# PROBLEMAS ALGORITMICOS EN LAS MATEMATICAS, II.

por

Ewald Burger

(Universidad de Colonia).

## 2. - LAS MAQUINAS DE TURING

Por lo tanto, en lugar de considerar algoritmos para determinar si una relación entre números naturales tiene lugar, podemos considerar algoritmos para calcular efectivamente los valores de una función aritmética. Es, pues, necesario precisar el concepto de función efectivamente calculable. Es esto lo que efectúa TURING : da un concepto preciso de máquina y define una función aritmética f como efectivamente calculable si existe una tal máquina para calcular el valor de f para cada argumento.

Vamos a considerar en detalle las máquinas de TURING. Es cierto que este concepto de máquina parecerá muy particular, pero muchos ensayos hechos por diferentes autores para definir máquinas efectivas más generales que las de TURING, siempre condujeron a conceptos de máquinas equivalentes a las de TURING. Además, el concepto de función efectivamente calculable por una máquina de Turing es equivalente a muchos otros conceptos exactos de función efectivamente calculable. Por lo tanto, la particularidad y la especialidad de las máquinas de TURING son aparentes tan sólo.

Pues bien, una máquina de TURING posee una cinta (en principio, de longitud infinita) para escribir sobre ella, la cual está dividida en una serie de cuadros, como se ve en la figura 3:

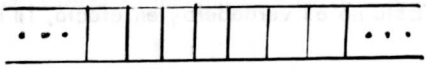


*Figura 3*

La máquina puede escribir una raya en un cuadro, o inversamente extinguir una raya anteriormente escrita en un cuadro. Está formada ella por el acoplamiento de un número finito de máquinas elementales, las cuales pueden ejecutar las actividades que a continuación les asignamos :

la **máquina de marcar** $M\rightarrow$ escribe una raya en el cuadro considerado en la cinta, en caso de que todavía no se encuentre una tal raya en el cuadro ;

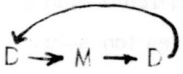
la **máquina de extinción** $E\rightarrow$ extingue la raya en el cuadro considerado de la cinta, en caso de que se encuentre una tal raya en el cuadro considerado ; en este caso decimos que la máquina marca un 0 ;

la **máquina de corriente a izquierda** $I\rightarrow$ pasa del cuadro considerado al cuadro limítrofe a la izquierda ; y

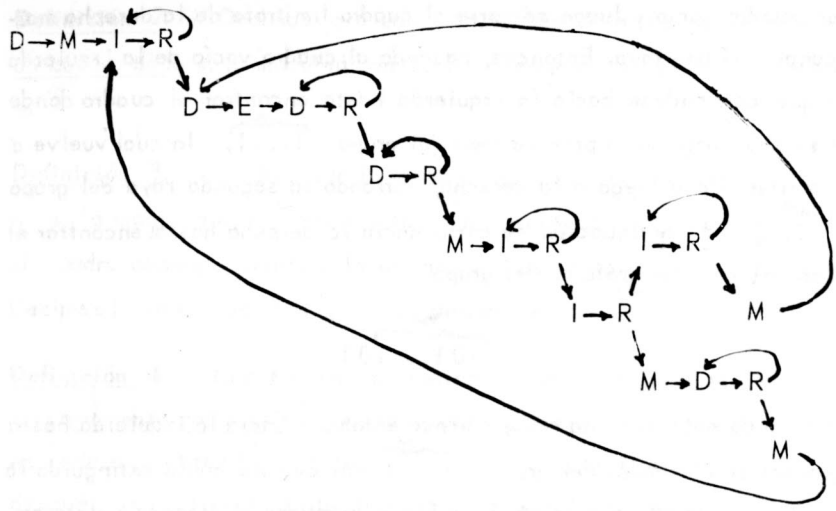
la **máquina del corriente a derecha** $D\rightarrow$ pasa del cuadro considerado al cuadro limítrofe a la derecha. Por fín,

la máquina $R$ (**máquina de decisión o reducción**) determina que el procedimiento continue de una cierta manera si se encuentra una raya en el cuadro considerado (flecha superior), o que continue de una otra cierta manera si no se encuentra una raya en el cuadro considerado ( flecha inferior ).

