MÁQUINAS DE TURING (1)

ANDRÉS SICARD RAMÍREZ

1. SÍNTESIS

Los conceptos de ALGORITMO y COMPUTADOR UNIVERSAL son definidos desde la lente de la informática teórica y en particular, a partir de las definiciones realizadas por Alan Mathison Turing, en su célebre artículo "*On computable numbers, with an application to the Entscheidungsproblem*" (Sobre números computables, con una aplicación al problema de la decisión) [Tur36]. Para ello, nos situamos en el contexto histórico adecuado, ilustramos los conceptos tanto desde sus definiciones formales como informales y finalmente, describimos la solución obtenida por Turing al PROBLEMA DE LA DECISIÓN.

2. INTRODUCCIÓN, MARCO HISTÓRICO Y MARCO CONCEPTUAL

En la actualidad contamos con una noción muy precisa del concepto general de ALGORITMO (2) (palabra que procede del nombre del matemático persa del siglo IX, Abu Ja'far Mohammed ibd Mûsa al-Khowâriz [Pen95]), pero la situación no era la misma a principios de este siglo.

Como consecuencia del descubrimiento (a finales del siglo pasado y a comienzos de éste) de las paradojas o antinomias en la teoría de conjuntos y la situación que esto generó (situación muy importante dado el papel de teoría base que desempeña la teoría de conjuntos en la matemática), se abordó un problema más amplio que comprendía la fundamentación de la matemática y la lógica. Tres escuelas principales surgieron con directrices muy marcadas acerca de la fundamentación de la matemática: La escuela logicista, la escuela intuicionista y la escuela formalista.

La escuela logicista (Dedekind, Frege, Peano, Whitehead, Rusell) establece como tesis que: La matemática es una rama de la lógica, es decir las nociones matemáticas han de ser definidas en términos de las nociones lógicas y los teoremas de la matemática han de ser demostrados como teoremas lógicos [Kle74]. Dentro de esta escuela mencionaremos el intento de trasladar los conceptos matemáticos a enunciados de la lógica iniciado por Gottlob Frege (1848-1925) en su trabajo "*Bergriffsschrift*" (1879) (La notación de los conceptos) [Hop84] y la monumental obra de los "*Principia Mathematica*"(1910-13) de Bertrand Russell y Alfred North Withehead.

(1) El autor agradece a los revisores de este artículo por las sugerencias de fondo y de forma realizadas. Por supuesto soy el único responsable de los posibles defectos que puedan encontrarse en él.

(2) En realidad existen varias nociones (equivalentes) de este concepto.

ANDRÉS SICARD RAMÍREZ. Ingeniero de Sistemas, Universidad EAFIT. Estudiante de la maestría en Ingeniería Informática, Universidad EAFIT. Profesor del departamento de Ciencias Básicas, Universidad EAFIT. Email: asicard@sigma.eafit.edu.co.

De la escuela intuicionista (Brouwer) únicamente mencionaremos que no acepta las leyes lógicas cuando se aplican a conjuntos infinitos, en particular no acepta la ley del tercio excluido (la matemática no intuicionista y la matemática intuicionista difieren esencialmente en su consideración del infinito).

La escuela formalista (Hilbert, Ackerman, von Newman, Bernays) encabezada por David Hilbert (1862-1943) (considerado por algunos el matemático más brillante de nuestro siglo), propuso un programa de formalización de las matemáticas que consistía en: La matemática debe ser formulada como una teoría axiomática formal, y deberá demostrarse que esta teoría es consistente, es decir libre de contradicciones [Kle74]. La formalización estricta de una teoría da por resultado lo que denominamos un sistema formal y el objeto de estudio de los "formalistas" es precisamente este sistema formal considerado como un *todo,* utilizando para ello métodos meta-matemáticos o meta-lógicos que permitan estudiar las propiedades meta-matemáticas (consistencia, decidibilidad, completitud, etc.) del sistema en cuestión.

En 1900, en el Congreso Internacional de Matemáticos, realizado en París, Francia; Hilbert propuso 23 problemas que deberían marcar el desarrollo de las matemáticas en los años siguientes. El décimo de estos problemas consistía en: ¿Es posible encontrar un procedimiento mecánico para calcular la solución de una clase particular de ecuaciones (ecuaciones diofánticas [3]). El mismo Hilbert generalizó el problema y lo presentó en 1928 en el Congreso Internacional de Matemáticos, realizado en Bolonia, Italia [Pen95]; esta generalización es conocida actualmente como el "*Entscheidungsproblem*" (PROBLEMA DE LA DECISIÓN) [4] y su enunciado es: ¿Existe algún método o procedimiento mecánico, que pueda *en principio,* decidir (resolver) *todas* las preguntas (problemas) matemáticos?. Expresado en términos más "formalistas", el PROBLEMA DE LA DECISIÓN consistía en encontrar un método *general* para determinar si una fórmula era o no verdadera en un sistema formal dado. Tal como está expresado, el PROBLEMA DE LA DECISIÓN hace referencia al *caso general* (cualquier pregunta matemática, cualquier fórmula, cualquier sistema formal), pero es posible plantearlo para un *caso en particular* (un dominio específico). Como era de esperarse, dada su condición de líder de la escuela formalista, Hilbert estaba convencido de la respuesta afirmativa al PROBLEMA DE LA DECISIÓN.

Veamos la situación en la que nos encontramos, si la tesis logicista es correcta y el PROBLEMA DE LA DECISIÓN es resuelto favorablemente, todo el conocimiento matemático quedaría reducido a un simple cálculo *mecánico,* es decir todas las cuestiones matemáticas podrían ser resueltas, no existirían ni paradojas, ni conjeturas; la matemática sería una ciencia "muerta".

El sorprendente e inesperado resultado conocido hoy en día como TEOREMA DE INCOMPLETITUD de Gödel, obtenido por el brillante lógico austríaco Kurt Gödel (1906-1978), en su artículo : "Uber formal unentscheidbare Sätze der Principa Mathematica und verwandter Systema I" (Sobre sentencias formalmente indecidibles de *Principia Mathematica* y sistemas afines), publicado en 1931 a la edad de tan sólo 25 años, marcó el fin del ambicioso programa de formalización de las matemáticas, pero es necesario decirlo, "justifica" la existencia de los matemáticos hoy en día. Para comprender la importancia del resultado del Gödel sería necesario entrar a definir ciertos conceptos de meta-matemática o meta-lógica (sistema formal, consistencia, completitud, decidibilidad, etc.), pero ésto nos alejaría del objetivo de nuestro artículo. Lo que si podemos decir, es que el resultado de Gödel estableció la insolubilidad del PROBLEMA DE LA DECISIÓN para un caso en particular (en el caso de decidir si una fórmula es verdadera o no en un sistema formal que contenga por lo menos al sistema formal de la aritmética), con lo cual estableció el carácter irresoluble del PROBLEMA DE LA DECISIÓN en su caso general.

Alan Mathison Turing (1912-1954) también demostró el carácter irresoluble del PROBLEMA DE LA DECISIÓN planteado por Hilbert, pero por un camino diferente al de Gödel. Turing vió la necesidad de contar con una noción más precisa del concepto de PROCEDIMIENTO DE DECISIÓN al cual hace referencia

---

(3) Ecuaciones algebraicas con una o más incógnitas, de coeficientes enteros y de las que interesa únicamente las soluciones enteras.

(4) Algunos autores [Hop84] mencionan que el problema de la decisión es una generalización del segundo problema planteado por Hilbert en el congreso de 1900. Este problema consistía en demostrar que los axiomas de la aritmética eran consistentes.

el PROBLEMA DE LA DECISIÓN, para lo cual concibió la noción de COMPUTABILIDAD. Se cree que la idea de que el PROCEDIMIENTO DE DECISIÓN es un procedimiento mecánico, es decir puede ser ejecutado por un ente sin inteligencia, siempre y cuando tenga acceso a las instrucciones indicadas por él (éstas sí creadas con inteligencia) llevó a Turing a concebir la idea de una máquina abstracta para describir este concepto. Esta máquina abstracta se conoce actualmente como la MÁQUINA DE TURING y corresponde a la noción del concepto de ALGORITMO. Turing replanteó el PROBLEMA DE LA DECISIÓN en términos de sus máquinas y demostró que este problema no tenia solución (para su caso general)[5].

