se llama "computación".
- Los estados finales son "de parada", en general.

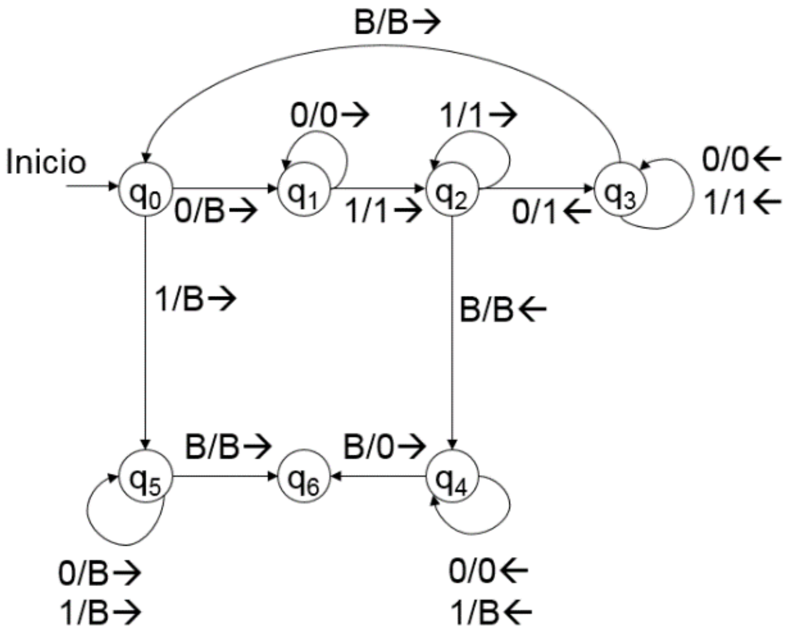


Figura 3. Ejemplo de diagrama de transición de una MT de 7 estados.

5. Aceptación de una palabra en una máquina de Turing

El funcionamiento de una MT para reconocer una palabra w es el siguiente:

1. Colocar la palabra w en la cinta de la MT.
2. Colocar la cabeza de lectura/escritura en el primer símbolo o justo en el de la izquierda del primer símbolo. Es decir, hay que colocar la cabeza en un lugar conocido y "cerca" del comienzo de la palabra w que se quiere estudiar si pertenece o no al lenguaje aceptado pro esa máquina de Turing.
3. Poner la MT en marcha desde el estado inicial, q0

"Poner en marcha" la máquina M significa iniciar los pasos de cómputo posibles según su función de transición. Si la máquina está representada por un diagrama de transición, hacer funcionar M quiere decir recorrer en todo momento los arcos que son posibles, teniendo en cuenta la posición de la cabeza de lectura/escritura (que está leyendo la máquina en su cinta) y el estado actual en el que se encuentra M. Co mesa información de entrada, el arco que se recorre indica a qué nuevo estado llegará M, qué debe escribir en la cinta de entrada/salida y hacia dónde debe desplazar la cabeza lectora/escritora.

La máquina, en un momento determinado del cómputo, puede encontrar dos caminos posibles. En ese caso, la máquina decide de forma no determinista, es decir, a priori el modelo no explica qué camino recorrerá efectivamente. Esto, en la práctica, supone que la máquina toma todos los caminos posibles en cada momento, y realiza los cómputos asociados a cada camino de forma paralela.

Si la MT no encuentra, en un momento dado, ningún camino posible que recorrer, es decir, la función de transición no está definida para ese par "estado, letra) o, si se conoce un diagrama de transición para dicha máquina M, no hay ningún narco elegible que salga del estado actual de M, se dice que la máquina ha alcanzado un estado de parada y el cómputo se detiene en ese momento.

La palabra será aceptada si al menos uno de los caminos posibles, uno de los cómputos, consigue alcanzar una configuración de parada sobre un estado de aceptación.

- Definición: Sea M= $(Q,\Sigma,\Gamma,q_0,b, F,\delta)$ una MT. Se define el lenguaje aceptado por M, como:

  $L(M) = \{w \in \Sigma^* \mid q_0w \mid\!-^* w_1pw_2,\ p \in F,\ w_1,w_2 \in \Gamma^* \wedge M \text{ está parada}\}$

  - Una cadena se rechaza si no puede llegar a un estado final y parar en él.