Vamos a aclarar el funcionamiento de las máquinas de TURING con dos ejemplos : en primer lugar, la máquina acoplada de la manera siguiente :




Esta máquina escribe sobre la cinta la serie 0101010..... Podría pensarse que una máquina de TURING puede solamente escribir series periódicas de signos. Esto no es verdadero ; en efecto, la máquina




escribe la serie 0101101110.... Veámoslo : empezando en un cuadro vacío, la máquina escribe en primer lugar 01 y pasa en seguida al cuadro de la izquierda. Suponiendo ahora aue la máquina ha escrito la serie

$$010110\ldots0\overbrace{1\ldots1}^{n} \qquad (n > 1)$$

vamos a mostrar aue después de un número finito de movimientos la máquina ha escrito

$$010110\ldots0\overbrace{1\ldots1}^{n}0\overbrace{1\ldots1}^{n+1}$$

En efecto : la máquina se corre sucesivamente a la izquierda hasta encontrarse en el postrer cuadro vacío de la serie

$$010110\ldots0\overbrace{1\ldots1}^{n}$$

Entonces la máquina se correa a la primera raya del grupo $\overbrace{1\ldots1}^{n}$ y la extingue. Luego la máquina se corre a la derecha hasta encontrar

un cuadro vacío y luego se corre al cuadro limítrofe de la derecha marcando allí una raya. Entonces, pasando al cuadro vacío de la izquierda sigue corriéndose hacia la izquierda hasta encontrar el cuadro donde habíamos borrado la primera raya del grupo $\overbrace{1\ldots1}^{n}$, la cual vuelve a escribir. Pasa luego a la derecha, borrando la segunda raya del grupo $\overbrace{1\ldots1}^{n}$; a continuación se corre hacia la derecha hasta encontrar el cuadro vacío limítrofe con el grupo

$$\overbrace{101\ldots101}^{n}$$

marcando entonces una raya; córrese entonces hacia la izquierda hasta encontrar el cuadro del grupo $\overbrace{1\ldots1}^{n}$ al cual le había extinguido la raya, volviéndola a escribir, y vuelve a repetirse el proceso entero empezando por extinguir la tercera raya del grupo $\overbrace{1\ldots1}^{n}$. Después de n ciclos de este género, se llega a tener en la cinta lo siguiente :

$$010\ldots01\overbrace{1\ldots1}^{n}0\overbrace{1\ldots1}^{n}$$

corriendo enseguida a la derecha para marcar una $(n+1)$-ésima raya. Considerando este cuadro final, la máquina pasa nuevamente a la izquierda sucesivamente para empezar entonces el proceso que nos dará el grupo de $n+2$ rayas. Y así sucesivamente.

Damos a continuación algunas definiciones que completan la explicación de las máquinas de TURING.

**Definición 1.** - Decimos que una máquina de TURING está colocada sobre el argumento $(n_1, \ldots, n_v)$ si en la cinta está escrita la serie

$$0\overbrace{1\ldots1}^{n_1}0\overbrace{1\ldots1}^{n_2}0\ldots0\overbrace{1\ldots1}^{n_v}$$

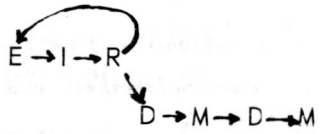
y la máquina considera el cuadro en el cual se encuentra la última raya del grupo $\overbrace{1\ldots1}^{n_v}$.

**Definición 2.** - Decimos que una máquina de TURING se detiene en un cuadro dado si la máquina considera este cuadro y cesa su movimiento.