Como mencionamos en la sección 1 este artículo desarrolla dos conceptos, el concepto de ALGORITMO y el concepto de COMPUTADOR UNIVERSAL, y hasta el momento solo hemos hablado acerca del primero de ellos. Pues bien, en el mismo artículo donde Turing expuso su máquina, describió lo que hoy conocemos como MÁQUINA UNIVERSAL DE TURING, que en pocas palabras es una máquina que "ejecuta" máquinas de Turing. Resulta que esta MÁQUINA UNIVERSAL DE TURING, es el modelo teórico de nuestros actuales computadores, es un COMPUTADOR UNIVERSAL (claro, con las características de una máquina abstracta por supuesto), pero en teoría, cualquier ALGORITMO que pueda ser ejecutado por un computador, puede ser ejecutado por una MÁQUINA UNIVERSAL DE TURING. Como era de esperarse, se han propuesto diferentes modelos de COMPUTADOR UNIVERSAL, siendo el último de ellos (de acuerdo a lo que conocemos), el computador cuántico, del cual se afirma que, desde el punto de vista de la clase de problemas que puede computar, tiene el mismo poder que una MÁQUINA UNIVERSAL DE TURING, pero puede resolver los problemas más "rápido" [Pen95] (consideramos que por ésto pierde interés, desde la perspectiva teórica de ampliar los límites actuales de lo que es EFECTIVAMENTE CALCULABLE, pero se constituye en un modelo muy prometedor para el área de la teoría de complejidad algorítmica).

Bueno, confiamos que el lector sabrá disculpar esta extensa, pero necesaria introducción. Lo invitamos a continuar por este representativo y esperamos, ilustrativo viaje, por el mundo de las máquinas de Turing.

(5) Hubiéramos deseado hacer mención a la tesis de Church: Toda función efectivamente calculable (predicado efectivamente decidible) es recursiva general [Kle74], a sus consideraciones epistemologías (sugeridas por el profesor Raúl Gómez); a las diferentes nociones de función efectivamente calculable: recursividad general (Gödel, Herbrand), λ -definibilidad (Church, Kleene), computabilidad (Turing, Post) y las diferentes nociónes de algoritmo: máquinas de Turing, función recursiva, sistemas de Thue, etc.; pero lo hemos evitado por dos razones, la primera es que no contamos con los elementos necesarios para abordar estos temas adecuadamente y en segundo lugar consideramos que éstos "merecen" un artículo independiente.

Se cree que la idea de que el procedimiento de decisión es un procedimiento mecánico, es decir puede ser ejecutado por un ente sin inteligencia, siempre y cuando tenga acceso a las instrucciones indicadas por él (éstas sí creadas con inteligencia) llevó a Turing a concebir la idea de una máquina abstracta para describir este concepto. Esta máquina abstracta se conoce actualmente como la Máquina de Turing y corresponde a la noción del concepto de algoritmo.

3. DESCRIPCIÓN INFORMAL DE LA MÁQUINA DE TURING

Como mencionamos en la sección 2 Turing construyó una máquina abstracta o si se prefiere una máquina matemática conocida en nuestros días como MÁQUINA DE TURING, con la cual describió la noción de ALGORITMO.

Un ALGORITMO recibe algunos datos de entrada, los procesa y genera unos resultados [6]. Usualmente cuando ejecutamos un ALGORITMO a mano, utilizamos el papel como mecanismo de entrada-salida del ALGORITMO. Veamos inicialmente cuál es el mecanismo de entrada-salida utilizado por las máquinas de Turing.

(6) Estrictamente hablando, no todos los algoritmos necesitan de datos de entrada y tampoco tienen porque generar (escribir) resultados (aunque siempre es posible hacer las generalizaciones y modificaciones del caso para que así fuese).

La MÁQUINA DE TURING utiliza una cinta infinita como mecanismo de entrada-salida. La cinta está dividida en celdas las cuales pueden contener sólo un símbolo de un alfabeto *finito* de símbolos indivisibles (el símbolo vacío se representa por □ y por convención hace parte de este alfabeto). Técnicamente hablando decimos que la cinta, es una cinta unidimensional bi-infinita (infinita en ambos extremos) [7].

En cada instante (discreto) de tiempo, la MÁQUINA DE TURING, "observa" una celda de la cinta y tiene la capacidad de leer-escribir (un símbolo) sobre ésta. Adicionalmente la máquina tiene la capacidad de moverse sobre la cinta, es decir, se puede desplazar una celda a la derecha, o una celda a la izquierda, o no desplazarse [8], con respecto a su posición actual [9]. Aunque hemos mencionado que es la máquina la que se desplaza sobre la cinta, no existe ningún inconveniente, desde el punto de vista conceptual, el considerar que es la cinta la que se desplaza sobre la máquina (aunque esto presenta algunos problemas teóricos de cómo desplazar una cinta bi-infinita). Nuestra convención será, que es la máquina la que se desplaza sobre la cinta.

Además del alfabeto (conjunto *finito* de símbolos indivisibles), la máquina tiene también un conjunto *finito* de estados, los cuales representan, como su nombre lo indica, el estado en el que está la máquina (en algún instante discreto de tiempo).

(7) No existe ningún inconveniente en considerar la cinta como una cinta unidimensional infinita, o una cinta bidimensional, o inclusive en su caso general, n-dimensional. Se ha demostrado, desde el punto de vista del comportamiento de la Máquina de Turing, su equivalencia para cualquiera de estas cintas.

(8) Algunos autores eliminan esta posibilidad, pero esto no representa ningún inconveniente (Turing, si la considero en su descripción original).

(9) La capacidad de movimiento de una máquina de Turing está determinada por el tipo de cinta que utiliza. Para el caso de una cinta unidimensional, la máquina se podrá desplazar a la derecha, a la izquierda o no hacerlo; para el caso de una cinta bidimensional será necesario contar además con desplazamientos hacia arriba y hacia abajo; y en general se añadirá una pareja de desplazamientos, por cada nueva dimensión de la cinta.

Definimos una situación de la máquina, como una dupla formada por un estado (perteneciente al conjunto finito de estados) y un símbolo (perteneciente al alfabeto). Para cada instante de tiempo la situación actual de la máquina está determinada por el estado actual en el cual se encuentra y por el símbolo que está en la celda, sobre la cual la máquina está situada (la acción de leer el símbolo de la celda es automática). Es necesario entonces antes de comenzar la ejecución de la máquina, definir su situación inicial (estado inicial, posición inicial sobre la cinta).

El elemento más importante de una MÁQUINA DE TURING lo constituye el conjunto *finito* de instrucciones. Éste representa el comportamiento de la máquina. Cada instrucción está compuesta por dos partes; la primera dice cuándo se debe ejecutar la instrucción, es decir en que situación (estado, símbolo), la segunda parte indica que hace la instrucción, lo cual consiste en: escribir un símbolo (en la posición actual), ejecutar un movimiento sobre la cinta y colocar la máquina en un nuevo estado. Es decir cada instrucción está compuesta por cinco elementos: estado-actual, símbolo-leer, símbolo-escribir, movimiento, estado-siguiente; donde los dos primeros indican *cuándo* y los tres restantes *qué hacer.*

Veamos cómo opera una MÁQUINA DE TURING: La máquina busca en su conjunto de instrucciones una instrucción que concuerde (la primera parte) con la situación actual (estado actual, símbolo actual) de la máquina. Si la encuentra, la ejecuta, escribiendo el símbolo, realizando el movimiento y pasando al estado indicado por la instrucción hallada. En este momento la máquina se encuentra en una nueva situación actual y repite el proceso. Si por el contrario la máquina no encuentra una instrucción que concuerde con la situación actual, la máquina se detiene y se da por finalizada su ejecución. Éste es el contexto adecuado para la siguiente pregunta: ¿Es posible que la máquina tenga más de una instrucción para una situación (actual) en particular? De acuerdo con la respuesta obtenemos dos clases diferentes de máquinas de Turing. Las MÁQUINAS DE TURING DETERMINÍSTICAS (*automatic machine*, en el original) *no* permiten definir más de una instrucción para una situación en particular; por el contrario las MÁQUINAS DE TURING NO DETERMINÍSTICAS (*choice automatic machine* en el original) *si* lo permiten.