- Si la MT con una cadena w como entrada, entra en un bucle infinito, la palabra no formará parte del lenguaje L(M) porque, al no parar nunca, no producirá una aceptación.

6. Aceptación de una palabra en una máquina de Turing

Las máquinas de Turing son herramientas teóricas muy potentes para la aceptación de lenguajes y, por lo tanto, para la solución de problemas. No obstante, es posible que modificando alguno de sus aspectos, podamos ganar en eficiencia, haciendo lo mismo para más sencillo, o tal vez incluso en eficacia, consiguiendo reconocer más lenguajes que con la máquina de Turing clásica.

Son muchas las modificaciones que se pueden hacer al concepto de máquina de Turing, redefiniendo alguna de los elementos que la constituyen, es decir, modificando alguno de estos aspectos, como, por ejemplo:

1. Modificaciones en sus movimientos de la cabeza lectura/escritora
2. Modificando su cinta de entrada/salida
3. Modificando el número de cintas de entrada/salida y, por lo tanto, modificando el número de cabezas de lectura/escritura.

6.1. Modificaciones en el movimiento de la cabeza

Si M es una MT, cada vez que actúa su función de transición, la cabeza de lectura/escritura se tiene que desplazar bien una celda a la izquierda (L) o bien una celda a la derecha (R), pero se puede modificar la definición de máquinas de Turing para permitir una tercera posibilidad: no mover la cabeza.

- = $(Q,\Sigma,\Gamma,q_0,b, F,\delta)$

En esta máquina modificada la función de transición toma esta forma:

- δ: Q x Γ → Q x Γ x {R,L,S}
- $\delta(q,a) = (p,a',S)$

Este nuevo movimiento S (Stop) resulta muy útil para simplificar MT, ya que permite cambiar de estado o reescribir una celda de la cinta sin necesidad de desplazar la cabeza.

Es fácil comprobar que esta modificación no produce más potencia de cómputo, es decir, que estas nuevas máquinas no hacen nada que no fueran capaces de hacer las tradicionales, ya que este nuevo movimiento S se puede conseguir en una máquina convencional mediante un doble paso de cómputo, uno dejando la cinta exactamente igual y desplazando la cabeza a la derecha con un movimiento R, para justo a continuación hacer un movimiento (de nuevo, sin tocar la cinta) y ejecutando un movimiento L. La combinación de ambos movimientos, RL, hace que la cabeza se haya quedado en la misma posición que estaba al comenzar. Si se ajusta bien el estado en el que debe terminar este doble movimiento, se simula sin problemas el nuevo movimiento S.

6.2. MT con cinta semi-infinita

La cinta de una MT es infinita por ambos extremos. Como consecuencia, los movimientos de la cabeza lectura/escritora no tiene que preocuparse de si encontrarán una celda que leer/escribir o se saldrán de la cinta. Eso facilita mucho la construcción teórica de la función de transición.

Otra consecuencia, menos favorable, es que no existe una “primera celda” en la cinta de la máquina. Si existiera dicha primera celda, el cómputo de la máquina podría iniciar en ella y volver a esa posición con mucha facilidad.

Para aprovechar esta ventaja, se propone una modificación de la máquina de Turing consistente en limitar la cinta por uno de sus lados:

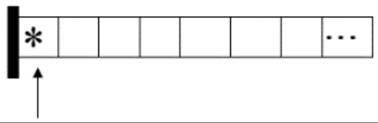


Figura 4. Representación de la cinta de entrada/salida de un MT con cinta semi infinita.

Es evidente que esta modificación no supone un aumento en la capacidad de cómputo, es decir, todos los lenguajes reconocidos por esta MT modificada serán también reconocibles con una MT convencional. Lo único que tiene que hacer esa MT convencional es dejar una sección de cinta sin utilizar a la izquierda y “marcar” como primera celda la más a la izquierda de las que usa.

6.3. MT con múltiples pistas

De nuevo, una modificación sobre la naturaleza de la cinta de la máquina consiste en permitirla tener más de una letra en cada celda. Es decir, una cinta con varias pistas, de forma que la cabeza de lectura/escritura puede leer o escribir de una sola vez, no una letra, sino un vector de letras, una letra en cada pista.

Ejemplo: M.T. de 3 pistas:

a	b	b	a	a	B	B	B
a	b	b	a	a	B	B	B
a	b	a	a	a	B	B	B

Figura 5. Representación de la cinta de entrada/salida de un MT con cinta semi infinita.

