MÁQUINAS DE TURING

¿QUE SON Y COMO FUNCIONAN?

Una máquina de Turing consiste, básicamente, en una cinta infinita, dividida en casillas. Sobre esta cinta hay un dispositivo capaz de desplazarse a lo largo de ella a razón de una casilla cada vez. Este dispositivo cuenta con un cabezal capaz de leer un símbolo escrito en la cinta, o de borrar el existente e imprimir uno nuevo en su lugar. Por último, contiene además un registro capaz de almacenar un estado cualquiera, el cual viene definido por un símbolo. Los símbolos que definen el estado del dispositivo no tienen por qué coincidir con los símbolos que se pueden leer o escribir en la cinta. En los programas presentados en el artículo, los posibles símbolos a leer o escribir en la cinta son el 0 y el 1, y los posibles estados se representan con letras mayúsculas.

La máquina tiene un funcionamiento totalmente mecánico y secuencial. Lo que hace es leer el símbolo que hay en la casilla que tiene debajo. Después toma el símbolo del estado en que se encuentra. Con estos dos datos accede a una tabla, en la cual lee el símbolo que debe escribir en la cinta, el nuevo estado al que debe pasar y si debe desplazarse a la casilla izquierda o derecha.

Para comprender mejor, vamos a ver un simple ejemplo: sea la máquina de Turing capaz de leer o escribir los símbolos 0 y 1 en la cinta (en la definición original de Turing, el número de símbolos a usar podía ser cualquiera, con la única condición de ser un número finito, y no tenían por qué ser números; sin embargo, en aplicaciones prácticas se suelen limitar a estos dos), y que puede tener los estados A, B y C (una máquina de Turing puede tener cualquier número de estados; la única condición es que sea un número finito). Supongamos que definimos la siguiente tabla:

estado inicial	símbolo leído	nuevo estado	nuevo símbolo	sentido de avance
A	0	B	1	DERECHA
A	1	B	0	IZQUIERDA
B	0	A	1	DERECHA
B	1	C	0	DERECHA
C	0	A	0	IZQUIERDA
C	1	C	0	DERECHA

La cual vamos a simplificar de la siguiente manera:

```
            0           1

A        B,1,>       B,0,<

B        A,1,>       C,0,>

C        A,0,<       C,0,>
```

Hemos puesto los posibles estados en columna, y los posibles símbolos en fila, y hemos expresado el nuevo estado, símbolo y sentido todo junto. El sentido lo expresamos con la dirección en la que apunta el símbolo < o >.

Vamos a poner nuestra máquina sobre esta cinta:

```
       cabezal
          v
 ... 0 0 0 0 0 1 0 0 0 0 ...
```

Indicaremos el estado actual de la máquina encima del cabezal. Veamos los sucesivos pasos de esta máquina si partimos del estado A:

```
1)        A                 El estado es A y leemos un cero;
          v                 luego debemos cambiar al estado B,
... 0 0 0 0 0 1 0 0 0 0 ...  escribir un 1 y movernos a la derecha

2)          B               El estado es B y leemos un cero;
            v               luego debemos cambiar al estado A,
... 0 0 0 1 0 1 0 0 0 0 ...  escribir un 1 y movernos a la derecha

3)            A             El estado es A y leemos un uno;
              v             luego debemos cambiar al estado B,
... 0 0 0 1 1 1 0 0 0 0 ...  escribir un 0 y movernos a la izquierda

4)          B               El estado es B y leemos un uno;
            v               luego debemos cambiar al estado C,
... 0 0 0 1 1 0 0 0 0 0 ...  escribir un 0 y movernos a la izquierda

5)        C                 El estado es C y leemos un uno;
          v                 luego debemos cambiar al estado C,
... 0 0 0 1 0 0 0 0 0 0 ...  escribir un 0 y movernos a la derecha
```

```
6)          C                   El estado es C y leemos un cero;
            v                   luego debemos cambiar al estado A,
... 0 0 0 0 0 0 0 0 0 0 ...     escribir un 0 y movernos a la izquierda

7)        A                     El estado es A y leemos un cero;
          v                     luego debemos cambiar al estado B,
... 0 0 0 0 0 0 0 0 0 0 ...     escribir un 1 y movernos a la derecha
```

La ejecución de esta máquina seguiría indefinidamente, rellenando la cinta con unos y ceros de una manera más o menos aleatoria. Realmente, una máquina de Turing útil debería poder detenerse; esto es, tener un estado en el que se detiene. Dicho estado se alcanzaría igual que cualquier otro estado. Esto es, supongamos que el estado D es el de paro; lo único que debemos hacer es que, cuando la máquina haya terminado el cálculo, pase a estado D; de este modo se detiene y permite examinar la cinta para buscar el resultado.