4. DESCRIPCIÓN FORMAL DE LA MÁQUINA DE TURING

Definimos formalmente una MÁQUINA DE TURING (determinística)[10] por una quíntupla:

$$\mathbf{M} = <K, \Sigma, M, q_x, I>\ ^{(11)} \text{ donde:}$$

$K = \{q_0, q_1, q_2, ..., q_n\}$: Conjunto finito de estados de la máquina ($K \neq \varnothing$).

$\Sigma = \{s_0, s_1, s_2, ..., s_m\}$: Alfabeto. Conjunto finito de símbolos de entrada-salida. Adoptamos por convención que $s_0 = \square$ (símbolo vacío). ($\Sigma - \{s_0\} \neq \varnothing$).

$M = \{L, R, N\}$: Conjunto de movimientos (L: izquierda, R: derecha, N: no movimiento) [12].

q_x : Estado inicial ($q_x \in K$)

I: Es una función definida de un subconjunto [13] de $K \times \Sigma$ en $\Sigma \times M \times K$.
I también puede ser definida como un conjunto *finito* [14] $I = \{i_0, i_1, i_2, ..., i_p\}$ donde cada i_j es una quíntupla de la forma: $q_m\ s_m\ s_n\ m\ q_n$, donde $q_m, q_n \in \Sigma$; $s_m, s_n \in K$; $m \in M$ (utilizaremos esta forma para definir el conjunto I).

Con respecto a la información inicial que contiene la cinta y la posición inicial de la máquina sobre ésta, se suministran "informalmente" antes de comenzar la ejecución de la máquina.

Si la máquina se encuentra en la situación actual (q_m, s_m) y encuentra una instrucción $i_j : q_m, s_m, s_n, m, q_n$, entonces cambia s_m por s_n, realiza el movimiento indicado por 'm' y pasa al estado q_n (puede ocurrir que $q_m = q_n$ ó $s_m = s_n$); de lo contrario la máquina finaliza su ejecución.

Veamos algunos ejemplos de máquinas de Turing y del comportamiento de las mismas:

Ejemplo No. 1:

Sea $\mathbf{M}_1 = <K, \Sigma, M, q_x, I>$ una Máquina de Turing definida por:

$K = \{q_0, q_1, q_2, q_3\}$
$\Sigma = \{0, 1\}$ [15]
$q_x = q_0$
$I = \{i_0, i_1, i_2, i_3\}$ donde:

$i_0 : q_0\ \square\ 0\ R\ q_1$
$i_1 : q_1\ \square\square\ R\ q_2$
$i_2 : q_2\ \square\ 1\ R\ q_3$
$i_3 : q_3\ \square\square\ R\ q_0$

Antes de comenzar la ejecución de $\mathbf{M}_1$, necesitamos definir la secuencia de símbolos iniciales en la cinta

---

(10) Este artículo hace referencia exclusivamente a máquinas de Turing determinísticas, por la tanto la palabra 'determinísticas' se omitirá desde este momento.

(11) Esta definición puede tener varias variantes de acuerdo al "gusto" de cada autor. Una de ellas es definir la máquina de Turing como un conjunto finito instrucciones, ya que este conjunto da implícitamente el conjunto de símbolos utilizados y los posibles estados de la máquina.

(12) Todas las máquinas de este artículo operan sobre un cinta unidimensional bi-infinita. De ahí que este conjunto es el mismo para todas.
Hemos utilizado las iniciales de los nombres en inglés de cada movimiento, para poder ilustrar la codificación y la enumeración originales construidas por Turing para sus máquinas (ver sección 5).

(13) El lector debe notar que la función I está definida sobre un subconjunto de $K \times \Sigma$, porque no es necesario que exista una instrucción para cada una de las situaciones teóricas (es el conjunto formado por $K \times \Sigma$) en las que puede estar la máquina. Además el concepto de función refleja la característica de las máquinas de Turing determinísticas.

(14) La finitud de este conjunto está determinada implícitamente por la finitud del conjunto de estados (K) y la finitud del alfabeto (Σ).

(15) Recordamos al lector que por convención, el símbolo vacío ($\square$) pertenece al alfabeto.

y la posición inicial de la máquina sobre la cinta. Inicialmente la cinta se encuentra vacía, la máquina está en alguna posición de la cinta (representada por la celda sombreada) y además se encuentra en el estado inicial q_0.

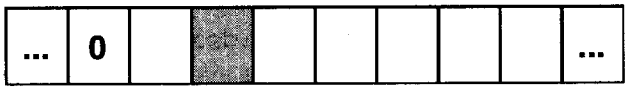
Estado actual: q_0

Comencemos la ejecución: La máquina busca en su conjunto de instrucciones una instrucción que comience por la situación actual (q_0, □) y encuentra la instrucción 'q_0 □ 0 R q_1' (instrucción i_0), entonces la máquina procede a cambiar □ por 0, se mueve una posición a la derecha y pasa al estado q_1. Después de ejecutar la primera instrucción tenemos la siguiente situación:

Estado actual: q_1

Ahora la máquina está en la situación actual (q_1, □) para la cual corresponde la instrucción 'q_1 □□ R q_2' (instrucción i_1), entonces la máquina se mueve a la derecha y pasa al estado q_2 (porque la instrucción indica que cambie □ por □ lo que se traduce en no escribir nada). Ahora tenemos:

Estado actual: q_2

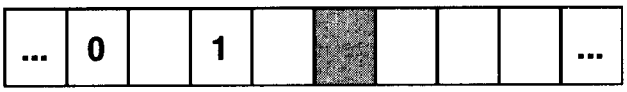


Para la situación actual (q_2, □) corresponde la instrucción 'q_2 □ 1 R q_3' (instrucción i_2), entonces la máquina cambia □ por 1, se mueve a la derecha y pasa al estado q_3.

Estado actual: q_3

Ahora para la situación actual (q_3, □) corresponde la instrucción 'q_3 □□ R q_0' (instrucción i_3), que le dice a la máquina que se mueva a la derecha y pase al estado q_0.

Estado actual: q_0



En este punto la máquina se encuentra de nuevo en la situación (q_0, □) y repite el ciclo indefinidamente.

La tabla de la página siguiente muestra el comportamiento de la máquina $\mathbf{M}_1$ para la ejecución de sus primeros "pasos".

Algunas observaciones:
Particularmente esta máquina no se detiene nunca, además utiliza la convención propuesta por Turing de "trabajar" en celdas intercaladas, esto con el propósito de utilizar los espacios entre las celdas ocupadas para colocar "marcas" temporales que se necesiten durante la ejecución. Pero no todas las máquinas tienen estas características, veamos otro ejemplo:

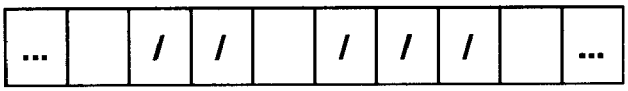
Ejemplo No. 2

Codifiquemos los números naturales en el alfabeto de "palitos" de la siguiente forma:

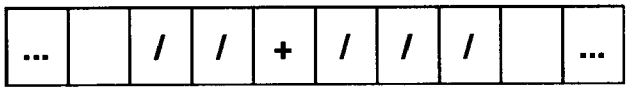
1 = /
2 = //
3 = ///
.
.
.
n = /.../ (n palitos)

La máquina que vamos a construir calcula la suma de dos números naturales representados en el código "palitos".