**Definición 3.** - Se dice que la máquina ha calculado el número entero n si la máquina está colocada sobre el argumento n y se detiene en el cuadro correspondiente a la última raya de la derecha: $0\overbrace{1\ldots1}^{n}0$. Decimos también que la máquina se detiene en $0\overbrace{1\ldots1}^{n}0$

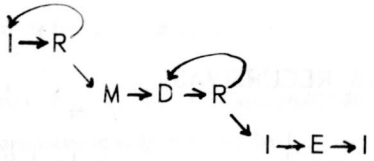
**Definición 4.** - Una función aritmética f de v argumentos se dice efectivamente calculable si existe una máquina de TURING que puesta en cada argumento $(n_1, \ldots, n_v)$ tiene calculado el número $f(n_1, \ldots, n_v)$ después de un número finito de pasos o movimientos de la máquina.

*Ejemplo 1.* - La máquina




colocada sobre el argumento n, siempre se detiene en 0110 y por tanto calcula la función $f(n) = 2$

*Ejemplo 2.* - La máquina




colocada en el argumento $(m, n)$ calcula la función $f(m, n) = m + n$; en efecto, si en la cinta aparece

$$0\overbrace{1\ldots1}^{m}0\overbrace{1\ldots1}^{n}0$$

y la máquina considera el cuadro de la postrer raya, se traslada sucesi-

vamente a la izquierda hasta colocarse en el cero comprendido entre los grupos de m y n rayas; allí hace una marca, trasladándose por la derecha al cuadro vacío limítrofe con la última raya del grupo $\overbrace{1\ldots1}^{n}$; obteniéndose entonces en la cinta la configuración

$$0\overbrace{1\ldots1\ldots1}^{m+n+1}0$$

con la máquina puesta en el argumento $m+n+1$; borra enseguida la última raya de esta configuración y retrocede entonces para ponerse en el argumento $m+n$. Aquí la máquina se detiene.

Naturalmente, estos son ejemplos triviales de funciones efectivamente calculables, pero debemos contentarnos aquí con ellos, remitiéndonos, para un desarrollo ulterior de la teoría de las máquinas de TURING y de los algorítmos en general, a los siguientes libros :

DAVIS, *Computability and unsolvability, New York 1958*

HERMES, *Anfzählbarkeit, Entscheidbarkeit und Berechenbarkeit, Berlín, 1961.*

KLEENE, *Introduction to Metamathematics, Ansterdam, 1952*

El libro de KLEENE, a pesar de ser de temática más general, consagra un capítulo a las máquinas de TURING y sobre todo a sus aplicaciones a la lógica y a las investigaciones de los fundamentos de la matemática.

## 3. - FUNCIONES RECURSIVAS.

Ya hemos indicado que además de la definición por medio de máquinas de TURING, existen algunas otras posibilidades de definir el concepto ce función efectivamente calculable. La definición por medio de máquinas de TURING tiene la gran ventaja de que es muy fácil de concebir desde un punto de vista intuitivo. Pero tomando desde un principio el punto de vista matemático sería tal vez natural proceder de o-

tro modo, a saber : consignar en primer lugar algunas funciones sencillas, claramente calculables, e indicar en segundo lugar algunas operaciones que transformen funciones calculables en funciones calculables, y por fín definir las funciones calculables como aquellas funciones que se pueden obtener por medio de un número finito de las operaciones indicadas tomando como punto de partida las funciones sencillas antes consignadas. Es éste el método por el cual GODEL y KLEENE han introducido el concepto de función calculable, o como se dice de función recursiva; vamos a considerar esta definición de GODEL-KLEENE en detalle y a indicar los puntos principales de la demostración de la equivalencia entre funciones recursivas y funciones calculables según TURING.