Vemos que esta máquina no hace gran cosa. Sin embargo, una máquina de Turing puede hacer cosas útiles, tales como sumar dos números, multiplicarlos, copiarlos, etc. Disponiendo de una máquina con el suficiente número de estados, podríamos hacer con ella cualquier operación que un ordenador normal pudiese realizar.

Las máquinas de Turing plantean una deducción bastante curiosa: dado que en ellas se puede realizar cualquier trabajo computable, es posible programarlas para que simulen el comportamiento de un potente ordenador. Y como una máquina de Turing puede ser codificada en CUALQUIER ordenador, por pequeño que sea, sería posible (si disponemos de memoria suficiente, claro) emular en nuestro ordenador de casa una máquina de Turing que simule un superordenador. Esto significa que todos los ordenadores pueden realizar exactamente el mismo tipo de tareas, y que los cálculos que pueda realizar el más grande los puede llevar a cabo también el más pequeño. La única diferencia sería, obviamente, la velocidad.

SUMA DE DOS NUMEROS

Una de las tareas más simples que puede llevar a cabo una máquina de Turing es la suma de dos números. Para ello debemos definir primero una convención para representar dichos números en la cinta. En principio, podríamos pensar que, al usar los símbolos 0 y 1, podemos representar los números directamente en binario; sin embargo, la cantidad de operaciones necesarias para trabajar con ellos hace que el número de estados de la máquina sumadora crezca sorprendentemente, debido a que es un sistema de numeración posicional.

Para simplificar el proceso, vamos a representar cada número por una cadena con tantos unos como indique dicho número; así, para representar el tres, escribiríamos tres unos seguidos; para un cinco, cinco unos.

Veamos cómo podemos sumarlos. Si tenemos dos números en una cinta, separados uno de otro por un cero, la forma más fácil de sumarlos sería convertir uno de los unos de los extremos en cero, y cambiar el cero de separación por un uno. De este modo tendríamos una cadena formada por tantos unos como indica la suma de los dos números originales. Veámoslo con un ejemplo:

sumemos tres y cinco:

cinta inicial:

0000111011111000

ponemos un cero en el lugar del uno de la izquierda:

0000011011111000

ponemos un uno en el lugar del cero de separación:

0000011111111000

el resultado: ocho unos, el número ocho.

Para implementar esto, supondremos que la máquina se encuentra a la izquierda. Empezaremos en el estado A. El estado inicial no tiene por qué ser necesariamente el primero de la lista de estados, pero es costumbre que sea así. De todos modos, si no es el primero, una simple reordenación del programa resuelve el problema.

Dado que el cabezal no tiene por qué estar inmediatamente antes del primer número, sino que puede estar más alejada, debemos asegurarnos de que llega hasta él sin alterar nada. Lo que debemos hacer es que, mientras lea un cero y esté en estado A, escriba un cero (así no altera nada), vuelva a estado A, y se mueva una casilla a la derecha.

En cuanto llegue al primer número, se encontrará un uno que es preciso convertir a cero antes de seguir. Por tanto, si se encuentra en estado A y encuentra un uno, debe escribir un cero y moverse a la derecha para acercarse al cero de separación. Pero hay que tener en cuenta que, si permaneciese en estado A, como después del primer uno vienen más, los pondría todos a cero; por tanto, debemos añadir un nuevo estado: el B. En cuanto encuentre un uno en estado A, deberá pasar a este nuevo estado B.

Una vez que ha puesto ese uno a cero, debe desplazarse hasta llegar al cero de separación, pero sin alterar nada; luego, si estando en estado B encuentra un uno, debe poner un uno, moverse a la derecha, y pasar de nuevo a estado B. Pero en cuanto detecte un cero, significa que ha llegado a la separación, luego si estando en B encuentra un cero, debe ponerlo a uno.