Uno de los aspectos a tener en cuenta en el diseño de una Máquina de Turing es la forma cómo están representados en la cinta los datos de entrada. Supongamos que deseamos realizar la suma de 2 y 3. Algunas de la representación que podríamos tener son:

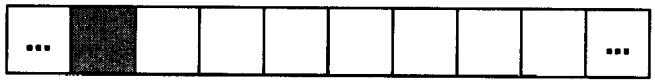
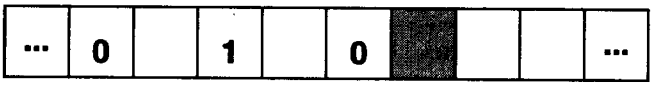
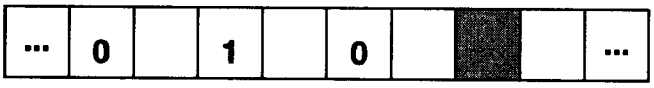
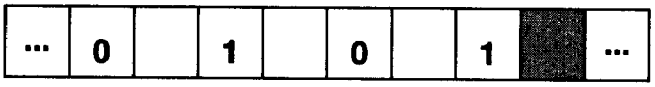


En este caso los números a sumar están separados por un espacio en blanco.



En este caso los números a sumar están separados por un signo '+'.

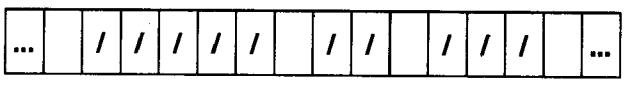
Tabla: Comportamiento máquina $\mathbf{M}_1$:

Paso	Estado actual	Instrucción ejecutada	
0	q_0		
1	q_1	i_0	
2	q_2	i_1	
3	q_3	i_2	
4	q_0	i_3	
5	q_1	i_0	
6	q_2	i_1	
7	q_3	i_2	
.	.	.	
.	.	.	
.	.	.	

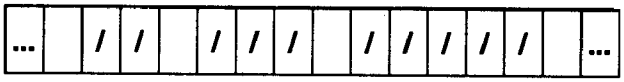
En este caso los números a sumar están separados por n espacios en blanco.

Hemos escogido la primera representación para construir nuestra máquina sumadora de palitos.

Otro de los aspectos a considerar es la forma en que se representará en la cinta la respuesta generada por la máquina, para nuestro caso algunas representaciones podrían ser:



En este caso la respuesta aparece a la izquierda de los datos de entrada.



En este caso la respuesta aparece a la derecha de los datos de entrada.

Y en este caso la respuesta sobreescribe los datos de entrada. Ésta va a ser nuestra elección.

Nuestra máquina está definida por:

$$\mathbf{M}_2 = \langle K, \Sigma, M, q_x, I \rangle \text{ donde:}$$

$$K = \{q_0, q_1, q_2, \textit{stop}\}$$

Antes de continuar, hablemos un poco acerca de este estado *stop* en nuestra máquina. En la construcción de gran parte de máquinas se espera que éstas finalicen (a diferencia de nuestro primer ejemplo). Es necesario conocer si dicha finalización fue exitosa o no, es decir, si la máquina finalizó porque estaba programada para ello o finalizó porque no encontró una instrucción para la situación actual en la que se encontraba. Para esto se acostumbra dotar a la máquina de un estado de parada exitosa normalmente llamado *stop*, entonces cuando es alcanzado, la máquina finaliza su ejecución (pues no existe ninguna instrucción que contemple a *stop* como su primer componente), pero el diseñador de la máquina sabe que fue una terminación exitosa. Observe el lector que el uso de esta convención no modifica para nada lo dicho hasta el momento.

Continuemos con la definición de nuestra máquina

$\Sigma = \{/\}$; $q_x = q_0$; $I = \{i_0, i_1, i_2, i_3, i_4\}$ donde:

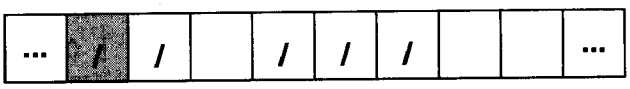
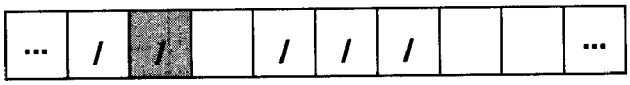
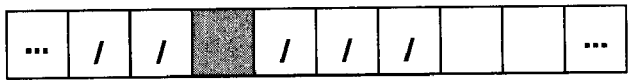
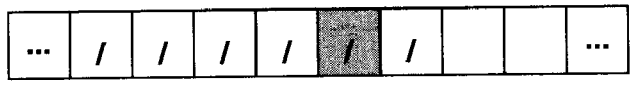
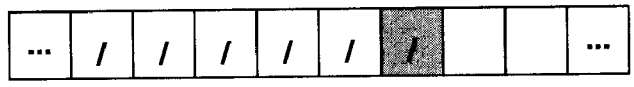
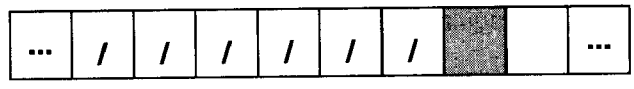
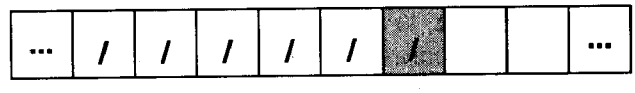
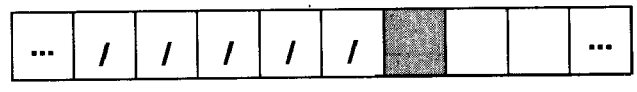
$i_0 : q_0 \,//\, R\, q_0$; $i_1 : q_0\, \square / R\, q_1$ recorren el primer número (hasta el primer espacio en blanco) (16).

$i_2 : q_1 \,//\, R\, q_1$; $i_3 : q_1\, \square\square L\, q_2$ recorren el segundo número (hasta el siguiente espacio en blanco).

$i_4 : q_2 / \square\, N\, stop$; elimina el último palito del segundo número y se detiene.

Inicialmente la cinta contiene dos números naturales representados en el código "palitos" separados por un espacio en blanco. La máquina se encuentra situada sobre el primer palito de izquierda a derecha. Veamos el comportamiento de la máquina $\mathbf{M}_2$:

Tabla: Comportamiento máquina $\mathbf{M}_2$, suma de 2 y 3 en notación "palitos":

Paso	Estado actual	Instrucción ejecutada	
0	q_0		
1	q_0	I_0	
2	q_0	I_0	
3	q_1	I_0	
4	q_1	I_1	
5	q_1	I_1	
6	q_1	I_1	
7	q_2	I_3	
8	*stop*	I_4	

(16) En algunas ocasiones se acostumbra documentar la tarea que realiza una instrucción o un conjunto de éstas.

5. ENUMERACIÓN Y CODIFICACIÓN DE LAS MÁQUINAS DE TURING

Para desarrollar la solución al caso general del PROBLEMA DE LA DECISIÓN, Turing utilizó una enumeración muy particular de sus máquinas. Esta enumeración tiene como base un mecanismo de codificación de las mismas. Por otro lado, la codificación de las máquinas es necesaria para definir un procedimiento que sea capaz de ejecutar cualquier MÁQUINA DE TURING (en la sección 7 desarrollaremos esta idea). Veamos entonces, cuál fue la codificación y la enumeración para sus máquinas, propuesta por Turing [17].

De acuerdo con la definición formal de una MÁQUINA DE TURING, tenemos el siguiente formato de instrucción:

$q_i\ s_j\ s_k\ m\ q_l$; donde i, j, k, l ≥ 0 y m ∈ M = { L, R, N }.

Entonces, reemplazamos cada instrucción de acuerdo a la siguiente codificación:

q_i : se reemplaza por una 'D' seguida de 'A' repetida 'i' veces

s_j : se reemplaza por una 'D' seguida de 'C' repetida 'j' veces

Entonces las instrucciones se reescriben utilizando este código y se separan por un ';'. Cuando describimos las instrucciones de nuestra máquina utilizando este sistema de codificación, decimos que la máquina está representada en su DESCRIPCIÓN ESTÁNDAR (*standard description* en el original). Las instrucciones de la descripción estándar estarán formadas con símbolos del siguiente alfabeto Σ = { A, C, D, R, L, N, ; }.