En el ejemplo, la cabeza lectora/escritora lee la terna (b, b, a).

En una MT de k-pistas el alfabeto de cinta está formado por todas las k-tuplas posibles con las letras del alfabeto de cinta, incluyendo, naturalmente, el símbolo de cinta vacía.

Esta modificación de la MT permite realizar operaciones donde intervengan varios datos de entrada con mayor rapidez, ya que se pueden leer todos en una sola operación. No obstante, no supone un aumento en el poder de cómputo: todo lo que puede hacer una MT con k-pistas, se puede realizar también con una MT de una sola pista.

Para simular con una sola pista, muchas pistas, basta con aprovechar que la cinta de la MT convencional es infinita. Así, se puede construir una MT que utilice k celdas de su cinta por cada celda con k-pistas de la MT que se pretende simular. El trabajo en una sola pista puede ser mucho más tedioso, pero el resultado será el mismo.

6.4. MT con cinta plana

Una modificación posible de la cinta, mucho más compleja, es pensar en una cinta que sea infinita en dos dimensiones, es decir, que en lugar de una cinta lo que tenga la máquina sea un plano como en la figura:

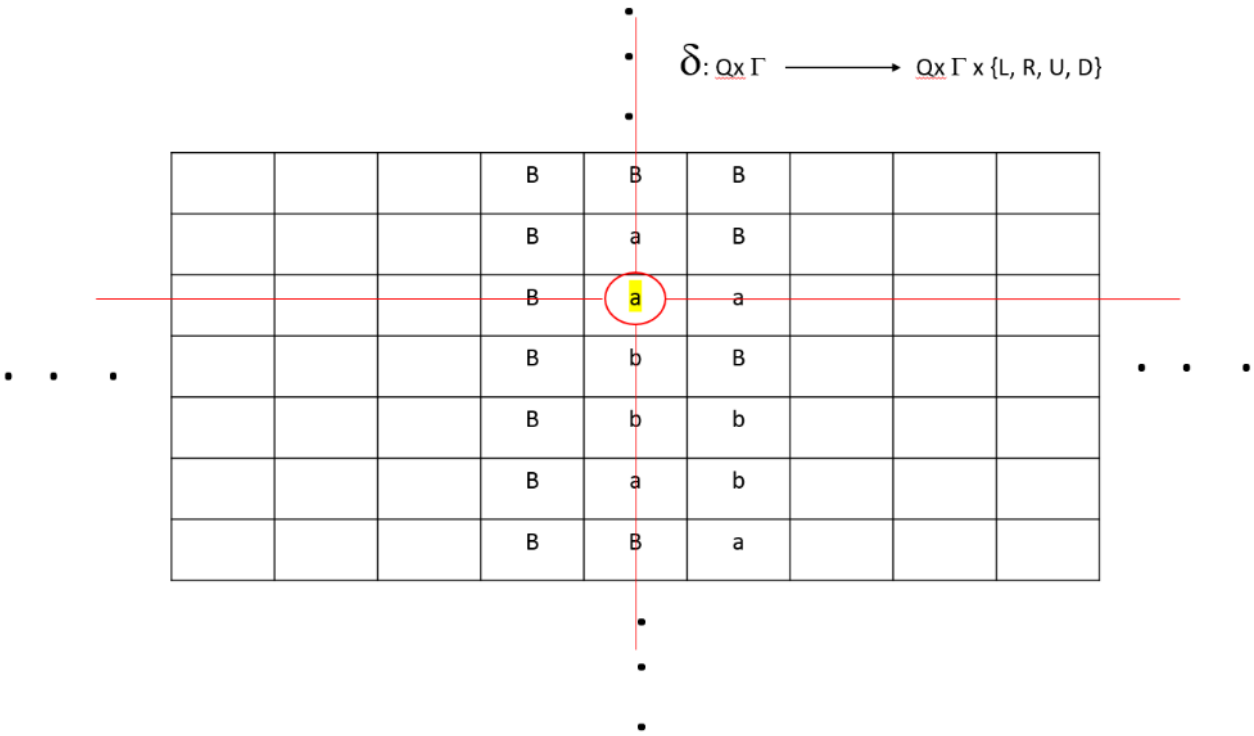


Figura 6. Representación de la cinta MT con cinta plana

En esta MT, la función de transición debe incluir nuevos movimientos que la permitan desplazarse a la izquierda (L), a la derecha (R), hacia arriba (U) y hacia abajo (D). A menudo, se utiliza para estos movimientos los nombres de los puntos cardinales en un mapa (en inglés): N, S, E, W.