Ahora que la suma está terminada, la máquina debe pararse, pasando al estado correspondiente. Le denominaremos @ (arroba), para distinguirlo de otros posibles estados. En cuanto lo alcanza, la máquina ya no leerá ningún símbolo más ni se desplazará.

Veamos el programa de esta máquina de sumar:

	0	1
A	0,A,>	0,B,>
B	1,@	1,B,>

COPIA DE UNA CADENA DE UNOS A CONTINUACION DE ELLA MISMA

Puede ser útil para algunas aplicaciones obtener una copia de un número a continuación de él mismo, bien sea como parte del proceso, bien porque éste lo modifica y queremos mantener el original inalterado.

Una posibilidad para copiar una cadena de N unos es usar una máquina de N estados; sin embargo, esto hace aumentar su número de manera poco menos que incontrolable; por eso nos conviene hallar un sistema que no consuma tantos estados, y, sobre todo, que funcione para cualquier cadena, no solo para las de longitud N.

Una de las maneras más cómodas es usar un cero como contador. Esto es: colocamos un cero en lugar del primer uno de la cadena, nos desplazamos hasta el final de ella, colocamos un uno, y volvemos hasta donde esté el cero. Lo cambiamos a un uno, pasamos a la siguiente casilla, ponemos un cero y repetimos el proceso. De este modo, el cero se irá corriendo a lo largo de la cadena original, y cada vez que se desplace una posición, aparecerá un nuevo uno al final de la cadena copiada. La máquina se detendrá en cuanto intente poner un cero en el lugar que ocupe otro cero en vez de un uno, pues significará que ha llegado al cero de separación de la cadena original y la cadena copia.

Veamos un programa que cumple lo expuesto, comentado:

```
                  0                                    1
A 0,A,> Nos movemos hacia la derecha    0,B,> Ponemos un cero en el
        hasta el principio de la cadena.      primer uno de la cadena.

B 0,C,> Vamos a la derecha, a través de 1,B,> Vamos desde el marcador
        la separación entre cadenas.          hasta la separación.

C 1,D,< Ponemos un uno en el extremo    1,C,> Vamos hacia la derecha
        de la cadena copia.                   de la cadena copia.

D 0,E,< Vamos hacia la izquierda desde  1,D,< Vamos hacia la izquierda
        la separación.                        por la copia.

E 1,H,< El marcador está a la izquierda 1,F,< Vamos hacia la izquierda
        de la separación. Fin de la copia.    desde la separación.
```

```
F 1,G,> Iniciamos el cambio de lugar     1,F,< Seguimos hasta encontrar
        del marcador.                          el marcador.

H ----- Nunca se da esta condición. No      0,B,> Terminamos el desplaza-
        programar o poner un bloque                miento del marcador y
        ficticio.                                  recomenzamos.

I  0,@  La cadena ya está copiada.         1,I,> Nos movemos hasta el
        la máquina debe detenerse.                 inicio de la cadena original.
```

PRODUCTO DE DOS NUMEROS

Si analizamos el producto de dos números, vemos que se trata simplemente de sumar el primero a sí mismo tantas veces como indica el segundo. Y como trabajando con nuestro particular sistema de numeración, la suma no es más que unir dos cadenas, vemos que esta es una tarea perfecta para nuestra rutina de copiado.

La manera de hacerlo es muy simple: debido a una casualidad totalmente intencionada :-) , si hay dos números seguidos (con el cero de separación entre ellos, claro) y mandamos a nuestra rutina que copie el primero (el de más a la izquierda, en nuestro caso), lo hará a continuación del segundo, pero sin separarlo; en otras palabras, habrá sumado al segundo una copia del primero. De este modo, si activamos la rutina de copia sobre el segundo número a multiplicar, tantas veces como indique el primero, lo que haremos será sumar el segundo número a sí mismo tantas veces como indique el primero. Los habremos multiplicado.

Para que la rutina de copia se active tantas veces como indica el primer número, haremos un contador con un cero, de la misma manera que hicimos en la propia rutina de copia. Así de fácil.