Asociamos además a cada símbolo del alfabeto un número natural por medio de la siguiente función:

$$f: \Sigma \rightarrow N \text{ donde } f(s) = \begin{cases} 1 & \text{si } s = A \\ 2 & \text{si } s = C \\ 3 & \text{si } s = D \\ 4 & \text{si } s = L \\ 5 & \text{si } s = R \\ 6 & \text{si } s = N \\ 7 & \text{si } s = ; \end{cases}$$

Entonces, remplazamos cada símbolo de la descripción estándar de una MÁQUINA DE TURING por el valor que tiene asociado y el número obtenido es llamado el NÚMERO DE DESCRIPCIÓN (*description number* en el original) de la máquina en cuestión.

Como ejemplo, hallemos la descripción estándar y el número de descripción para la máquina $\mathbf{M}_2$ (ejemplo No. 2):

El conjunto de instrucciones de $\mathbf{M}_2$ consta de:

i_0 : q_0 / / D q_0
i_1 : q_0 □ / D q_1
i_2 : q_1 / / D q_1
i_3 : q_1 □□ L q_2
i_4 : q_2 / □ N *stop*

Renombremos los símbolos utilizados por $\mathbf{M}_2$ así: □ por s_0 y / por s_1, además renombremos el estado *stop* por el estado q_3. Ahora podemos escribir de nuevo las instrucciones y obtener la descripción estándar de cada una de ellas:

	Instrucción original	Instrucciones renombradas	Descripción estándar
i_0	q_0 / / R q_0	$q_0\ s_1\ s_1$ R q_0	DDCDCRD
i_1	q_0 □ / R q_1	$q_0\ s_0\ s_1$ R q_1	DDDCRDA
i_2	q_1 / / R q_1	$q_0\ s_0\ s_1$ R q_1	DDDCRDA
i_3	q_1 □□ L q_2	$q_1\ s_0\ s_0$ L q_2	DADDLDAA
i_4	q_2 / □N q_3	$q_2\ s_1\ s_0$ N q_3	DAADCDNDAAA

(17) En la actualidad se han propuesto varias codificaciones y enumeraciones para las máquinas de Turing.

Con lo cual la descripción estándar para la máquina M_2 es (como mencionamos el ';' se utiliza para separar las instrucciones):
DDCDCRD;DDDCRDA;DDDCRDA;DADDLDAA;
DAADCDNDAAA;

y su número de descripción es:
33232537333253173332531731334311731132363111 7

Podemos observar que para cada MÁQUINA DE TURING existe un número natural que la representa, más no todo número natural representa una MÁQUINA DE TURING. Además de acuerdo con la enumeración anterior es posible demostrar que el conjunto de las máquinas de Turing es enumerable [18] (y en particular infinito), tal como lo es el conjunto de los ALGORITMOS que se pueden construir con algún lenguaje (y ésto debe ser así, si somos consistentes con la idea que cualquier algoritmo puede ser representado por una Máquina de Turing).

6. MÁQUINAS ESQUELETOS

Al obtener alguna experiencia en la construcción de máquinas de Turing, el lector observará que ciertos "procedimientos simples" usualmente hacen parte de éstas; por ejemplo los procedimientos de: buscar un símbolo particular, borrar una secuencia de símbolos, mover una secuencia de símbolos, entre otros. Por otra parte, máquinas que realicen algún procedimiento en particular pueden hacer parte de máquinas más complejas (por ejemplo, piense el lector en utilizar la máquina que suma para construir la máquina que multiplica).

Los lectores con experiencia en el desarrollo de *software* podrían pensar en la construcción de cierto tipo de "máquinas-procedimiento" que pudieran ser invocadas por otras máquinas. En el contexto de las máquinas de Turing, estas "máquinas-procedimiento" reciben el nombre de MÁQUINAS ESQUELETO (skeleton table en el original) [19] y "simplemente" se utilizan como ayuda para la especificación de otras máquinas de Turing.

(18) Un conjunto es enumerable si existe una función sobreyectiva del conjunto de los número naturales al conjunto en cuestión.

(19) Consideramos más conveniente usar el nombre de 'máquina esqueleto' en lugar de 'tabla esqueleto' como correspondería a la traducción.

Para utilizar MÁQUINAS ESQUELETO se ejecutan los siguientes pasos:

* Se construyen la(s) MÁQUINA(S) ESQUELETO necesarias.
* En el momento de la definición de las instrucciones de la MÁQUINA DE TURING se incorporan las meta-instrucciones [20] que realizan la invocación de la(s) máquina(s) esqueleto (por convención utilizaremos el nombre de meta-máquina cuando la MÁQUINA DE TURING en cuestión contenga meta-instrucciones y continuaremos con el nombre de MÁQUINA DE TURING cuando no las contenga).
* Antes de realizar la ejecución de la meta-máquina, es necesario realizar un proceso de expansión de la misma, el cual la convierte en una MÁQUINA DE TURING. La descripción de este proceso de expansión es demasiado engorrosa en su caso general, pero esperamos que el ejemplo presentado a continuación permita ilustrarlo.
* Finalizado este proceso de expansión se procede a ejecuta la MÁQUINA DE TURING resultante [21].

Construyamos una MÁQUINA ESQUELETO ME_1 que realice la siguiente tarea: Busca la primera ocurrencia de izquierda a derecha de un símbolo 's' en una secuencia de símbolos, limitada a la izquierda por 'a' y a la derecha por 'b'. Entonces, si ME_1 encuentra el símbolo 's' pasa al estado q_x y se detiene (situada sobre el símbolo 's'), si no lo encuentra, la máquina pasa al estado q_y y se detiene (situada sobre el símbolo 'b').

El conjunto de instrucciones para ME_1 está dado por [22]:

$$ME_1\ (s, a, b, q_x, q_y)$$

(20) Agregamos el prefijo "meta" para indicar que estas instrucciones no hacen parte de la definición formal de una máquina de Turing.

(21) Siendo estrictos la comparación entre las máquinas esqueleto y los procedimientos utilizados en el desarrollo de software es incorrecta. Como mencionamos, la máquina esqueleto debe ser "expandida" antes de su utilización, por lo cual resulta más conveniente hacer mención a la analogía entre una máquina esqueleto y un macro.

(22) La explicación de la simbología utilizada para construir estas instrucciones está descrita a continuación de las mismas.

i_0: q_n ¬ a ¬ a L q_n ; busca el límite izquierdo (símbolo 'a') avanzando hacia la izquierda hasta encontrarlo.

i_1: q_n a a N q_{n+1} ; encontró el límite izquierdo (símbolo 'a').

i_2: q_{n+1} s s N q_x ; encontró el símbolo 's', entonces pasa al estado q_x y termina.

i_3: q_{n+1} b b N q_y ; llegó al final de la secuencia (símbolo 'b'), ésto quiere decir que el símbolo 's' no está entre la secuencia buscada, entonces la máquina pasa al estado q_y y termina.

i_4: q_{n+1} ¬ s ¬ s R q_{n+1} ; avanza hacia la derecha buscando el símbolo 's' una vez que ha analizado que el símbolo leído no es el límite derecho (símbolo 'b', instrucción i_3) ni el símbolo buscado (símbolo 's', instrucción i_2).

Observaciones importantes:

* Uso de parámetros

Aunque es posible construir MÁQUINAS ESQUELETO que no reciban parámetros, buena parte de la motivación para su uso, está fundamentada en que los pueden usar, característica que las dota de una mayor generalidad.

* Alfabeto y meta-instrucciones

Ya que es en la descripción de las instrucccio-nes de una meta-máquina donde se utilizan las máquinas esqueleto, es imposible que la MÁQUINA ESQUELETO, "conozca" apriori el alfabeto de la meta-máquina que la utiliza. Esto impide declarar explícitamente el alfabeto de la MÁQUINA ESQUELETO en cuestión. Esta situación nos obliga en algunos casos a utilizar meta-símbolos [23]. Como ejemplo, observe el lector la instrucción i_0 de la máquina $\mathbf{ME}_1$, el meta-símbolo en cuestión es '¬a' que en realidad hace referencia a los símbolos diferentes del símbolo 'a' (algo similar ocurre con el meta-símbolo '¬s' en la instrucción i_4). Entonces cuando se realice el proceso de expansión de la meta-máquina que utiliza la máquina $\mathbf{ME}_1$ será necesario, construir una instrucción por cada símbolo diferente de 'a' para expandir la instrucción i_0 y construir una instrucción por cada símbolo diferente de 's' para expandir la instrucción i_4. De acuerdo con la convención que estamos utilizando deberíamos hablar de META-MÁQUINAS ESQUELETO o de MÁQUINAS ESQUELETO con base en si la MÁQUINA ESQUELETO tiene o no tiene meta-instrucciones (construídas con meta-símbolos), pero continuaremos con el nombre de 'MÁQUINAS ESQUELETO'.