**Definición 5.** - Una función aritmética se llama primitivamente recursiva si se puede obtener por un número finito de aplicaciones de las operaciones de sustitución y de definición inductiva tomando como punto de partida las funciones siguientes :

a) la función sucesor : $S(n) = n + 1$;

b) la función de identidad (proyección) : $I_v^k(n_1, \ldots, n_v) = n_k$ $(1 \leq k \leq n)$;

c)) las funciones constantes : $C(n_1, \ldots, n_v) = C$.

Aquí nos será de utilidad recordar las definiciones de las operaciones de sustitución y de definición inductiva :

**Defínción 6.** - Sean $h_1, \ldots, h_r$ r funciones de v argumentos y g una función de r argumentos. Entonces la función f de v argumentos definida por la relación

$$f(n_1, \ldots, n_v) = g(h_1(n_1, \ldots, h_v), \ldots, h_r(n_1, \ldots, n_v))$$

se dice una función obtenida por sustitución (Composición de funciones ).

**Definición 7.** - Sea g una función de v argumentos y h una función

de $v+2$ argumentos. Entonces la función $f$ de $v+1$ argumentos definida por

$$f(n_1,\ldots,n_v,0) = g(n_1,\ldots,n_v)$$

$$f(n_1,\ldots,n_v,n+1) = h(n_1,\ldots,n_v,n,f(n_1,\ldots,n_v,n))$$

se dice la función obtenida por <u>definición inductiva</u> mediante las funcionex $g$ y $h$.

Damos ahora algunos ejemplos de funciones definidas inductivamente : tomando $v = 1$, $g(n) = n$, $h(m,n,t) = S(t)$, la función resultante es $f(m,n) = m + n$. Así mismo, si $v = 1$, $g(n) = 0$ y $h(n,m,t) = t + n$, la función resultante es la función

$$f(n,0) = 0$$

$$f(n,m+1) = f(n,m) + n$$

es decir, $f(m,n) = nm$. También podemos demostrar que a partir de la función producto podemos definir inductivamente las funciones $(m,n) \longrightarrow m^n$ y $n \longrightarrow n!$. Muchas otras funciones como

$$|n-m| = \begin{cases} n-m & \text{si} \quad n \geq m \\ m-n & \text{si} \quad n < m \end{cases} \qquad sg(n) = \begin{cases} 0 & \text{si} \quad n = 0 \\ 1 & \text{si} \quad n > 0 \end{cases}$$

son también primitivamente recursivas, e intuitivamente efectivamente calculables. Vemos, pues, que la clase de las funciones primitivamente recursivas es bastante amplia; sin embargo, no todas las funciones efectivamente calculables son primitivamente recursivas (y queremos mostrar que toda función calculable es recursiva en el sentido de GODEL-KLEENE). ACKERMMAN fué el primero en demostrar la existencia de una función de carácte efectivamente calculable pero no primitivamente recursiva. La función de ACKERMMAN (un poco simplificada) se define por las ecuaciones siguientes :

$$f(0,n) = n + 1$$

$$f(m+1,0) = f(m,1)$$

$$f(m+1,\ n+1) = f(m,f(m+1,n))$$

Debemos en primer lugar demostrar que este sistema de ecuaciones define en realidad una función f, y que ésta es efectivamente calculable. La demostración puede efectuarse por medio de inducción completa con respecto a m. Para indicar lo esencial bastará aquí presentar, por ejemplo, el cálculo del valor $f(2,1)$:

$$\begin{aligned} f(2,1) &= f(1,f(2,0)) = f(1,f(1,1)) = f(1,f(0,f(1,0))) \\ &= f(1,f(1,0)+1) = f(1,f(0,1)+1) \\ &= f(1,3) = f(0,f(1,2)) = f(0,f(0,f(1,1))) \\ &= f(0,f(0,f(0,f(1,0)))) = f(0,f(0,f(0,f(0,1)))) \\ &= f(0,f(0,f(0,2))) = f(0,f(0,3)) \\ &= f(0,4) = 5 \end{aligned}$$