Los movimientos en esta MT son de la forma:

1. Cambia el estado según la letra leía en la celda donde apunta la cabeza de lectura/escritura.

2. Escribe un nuevo símbolo en dicha celda.

3. Desplaza la cabeza según el movimiento indicado en la función de transición. Puede subir una celda, bajar, ir a la izquierda o a la derecha.

La MT convencional sería un caso particular en esta nueva máquina, ya que se puede simular con una MT con un plano como cinta sin utilizar nunca los movimientos U, D. Es decir, este nuevo tipo de máquina tiene al menos el mismo poder de cómputo que la MT convencional.

La equivalencia en sentido contraria, esto es, demostrar que el comportamiento de una MT con un plano como cinta se puede simular con una MT convencional también es posible, aunque algo más complejo.

Basta darse cuenta de que podemos simular este plano, con cierta facilidad, con una cinta de tres pistas. En la pista 1, aparece el contenido de una celda del plano, y en las pistas 2 y 3, las coordenadas de dicha celda en el plano. Así, simular un movimiento U del plano en esta cinta de tres pistas, consiste en mover la cabeza lectura/escritora desde su posición actual (digamos que está en la posición (n, m) representada con una “n” en la pista 2 y una “m” en la pista 3) hasta la posición de la cinta que tenga (n,m+1) en sus cintas 2 y 3. Un movimiento D, equivale a buscar el vector (n,m-1), a la derecha, vector (n+1, m) y a la izquierda, (n-1,m), siempre leyendo las pistas 2 y 3.

Así, esta modificación de la MT tampoco consigue mayor poder de cómputo.

6.5. MT con múltiples cintas

Todas las modificaciones propuestas hasta ahora han mantenido la unicidad de la cinta de la maquina.

Otra modificación posible es dotar a la máquina de varias cintas independientes, con una cabeza de lectura/escritura cada una de ellas:

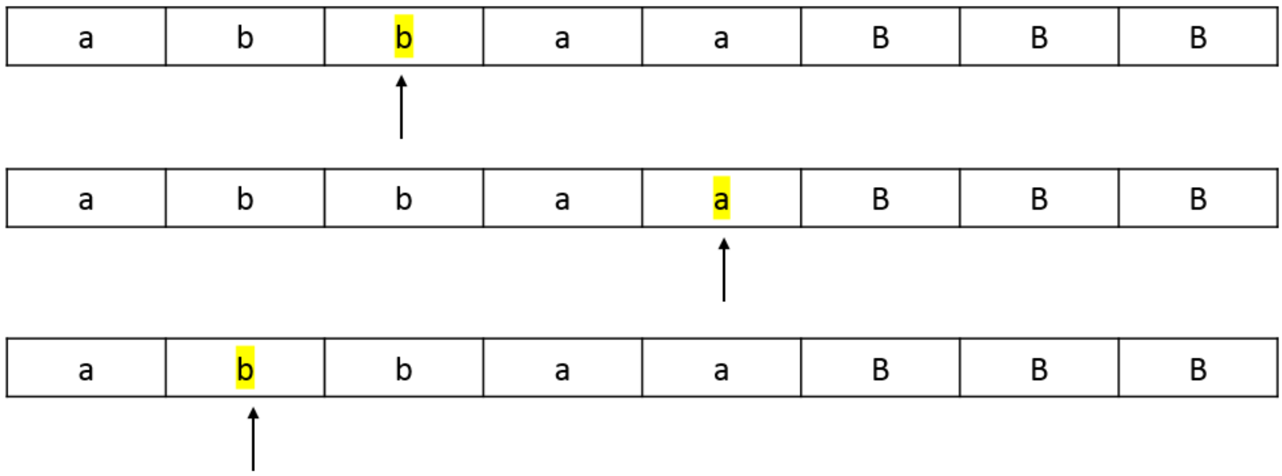


Figura 7. Representación de las cintas MT de tres cintas con tres cabezas de lectura/escritura