* Enumeración de estados

La meta-máquina puede utilizar en cualquier "punto" la MÁQUINA ESQUELETO. Por esto, la MÁQUINA ESQUELETO no puede hacer referencia a estados absolutos $\{q_1, q_2, q_3, ..., q_n\}$ ya que ellos pueden corresponder a estados de la meta-máquina que la utiliza. Como solución, las máquinas esqueleto hacen referencia a estados relativos $\{q_n, q_{n+1}, q_{n+2}, ..., q_{n+m}\}$ a los cuales se les asigna su valor absoluto en el momento de realizar la expansión de la meta-máquina que la utiliza (el cual involucra la expansión de la MÁQUINA ESQUELETO).

* Encadenamiento de máquinas esqueletos

Aunque no presentaremos un ejemplo de ésto, es posible construir una MÁQUINA ESQUELETO que a su vez esté compuesta de otra(s) máquina(s) esqueleto, que a su vez estén compuestas de otra(s) máquina(s) esqueleto y así sucesivamente. Es posible además, que uno de los estados de la MÁQUINA ESQUELETO consista en invocar una MÁQUINA ESQUELETO en la cual, uno de sus estados invoque también otra MÁQUINA ESQUELETO y así sucesivamente. Esto simplifica considerablemente el proceso de descripción, pero complica (en algunos casos, considerablemente también) el proceso de expansión. Esperamos que el lector comprenderá mejor la (considerable) complicación a la cual hacemos referencia, una vez hallamos expandido la meta-máquina utilizada como ejemplo.

(23) Utilizamos el prefijo "meta" para significar que estos meta-símbolos no representan símbolos individuales, sino por el contrario hacen referencia a un conjunto de símbolos con alguna característica en común.

* Precauciones en el uso de las máquinas esqueleto

Es necesario conocer para un correcto uso de las mismas, el comportamiento exacto de las MÁQUINAS ESQUELETO utilizadas, esto es, estado final, símbolo sobre el cual finaliza, etc. (para todos los posible casos). De lo contrario, la meta-máquina puede tener (Murphy garantiza que lo tendrá) un comportamiento imprevisto y como el lector imaginará, el proceso de depuración es un trabajo arduo de realizar.

Construyamos ahora una meta-máquina $\mathbf{MM}_1$ que invoque nuestra MÁQUINA ESQUELETO $\mathbf{ME}_1$ y realicemos el proceso de expansión para convertirla en una MÁQUINA DE TURING $\mathbf{M}_1$. (Durante este ejemplo utilizaremos el término 'instrucción', para designar tanto una instrucción como una meta-instrucción).

La meta-máquina $\mathbf{MM}_1$ va a operar sobre una secuencia de símbolos compuesta por vocales, limitada a la izquierda por el símbolo '1' y a la derecha por el símbolo '2'. $\mathbf{MM}_1$ va a remplazar la primera ocurrencia de izquierda a derecha, del símbolo 'o' por el símbolo '4' y la primera ocurrencia de izquierda a derecha, del símbolo 'u' por el símbolo '5'. Adicionalmente $\mathbf{MM}_1$ puede comenzar sobre cualquier símbolo de la secuencia inicial (incluidos los límites (símbolos '1' y '2')) [24].

El alfabeto de $\mathbf{MM}_1$ es $\Sigma = \{1, 2, 4, 5, a, e, i, o, u\}$

El conjunto de instrucciones está dado por [25]:

a_0: $\mathbf{ME}_1$ (o, 1, 2, q_x, q_y) ; busca la primera ocurrencia del símbolo 'o' en la secuencia inicial, si lo encuentra pasa al estado q_x, de lo contrario para al estado q_y.

a_1: q_x o 4 N $\mathbf{ME}_1$ (u, 1, 2, q_w, *stop*) ; cambia el símbolo 'o' por el símbolo '4' y busca la primera ocurrencia del símbolo 'u' en la secuencia, si lo encuentra pasa al estado q_w, de lo contrario pasa al estado *stop* (termina).

a_2: q_y 2 2 N $\mathbf{ME}_1$ (u, 1, 2, q_w, *stop*) ; el símbolo 'o' no se encontró, entonces busca la primera ocurrencia del símbolo 'u' en la secuencia, si lo encuentra pasa al estado q_w, de lo contrario pasa al estado stop (termina).

a_3: q_w u 5 N *stop* ; cambia el símbolo 'u' por el símbolo '5' y pasa al estado *stop* (termina).

Realicemos ahora la expansión de la meta-máquina $\mathbf{MM}_1$ para convertirla en una MÁQUINA DE TURING $\mathbf{M}_1$:

La primera instrucción: 'a_0: $\mathbf{ME}_1$ (o, 1, 2, q_x, q_y)' invoca la máquina esqueleto $\mathbf{ME}_1$ con unos valores específicos para sus parámetros, por lo que las instrucciones de $\mathbf{ME}_1$ adquieren la siguiente forma [26]:

---

(24) El comportamiento de esta meta-máquina es simple y sólo se construye con el propósito de ilustrar la utilización de máquinas esqueletos. Algún lector pensará que para realizar esta tarea no se necesita ninguna meta-máquina sino que por el contrario se puede construir la máquina de Turing directamente, estando nosotros de acuerdo con él (inclusive es posible simplificar la descripción de la meta-máquina).

(25) Enumeramos las instrucciones con a_0, a_1, a_2, y a_3 para evitar confusiones en el momento de realizar la expansión.

(26) Las "nuevas" instrucciones han sido enumeradas con p_0, p_1, p_2, p_3 y p_4, para evitar confusiones (indicadas en la página siguiente) .

Instrucciones originales de $\mathbf{ME}_1$ (s, a, b, q_x, q_y)	Instrucciones de $\mathbf{ME}_1$ con los parámetros (o, 1, 2, q_x, q_y) con los cuales ha sido invocada	Comentario
i_0: q_n ¬a ¬a L q_n i_1: q_n a a N q_{n+1} i_2: q_{n+1} s s N q_x i_3: q_{n+1} b b N q_y i_4: q_{n+1} ¬s ¬s R q_{n+1}	p_0: q_n ¬1 ¬1 L q_n p_1: q_n 1 1 N q_{n+1} p_2: q_{n+1} o o N q_x p_3: q_{n+1} 2 2 N q_y p_4: q_{n+1} ¬o ¬o R q_{n+1}	Remplazamos los símbolos en las instrucciones originales, de acuerdo con los parámetros, así: 's' por 'o', a' por '1', 'b' por '2', 'q_x' por 'q_x' y 'q_y' por 'q_y'

Ahora expandamos las instrucciones p_0, p_1, p_2, p_3 y p_4 [27].

Instrucción	Instrucciones de $\mathbf{M}_1$ (resultado de la expansión)	Comentario
p_0: q_n ¬1 ¬1 L q_n	f_0: q_0 a a L q_0 f_1: q_0 e e L q_0 f_2: q_0 i i L q_0 f_3: q_0 o o L q_0 f_4: q_0 u u L q_0 f_5: q_0 2 2 L q_0	En este caso la instrucción con el meta-símbolo '¬1', ha sido expandida a instrucciones que analicen todos los símbolos del alfabeto, excepto el '1'. Además se ha reemplazado el estado relativo q_n por el estado absoluto q_0. (Esta asignación de estados se realizará durante todo el proceso de expansión).
p_1: q_n 1 1 N q_{n+1}	f_6: q_0 1 1 R q_1	
p_2: q_{n+1} o o N q_x	f_7: q_1 o o N q_2	Al estado que representa que si se encontró el símbolo 'o' se le asignó el valor q_2.
p_3: q_{n+1} 2 2 N q_y	f_8: q_1 2 2 N q_3	Al estado que representa que no se encontró el símbolo 'o' se le asignó el valor q_3.
p_4: q_{n+1} ¬o ¬o R q_{n+1}	f_9: q_1 a a R q_1 f_{10}: q_1 e e R q_1 f_{11}: q_1 i i R q_1 f_{12}: q_1 u u R q_1	Similar a la expansión de la instrucción p_0, con la diferencia de que no se analiza el símbolo '2' (que fue analizado en f_8) y tampoco se analiza el símbolo '1' en razón del desplazamiento efectuado a la derecha por la instrucción f_6.