En segundo lugar debemos demostrar que la función de ACKERMMAN no es primitivamente recursiva. La demostración, demasiado larga para presentarla completa aquí, se apoya en el lema siguiente :

**Lema.** - Si f es la función de ACKERMMAN, para cada función aritmética $g(n_1,\ldots,n_v)$ existe un número natural c tal que

$$g(n_1,\ldots,n_v) < f(c,n_1+n_2+\ldots+n_v) \quad .$$

Ahora bien, si f fuese primitivamente recursiva, la función $g(n) = f(n,n) = f(I^1_1(n), I^1_1(n))$ sería también primitivamente recursiva, y por el lema existiría un número natural c tal que $g(n) = f(n,n) < f(c,n)$ para todo n. En particular, tendríamos $f(c,c) < f(c,c)$, lo cual es imposible. La demostración del lema puede hallarse en el citado libro de HERMES.

La existencia de la función de ACKERMMAN demuestra que la clase de las funciones primitivamente recursivas todavía no es lo bastante amplia para contener todas las funciones efectivamente calculables. Para que ello ocurra debemos admitir otra operación, además de la sustitución y la de definición inductiva, y que transforma también funciones efectivamente calculables en funciones efectivamente calculables. Esta operación se obtiene a partir del llamado $\mu$-operador, el cual se define de la manera siguiente :

**Definicion 8.** - Sea $g$ una función aritmética de $v+1$ argumentos que tenga la propiedad : para todo $n_1, \ldots, n_v$ existe $n$ tal que $g(n_1, \ldots, n_v, n) = 0$. La función $f$ definida por

$$f(n_1, \ldots, n_v) = \min \{ n \mid g(n_1, \ldots, n_v, n) = 0 \}$$

$$= \mu g : (n_1, \ldots, n_v)$$

se dice que función obtenida de $g$ por el $\mu$-operador.

Es claro que si $g$ es efectivamente calculable, $f$ también lo es; en efecto, para calcular el valor $f(n_1, \ldots, n_v)$ sólo es necesario calcular los valores $g(n_1, \ldots, n_v, 0)$, $g(n_1, \ldots, n_v, 1), \ldots$ hasta encontrar el primer $k$ tal que $g(n_1, \ldots, n_v, k) = 0$ (el cual existe por la hipótesis sobre $g$), con lo cual $k = f(n_1, \ldots, n_v)$. Naturalmente no es posible, en general, dar con anticipación una cota superior al número de pasos necesarios para la determinación de $f(n_1, \ldots, n_v)$. Estas consideraciones hacen plausible el hecho de que la aplicación del $\mu$-operador a funciones primitivamente recursivas, no produce en general una función primitivamente recursiva, sino una función efectivamente calculable más general. Es así como se puede demostrar, por ejemplo, que la función de ACKERMMAN se obtiene por medio de la aplicación del $\mu$-operador a una función primitivamente recursiva.

Ponemos en seguida la siguiente defnición :

**Definición 9.** - Una función aritmética se llama recursiva si se puede

obtener por un número finito de aplicaciones de las operaciones de sustitución, definición inductiva y aplicación del $\mu$-operador, tomando como funciones base las funciones S, $I^k_\nu$, C.

Un teorema fundamental de la teoría de algoritmos es entonces el siguiente:

*TEOREMA.* - Una función aritmética es recursiva si si es efectivamente calculable por una máquina de TURING.

**Demostración.** - La demostración de este teorema es bastante larga y aquí sólo daremos algunas indicaciones de los puntos decisivos y esenciales de la demostración. Para demostrar que cada función recursiva puede calcularse por una máquina de TURING, se procede en principio como hemos indicado en el caso de la función $f(n,m) = m+n$; es decir, se construye una máquina de TURING que sea capaz de calcular efectivamente la función en cuestión. Esta construcción no es difícil para las funciones S, $I^k_\nu$, C, y la podemos dejar como ejercicio al lector.