En esta máquina, en cada movimiento se realizan las siguientes acciones:

1. Cambia de estado dependiendo del estado actual y del contenido de todas las cintas.
2. Escribe un nuevo símbolo en cada una de las n celdas donde apuntan las n cabezas
3. Mueve cada cabeza de forma independiente. Así, la cabeza de la cinta 1 puede desplazarse a la izquierda (L1) y la cinta2 a la derecha (R2)

Es decir, en esta MT la función de transición es de esta forma:

$$\delta: Q x\ \Gamma^n \longrightarrow Q x\ \Gamma^n\ x\ (\{L, R\})^n$$

Claramente, una MT convencional es un caso particular de esta modificación, con n=1, así que todo lo que se podía hacer antes se puede realizar ahora.

Para demostrar que al revés igual, que todo lo que puede hacer una MT con n cintas lo puede hacer también una MT con una sola conta, resulta útil utilizar las MT de una cinta con varias pistas.

Por ejemplo, para simular M, MT de 3 contas, podemos usar M', MT con una cinta y 7 pistas (si tuviera n cintas, usaríamos, 2n+1 pistas).

En las pistas 1,3 y 5, se coloca el mismo contenido que en las tres cintas de M. Las pistas 2, 4 y 6, se dejan en blanco salvo una celda con un símbolo que representa la posición de la cabeza correspondiente en M. Por último, en la pista 7, se colocan dos símbolos, uno indicando la posición de la cabeza más a la izquierda, y otra identificando la posición de la cabeza más a la derecha, como se muestra en la figura 8:

B	b	b	a	a	B	B	B
		↑					
B	b	a	a	a	B	B	B
			↑				
B	b	b	a	a	B	B	B
	↑						
*				*			

Figura 8. Cinta de 7 pistas que simula una MT de tres cintas.

El comportamiento de M' para simular el de M sería así:

1. M' desplaza hacia la izquierda, sin tocar nada en la cinta, su cabeza lectura hasta que encuentra en la pista 7 el símbolo "*". Es decir, coloca su cabeza de lectura/escritura a la izquierda de todas las posiciones de las cabezas de la máquina M.
2. Desplaza su cabeza a la derecha una posición→ seguro que está sobre un símbolo de cabeza de lectura en la pista 2,4 o 6.
3. M' lee de golpe el vector de siete componentes que le corresponda. Busca en las cintas pares dónde hay un símbolo de cabeza y modifica el contenido de la cinta impar correspondiente para simular a la máquina M. Luego, borra el símbolo que

representa la posición de una de las cabezas de M y la copia o a la izquierda o a la derecha, según lo que hiciera M. Luego, vuelve a esta posición en su cinta.

4. M' se desplaza hacia la derecha hasta que encuentre una de estas dos cosas:
   a. El símbolo que indica que ha encontrado la posición de otra cabeza de M → actúa sobre la cinta correspondiente para simular el comportamiento de M en una de sus cintas.
   b. El símbolo * en la pista 7 → significa que ya ha procesado la cabeza de más a la derecha, es decir, ya ha procesado el movimiento de M al completo.

De esta forma, una máquina de una sola cinta puede simular a las máquinas que tienen cualquier número de cintas.

7. Otros formalismos para decidir la computabilidad

Existen otros formalismos, como las máquinas de registros ilimitados (MRI), que sirven para abordar la computación de un problema.

Una MRI es una máquina teórica con un conjunto infinito de registros y una colección muy corta de acciones posibles. Por ejemplo, las acciones pueden ser:

1. Asignar un valor a un registro.
2. Restar uno al valor del registro.
3. Comprobar si un registro es cero.
4. Sumar uno a un registro.
5. Provocar un salto a otra posición (instrucción) del código de la máquina.

Existen varias modificaciones de estas instrucciones que dan como resultado distintas definiciones de MRI.

Es fácil comprobar que cualquier problema resuelto mediante una MRI se puede también resolver con una MT. La demostración de esta afirmación es sencilla y consiste en modelar en una MT cada una de las instrucciones posibles en una MRI.

Así, cualquier problema resuelto por una MRI se puede resolver también con una máquina de Turing.

8. Tesis de Turing

Como ya se ha dicho, existen otros formalismos para resolver problemas. Y, como también se ha visto, las máquinas de Turing se pueden modificar para construir otras similares. Pero, analizando su capacidad para resolver problemas, resulta que no se conoce ningún formalismo más potente que las máquinas de Turing.