Hasta el momento la máquina $\mathbf{M}_1$ está formada por las instrucciones f_0, f_1, ..., f_{12}.

Expandamos ahora la instrucción 'a_1: q_x o 4 N $\mathbf{ME}_1$ (u, 1, 2, q_w, *stop*)'

Instrucción	Instrucciones $\mathbf{M}_1$ (resultado de la expansión)
a_1: q_x o 4 N $\mathbf{ME}_1$ (u, 1, 2, q_w, *stop*)	f_{13}: q_2 0 4 N q_4

(27) Las instrucciones de la máquina "final" $\mathbf{M}_1$ se enumeran con f_0, f_1, . . . , f_n para evitar confusiones.

Comentario: El estado q_x de la instrucción a_1 se le asignó el estado q_2, como lo mencionamos en la instrucción f_7. Además el estado siguiente de la instrucción a_1 consiste en invocar la *máquina esqueleto* $\mathbf{ME}_1$ con nuevos parámetros, lo cual nos conduce a realizar un expansión similar a la efectuada con la instrucción a_0. En este caso el estado q_4 de la instrucción f_{13} es utilizado de puente entre el "comienzo" de la instrucción a1 y el final de la misma ($\mathbf{ME}_1$ (u, 1, 2, q_w, *stop*)).

Expandamos entonces la instrucción $\mathbf{ME}_1$ (u, 1, 2, q_w, *stop*):

Instrucciones originales de $\mathbf{ME}_1$ (s, a, b, q_x, q_y)	**Instrucciones de $\mathbf{ME}_1$ con los parámetros (u, 1, 2, q_w, *stop*) con los cuales ha sido invocada**	**Comentario**
i_0: q_n ¬a ¬a L q_n i_1: q_n a a N q_{n+1} i_2: q_{n+1} s s N q_x i_3: q_{n+1} b b N q_y i_4: q_{n+1} ¬s ¬s R q_{n+1}	s_0: q_n ¬1 ¬1 L q_n s_1: q_n 1 1 N q_{n+1} s_2: q_{n+1} u u N q_w s_3: q_{n+1} 2 2 N *stop* s_4: q_{n+1} ¬u ¬u R q_{n+1}	Remplazamos los símbolos en las instrucciones originales, de acuerdo con los parámetros, así: 's' por 'u', 'a' por '1', 'b' por '2', 'q_x' por 'q_w' y 'q_y' por '*stop*'.

Entonces la expansión de las instrucciones s_0, s_1, s_2, s_3 y s_4 es:

Instrucción	**Instrucciones de $\mathbf{M}_1$ (resultado de la expansión)**	**Comentario**
s_0: q_n ¬1 ¬1 L q_n	f_{14}: q_4 a a L q_4 f_{15}: q_4 e e L q_4 f_{16}: q_4 i i L q_4 f_{17}: q_4 u u L q_4 f_{18}: q_4 2 2 L q_4 f_{19}: q_4 4 4 L q_4	A diferencia de la primera expansión de esta instrucción, (expansión de la instrucción p_0) en este caso se debe analizar el símbolo '4'.
s_1: q_n 1 1 N q_{n+1}	f_{20}: q_4 1 1 R q_5	
s_2: q_{n+1} u u N q_w	f_{21}: q_5 u u N q_6	Al estado que representa que si se encontró el símbolo 'u' se le asignó el valor q_6.
s_3: q_{n+1} 2 2 N *stop*	f_{22}: q_5 2 2 N *stop*	Al estado que representa que no se encontró el símbolo 'u' se le asignó el valor '*stop*'.
s_4: q_{n+1} ¬u ¬u R q_{n+1}	f_{23}: q_5 a a R q_5 f_{24}: q_5 e e R q_5 f_{25}: q_5 i i R q_5 f_{26}: q_5 o o R q_5 f_{27}: q_5 4 4 R q_5	Similar a la expansión de la instrucción s_0, con la diferencia de que no se analiza el símbolo '2' (que fue analizado en la instrucción f_{22}) y tampoco se analiza el símbolo '1' en razón del desplazamiento efectuado a la derecha por la instrucción f_{20}.

Hasta el momento la máquina $\mathbf{M}_1$ está formada por las instrucciones f_0, f_1, ..., f_{27}.

Expandamos ahora la instrucción 'a_2: q_y 2 2 N $\mathbf{ME}_1$ (u, 1, 2, qw, *stop*)'.

Instrucción	Instrucciones $\mathbf{M}_1$ (resultado de la expansión)
a_2: q_y 2 2 N $\mathbf{ME}_1$ (u, 1, 2, q_w, *stop*)	f_{28}: q_3 2 2 N q_4

Comentario: Esta instrucción representa el caso de que no se haya encontrado el símbolo 'o', con lo cual el estado q_y de la instrucción a_2 es el estado q_3, de acuerdo con la instrucción f_8. Además el estado siguiente de la instrucción a_2 es el mismo estado siguiente de la instrucción a_1, (que acabamos de expandir), entonces simplemente utilizamos el puente (estado q_4) definido en la expasión de la instrucción a_1 para este caso y damos por expandida la instrucción a_2.

Para finalizar, la expansión de la instrucción 'a_3: q_w u 5 N *stop*' es:

Instrucción	Instrucciones $\mathbf{M}_1$ (resultado de la expansión)
a_3: q_w u 5 N *stop*	f_{29}: q_6 u 5 N *stop*

Comentario: El estado q_x de la instrucción a_3 corresponde al estado q_6 de acuerdo a la instrucción f_{21}.

Entonces la máquina $\mathbf{M}_1$ está definida por:

$\Sigma = \{1, 2, 4, 5, a, e, i, o, u\}$;
$K = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, stop\}$; $q_x = q_0$;
$I = \{f_0, ..., f_{29}\}$ donde:

f_0: q_0 a a L q_0	f_{10}: q_1 e e R q_1	f_{20}: q_4 1 1 R q_5
f_1: q_0 e e L q_0	f_{11}: q_1 i i R q_1	f_{21}: q_5 u u N q_6
f_2: q_0 i i L q_0	f_{12}: q_1 u u R q_1	f_{22}: q_5 2 2 N *stop*
f_3: q_0 o o L q_0	f_{13}: q_2 0 4 N q_4	f_{23}: q_5 a a R q_5
f_4: q_0 u u L q_0	f_{14}: q_4 a a L q_4	f_{24}: q_5 e e R q_5
f_5: q_0 2 2 L q_0	f_{15}: q_4 e e L q_4	f_{25}: q_5 i i R q_5
f_6: q_0 1 1 R q_1	f_{16}: q_4 i i L q_4	f_{26}: q_5 o o R q_5
f_7: q_1 o o N q_2	f_{17}: q_4 u u L q_4	f_{27}: q_5 4 4 R q_5
f_8: q_1 2 2 N q_3	f_{18}: q_4 2 2 L q_4	f_{28}: q_3 2 2 N q_4
f_9: q_1 a a R q_1	f_{19}: q_4 4 4 L q_4	f_{29}: q_6 u 5 N *stop*

Esperamos que este ejemplo ilustre al lector sobre la potencia y simplicidad que produce la utilización de MÁQUINAS ESQUELETO. En nuestro caso una meta-máquina de cuatro instrucciones { a_0, a_1, a_2, a_3 } produjo una MÁQUINA DE TURING con 30 instrucciones { f_0, f_1, ..., f_{29}}.