En seguida se construyen máquinas de TURING capaces de efectuar las operaciones de sustitución, de definición inductiva y de $\mu$-operación. Estas construcciones son difíciles en principio ya que naturalmente exigen un esquema de acoplamiento de las máquinas elementales bastante extenso y complicado; en todo caso, la construcción explícita de dichas máquinas puede hallarse en los citados libros de DAVIS y HERMES. Una vez acabada la construcción de estas máquinas es fácil imaginarse ya cómo tienen que acoplarse con las máquinas de las funciones S, $I^k_\nu$ y C, para construir una máquina de TURING que efectivamente calcule una función recursiva arbitrariamente prefijada. El acoplamiento no es otra cosa que una imitación del procedimiento por el cual se define la función recursiva prefijada mediante las funciones recursivas iniciales y las otras operaciones características. Por lo tanto, hemos prácticamente demostrado que cada función recursiva es efectivamente calculable en el sentido de TURING.

Queda por demostrar recíprocamente que toda función efectivamente calculable por una máquina de TURING es recursiva. Para este fin se efectúa una aritmetización de las máquinas de TURING por medio de una numeración de GODEL. Una numeración de GODEL, recordamos, hace corresponder a cada elemento de un conjunto enumerable un número en tal forma que la transición de un elemento al número que la corresponde, y viceversa, se efectúa de manera efectiva. No es difícil, aunque un poco extenso y prolijo, introducir una numeración de GODEL para las máquinas de TURING, y también para el conjunto enumerable de todas las series finitas compuestas de los signos 0 y 1, i. e., para todas las configuraciones finitas de signos que pueden formarse en la cinta de una máquina de TURING en un intervalo finito de trabajo de la máquina. Esto sentado, consideramos el predicado aritmético de $v + 2$ argumentos

$$T_v(k, n_1, \ldots, n_v, \varrho)$$

el cual tiene lugar si $k$ es el número de GODEL de una máquina de TURING que puesta sobre el argumento $(n_1, \ldots, n_v)$ se detiene después de un proceso (de un número finito de pasos) tal que $\varrho$ es el número de GODEL de la serie (finita) completa de configuraciones escritas por la máquina durante todo el proceso. El punto decisivo es el de que por medio de una investigación detallada del funcionamiento de una máquina de TURING se puede demostrar que el predicado $T_v$ es primitivamente recursivo; de manera precisa, su función característica

$$\chi_{T_v}(k, n_1, \ldots, n_v, \varrho) = \begin{cases} 1 & \text{si tiene lugar } T_v \\ 0 & \text{si no tiene lugar } T_v \end{cases}$$

es primitivamente recursiva. Además se demuestra muy fácilmente la existencia de una función $U'$ primitivamente recursiva de un argumen-

to con la propiedad siguiente : si $\varrho$ es el número de GODEL de una serie finita de configuraciones escrita en la cinta de una máquina de TURING, entonces $U'(\varrho)$ es el número de GODEL de la postrera configuración de esta serie. Además es fácil ver que existe una función primitivamente recursiva U'' de un argumento con la propiedad siguiente : si n es el número de GODEL de la configuración $01\overbrace{\ldots}^{m}10$, entonces $U''(n) = m$. Pues bien, sea ahora f una función de v argumentos calculables por una máquina de TURING y sea k el número de GODEL de esta máquina ; entonces, evidentemente, para todo $(n_1, \ldots, n_v)$ existe $\varrho$ tal que

$$T_v(k, n_1, \ldots, n_v, \varrho)$$

y

$$f(n_1, \ldots, n_v) = U''(U'(\mu\varrho : T(k, n_1, \ldots, n_v, \varrho)))$$

$$= U''(U'(\mu\varrho : \chi_{T_v}(k, n_1, \ldots, n_v, \varrho) - 1 = 0))$$

y f es, pues, en realidad una función recursiva, tal como queríamos demostrar.

*(Recibido en octubre de 1962)*