Es decir, todas las modificaciones propuestas de la máquina de Turing, y todos los mecanismos conocidos para resolver problemas, resuelven, como mucho, la misma clase de problemas que la máquina de Turing primitiva.

Esto lleva a postular la Tesis de Turing:

Toda función computable es Turing computable.

Según esta tesis, por lo tanto, estudiar la Turing-computabilidad de un problema o de un lenguaje, es lo mismo que estudiar su computabilidad.

Aprender a construir máquinas de Turing se convierte así en sinónimo de aprender a resolver cualquier problema que pueda resolverse. Y aprender a distinguir entre los problemas para los que se puede realizar una MT y los que no, resulta útil para decidir si intentar o no hacer un programa informático que lo resuelva: si no es Turing-computable, no podremos programar su solución con ningún lenguaje ni mecanismo.

9. Construcción de Máquinas de Turing

La construcción de MT usando su definición y los gráficos de transición, resulta muy laboriosa. Para solucionar en parte este problema, la idea es construir una colección de MT sencillas y combinarlas para construir otras MT más complejas.

El mecanismo para esta combinación de máquinas es el siguiente:

- Comparten la misma cinta.
- Cuando una termina su ejecución, empieza la otra.
- El contenido de la cinta cuando empieza la segunda es el contenido con el que acabó la primera.
- La cabeza de L/E de la segunda comenzará sobre la celda en la que terminó la primera.
- Ambas MT tienen el mismo alfabeto de entrada y el mismo alfabeto de cinta.

Formalmente sería:

Definición: Sean M1, M2 dos MMT sobre el mismo alfabeto de entrada Σ y el mismo alfabeto de cinta Γ.

$M_1 = (Q_1,\Sigma,\Gamma,q_{01},b,F_1,\delta_1)$

$M_2 = (Q_2,\Sigma,\Gamma,q_{02},b,F_2,\delta_2)$ con $Q_1 \neq Q_2$

La combinación o composición de M_1 y M_2 se representa por M_1M_2

$M_1M_2 = (Q,\Sigma,\Gamma,q_0,b,F,\delta)$

donde:

$Q=Q_1\cup Q_2$

$q_0=q_{01}$

$F=F_2$

$$\delta(q,\sigma) = \begin{cases} \delta_1(q,\sigma), \text{ si } q\in Q_1 \text{ y } \delta_1(q,\sigma)\neq(p,\tau,X)\ \forall p\in F_1 \\ \delta_2(q,\sigma), \text{ si } q\in Q_2 \\ (q_{02},\tau,X) \text{ si } q\in Q_1 \text{ y } \delta_1(q,\sigma)=(p,\tau,X) \text{ para algún } p\in F_1 \end{cases}$$

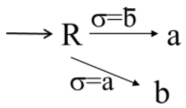
La colección clásica de máquinas de Turing sencillas útiles para realizar otras más complejas son:

- R, L: desplazan la cabeza a la derecha / izquierda.
- R_b: Busca el primer “blanco” a la derecha de la cabeza lectora.
- R_x: Busca el primer símbolo “x” a la derecha.
- L_x: Busca el primer símbolo “x” a la izquierda.
- R_xR_x: Combinación que busca el segundo símbolo “x” a la derecha.
- x: Escribe “x” y deja la cabeza en la misma celda. Primero escribe con desplazamiento y luego regresa.
- L_x: Busca el primer símbolo distinto de x a la izquierda de la cabeza lectora.
- R^2= RR: 2 desplazamientos a la derecha.

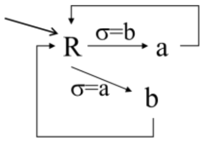
Con estas máquinas simples, se pueden construir otras que realizan operaciones más complejas.

Ejemplos:

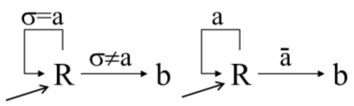
a. Si a la derecha de la cabeza hay una “a”, sustituirla por una “b”. si a la derecha hay un blanco, escribir una “a”



b. Sustituir las “a” por “b” y viceversa.



c. Escribir una “b” al final de la primera cadena de “a” de la entrada. En este ejemplo se la misma máquina con la notación usada hasta ahora y con una nueva notación más simple, aunque con el mismo significado.