Veamos un programa de máquina de Turing que lo hace. Los estados de la A a la H son los mismos que los de la rutina de copia, para ahorrar espacio en el artículo. Esto significa que la máquina no se debe poner en marcha en el estado A, sino en el estado J.

```
     0         1                  0         1
I  0,L,<     1,I,<          M  1,N,>     1,M,<
J  0,J,>     0,K,>          N  -----     0,K,>
K  0,A,>     1,K,>          O   0,@      1,O,<
L  1,O,<     1,M,<
```

El bloque ----- significa que esa condición nunca puede darse, por lo que se debe dejar sin rellenar o bien poner un bloque ficticio.

LA RUTINA BORRACINTAS

Un programa que, en principio, no pasa de ser una mera curiosidad, es el programa borracintas. Lo único que hace es colocar la cinta de una máquina de Turing a cero, eliminando todos los posibles unos que pudiese contener. Dado que el número de unos existentes no lo podemos conocer a priori, y dado que la cinta es infinita, ésta máquina, para que realmente sea eficiente, no podrá detenerse nunca. De todos modos, el número de unos no puede ser infinito, pues entonces nunca conseguiríamos borrar totalmente la cinta (por mucho tiempo que tuviésemos la máquina en funcionamiento, siempre quedarían unos sin borrar). Veamos como diseñarla.

Una posible opción sería hacer una máquina de un solo estado que, encuentre lo que encuentre, escriba un cero y pase a la casilla siguiente. Su listado sería:

```
     0         1
A  0,A,>     0,A,>
```

Vemos, sin embargo, que esta máquina no es totalmente efectiva, pues sólo elimina los unos que se hallen a la derecha de la posición de origen. Podríamos diseñar una máquina borracintas que trabaje hacia la izquierda, y usar primero una y luego otra; pero como la cinta es infinita, nunca sabríamos cuando podemos parar, pues pueden quedar aún unos por eliminar. Por eso debemos diseñar otra máquina más versátil que ésta.

La segunda versión de la máquina borracintas usa dos estados. Se trata de una unión de las dos máquinas borracintas anteriores. Lo que hace es desplazarse en un sentido hasta que encuentra un uno. En cuanto llega a él lo cambia por un cero y se desplaza en el sentido contrario hasta encontrar otro uno, repitiendo el proceso. Su listado sería:

	0	1
A	0,A,>	0,B,<

	0	1
B	0,B,<	0,A,>

Si bien esta máquina es notablemente superior, vemos que puede darse el caso de que haya más unos hacia el lado derecho de la posición inicial que hacia el lado izquierdo (no olvidemos que el número de unos es finito). Si se diese esta situación, en cuanto se acabasen los unos en uno de los lados, la máquina se escaparía por él, dejando los del otro lado sin borrar. Por fortuna, en esta vida todo tiene solución, y este problema no iba a ser menos.

Para conseguir eliminar todos los unos, podemos hacer que, cada vez que la máquina encuentre uno, lo ponga a cero, coloque un uno en la siguiente posición, e invierta su sentido. De este modo, los nuevos unos se comportarán como barreras. Cada vez que la máquina llega a una, la desplaza una posición e invierte su sentido. De este modo, la máquina siempre recorrerá por igual los dos sentidos.

Una precaución consiste en colocar al principio dos unos seguidos, pues puede ocurrir que en uno de los dos sentidos no haya unos, y la máquina se pierda. De este modo, al colocar esos dos unos, inicializamos las barreras. A cada vuelta, las barreras se irán ensanchando más. Cuando una barrera se pone encima de un uno, lo anula, de modo que al siguiente paso del cabezal quedará eliminado.

El programa puede ser el siguiente:

	0	1
A	1,B,>	1,B,>
B	1,C,<	1,C,<
C	0,C,<	0,D,<

	0	1
D	1,E,>	1,E,>
E	0,E,>	0,F,>
F	1,C,<	1,C,<

Los estados A y B sirven para inicializar las dos barreras. El código del programa empieza en C, desplazándose hacia la izquierda hasta encontrar un uno, momento en que lo cambia por un cero y pasa al estado D, que pone un uno una posición más hacia la izquierda e inicia el desplazamiento hacia la derecha, pasando al estado E. En el, el cabezal se desplaza hacia la derecha hasta encontrar un uno, cambiándolo por un cero y pasando al estado F en la siguiente casilla. En este estado, se pone un uno y se comienza el desplazamiento hacia la izquierda, volviendo al estado C y cerrando el bucle.