10. Máquina universal de Turing

Las máquinas de Turing encierran todo el poder del cómputo. Si queremos resolver un problema (que tenga solución), podemos crear una MT que lo consiga. Y, según la tesis de Turing, si somos capaces de demostrar que no hay una MT que lo resuelva, es que no tiene solución computable.

Dicho esto, conviene reflexionar un poco más:

¿Podemos crear una máquina de Turing que pueda resolver todos los problemas que pueden resolver todas las demás máquinas de Turing?

La respuesta es, si podemos. Podemos crear una máquina de Turing M que reciba como entrada, de alguna forma, el código que represente a otra MT M' y el código de una entrada para esa MT codificada, w. Esta M se puede comportar exactamente igual que la máquina M' lo haría con w como entrada. Codificar M' consiste fundamentalmente en codificar su función de transición completa, de manera que sea posible identificar cada valor de la función para cada par (p,a) para el que este definida, donde q es un estado de Q y a una letra del alfabeto de cinta.

A esta MT se le denomina **máquina de Turing universal (MTU).** Como M puede recibir como entrada el código de cualquier otra MT, puede comportarse como cualquier MT, así pues, esta MT encierra todo el poder del cómputo. Todo el poder del cómputo.

Una forma de abordar el problema de construir esta MTU es construir una MT con tres cintas como sigue:

- En la cinta 1, recibirá como entrada el código de la MT M que quiere simular.
- En la cinta 2, recibe como entrada el código de la palabra que M recibiría como entrada en el cómputo que la MTU quiere simular.
- En la cinta 3, se irá codificando el estado en el que en cada paso de cómputo estaría M. Así, al inicio, en la cinta 3 estará el código del estado inicial de M.

El funcionamiento de esta MTU es sencillo:

1. Revisa su cinta 2 buscando el código de la primera letra y recoge el código del estado representado en la cinta 3.
2. Con esa información, busca en la cinta 1 la transición que puede ser aplicada. Al encontrarla, descubre el código del nuevo estado al que saltaría M, la nueva letra que habría que escribir en su cinta y el movimiento L o R que habría que hacer.
    a. La MTU modifica la cina 3 para que refleje el nuevo estado.
    b. La MTU modifica la cinta 2 para que refleje la nueva letra en la cinta, sustituyendo a la letra que había leído.
    c. La MTU mueve la cabeza de lectura/escritura de la cinta 2 para colocarla al comienzo del código de la letra de la derecha (si el movimiento era R) o a la izquierda (si era L)
    d. La MTU termina cuando la combinación de los códigos de sus cintas 2 y 3 no se corresponden con el comienzo de ninguna transición viable en su cinta 1.

Cuando la MTU termina, en la cinta 2 queda el código de la salida de la máquina M y en la cinta 3 el código del estado en el que habría terminado M. La cinta 1 no ha sufrido modificaciones (solo la usado para leer) y sigue teniendo el código de la máquina simulada.

De esta forma, la MTU puede comportarse como cualquier máquina de Turing. Es decir, es capaz de realizar cualquier cómputo imaginable y posible.

Para el estudiante que quiera saber cómo debe ser el sistema de codificación adecuado para este proceso de simulación, se le sugiere que busque información sobre "numeración de Gödel". Esta numeración, utilizada por el lógico, matemático y filósofo Kurt Gödel, asigna un código de ceros y unos único a cada letra, a cada estado y a cada máquina de Turing, de forma que dos máquinas distintas tendrán un número de Gödel distinto, y viceversa, dos números de Gödel diferentes representarán a dos máquinas distintas.

11. Bibliografía

- Kelley, Dean (1.995). Teoría de Autómatas y Lenguajes Formales. Madrid (España): Prentice Hall. Capítulos del 4 al 7.
- Glenn Brookshear, J. (1.992). Teoría de la computación. México: Addison-Wesley Iberoamericana. Capítulos del 3 al 5.


function(el, post_id,
val('members_count', cur.count_out);
replaceClass(btn, 'color4_bg color2
val(btn, cur.subscribe_lang);
setStyle('an
cur.changinGroup
lockButton(btn);
if (cur.noAuth)
Widgets.oauth(
window.gotSess
if (autorzie
setTimeout
location.h
}
if (autorzied) {
ajax.post('/widget_community.php
onDone