7. MÁQUINA UNIVERSAL DE TURING

Alan Turing no sólo construyó la noción del concepto de ALGORITMO por medio de sus máquinas de Turing, sino que además (en el mismo artículo donde describió éstas), construyó el modelo teórico de nuestros computadores actuales por medio de una máquina abstracta conocida en nuestros días como MÁQUINA UNIVERSAL DE TURING [28].

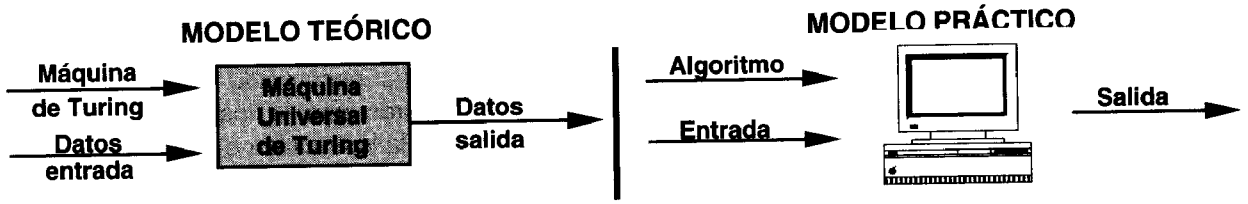
La MÁQUINA UNIVERSAL DE TURING es una MÁQUINA DE TURING muy particular. Recibe como entrada una MÁQUINA DE TURING y una secuencia de datos iniciales para la misma, entonces realiza la ejecución de la MÁQUINA DE TURING con la secuencia de datos dada.

En notación funcional, expresamos lo que hace una MÁQUINA UNIVERSAL DE TURING por: Sean $\mathbf{U}$ la MÁQUINA UNIVERSAL DE TURING, $\mathbf{M}$ una MÁQUINA DE TURING, α una secuencia de datos iniciales y β una secuencia de datos de salida, entonces: $\mathbf{U}(\mathbf{M}, \alpha) = \beta$ si y sólo si $\mathbf{M}(\alpha) = \beta$.

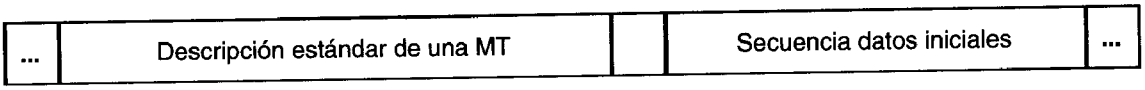
> Alan Turing no sólo construyó la noción del concepto de algoritmo por medio de sus máquinas de Turing, sino que además construyó el modelo teórico de nuestros computadores actuales por medio de una máquina abstracta conocida en nuestros días como Máquina Universal de Turing.

(28) Posteriormente a estos trabajos, Turing dirigió su interés hacia el área de la Inteligencia Artificial. En esta área se destaca su *test* para verificar si un computador presenta inteligencia, conocido como la *verificación de Turing*.

En el lenguaje informático expresamos lo que hace una MÁQUINA UNIVERSAL DE TURING por: Un computador (MÁQUINA UNIVERSAL DE TURING) ejecuta un ALGORITMO (MÁQUINA DE TURING) con una entrada (secuencia de datos iniciales) y produce una salida (datos de salida), es decir:



Veamos cómo opera la MÁQUINA UNIVERSAL DE TURING:
Como mencionamos la MÁQUINA UNIVERSAL DE TURING es una MÁQUINA DE TURING (muy particular). Antes de comenzar su ejecución, los datos de entrada (Máquina de Turing, secuencia de datos iniciales) deben estar en la cinta. La MÁQUINA DE TURING actúa como primer dato de entrada, y debe estar expresada en su descripción estándar (ver sección 5), a su derecha se escribe la secuencia de datos iniciales. Es decir, antes de comenzar la ejecución de la Máquina Universal de Turing, la cinta debe contener la siguiente información:



La MÁQUINA UNIVERSAL DE TURING está compuesta por un conjunto de instrucciones que "saben" cómo "ejecutar" la MÁQUINA DE TURING (expresada en su descripción estándar*)* con la secuencia de datos iniciales. La descripción exacta del comportamiento de MÁQUINA UNIVERSAL DE TURING es bastante compleja y no queremos agobiar al lector con los detalles. Como resumen podemos decir lo siguiente: La MÁQUINA UNIVERSAL DE TURING debe "capturar" la situación actual (estado actual, símbolo actual) de la MÁQUINA DE TURING; entonces debe ir a buscar una instrucción para esta situación actual, en la sección de la descripción estándar de la máquina; si la encuentra debe realizar lo que esta instrucción indica, en la sección de la secuencia de datos iniciales; "recordando" cual era la posición de la máquina que está ejecutando sobre los mismos, entonces de nuevo va y busca una instrucción para la nueva situación actual y así continua su proceso, hasta que la MÁQUINA DE TURING se detenga (si éste es el caso).

Existen varios "detalles" a tener en cuenta como por ejemplo: ¿qué pasa si la MÁQUINA DE TURING escribe sus resultados a la izquierda de la secuencia de datos iniciales?. Si éste es el caso, la MÁQUINA DE TURING despúes de escribir su primer resultado, comenzará a "dañar" su propia descripción estándar, lo cual por supuesto no se debe permitir. Para evitar esta situación se proponen dos soluciones: o se modifica la MÁQUINA DE TURING para que escriba sus resultados a la derecha de la secuencia de datos iniciales o se modifica la MÁQUINA UNIVERSAL DE TURING para que suministre tantas celdas en blanco a la izquierda de la secuencia de datos iniciales como necesite la MÁQUINA DE TURING que está ejecutando (usualmente se opta por la primera solución).

> Alan Turing asentó los cimientos teóricos de este edificio que hoy llamamos informática, al ofrecer una noción para los conceptos de algoritmo y de computador universal (aunque como ocurre frecuentemente en la ciencia, éste no era su objetivo inicial). Es por esta razón que la ciencia en general y en particular la informática lo consideran uno de sus mayores benefactores.

8. EL PROBLEMA DE LA PARADA

El problema de la decisión en su caso general fue replanteado por Turing en términos de sus máquinas así: ¿Es posible construir una MÁQUINA DE TURING **P** que sea capaz de responder si una

MÁQUINA DE TURING se detendrá o no para una secuencia de datos iniciales determinada?, es decir:

Sean **M** una *máquina de Turing* y α una secuencia de datos iniciales, ¿existe una *máquina de Turing* **P**, tal que:

$$\mathbf{P}(\mathbf{M},\alpha) = \begin{cases} 1 \text{ si } \mathbf{M} \text{ se detiene con } \alpha \\ 0 \text{ si } \mathbf{M} \text{ no se detiene con } \alpha \end{cases}$$

El replanteamiento realizado por Turing al PROBLEMA DE LA DECISIÓN se conoce como el PROBLEMA DE LA PARADA.

El lector observará que para ciertas Máquinas de Turing y para ciertas secuencias de datos iniciales, es posible determinar si la máquina se detendrá o no. Por ejemplo, para la máquina $\mathbf{M}_2$ (construida en el ejemplo No. 2, sección 4) y una secuencia de datos iniciales constituida por dos números naturales expresados en el código "palitos", no es difícil construir la máquina **P**. Esta es una solución al PROBLEMA DE LA PARADA para un caso particular y en realidad interesa la solución al PROBLEMA DE LA PARADA para su caso general, es decir, ¿es posible construir una máquina **P** que resuelva el PROBLEMA DE LA PARADA, para cualquier MÁQUINA DE TURING y para cualquier secuencia de datos iniciales?

Turing demostró por un procedimiento de *reduction an absurdo* que no es posible construir dicha máquina **P**, lo cual expresado en términos del PROBLEMA DE LA DECISIÓN, quiere decir que este problema en el caso general no tiene solución.

9. CONCLUSIONES

Alan Turing asentó los cimientos teóricos de este edificio que hoy llamamos INFORMÁTICA, al ofrecer una noción para los conceptos de ALGORITMO y de COMPUTADOR UNIVERSAL (aunque como ocurre frecuentemente en la ciencia, éste no era su objetivo inicial). Es por esta razón que la ciencia en general y en particular la informática lo consideran uno de sus mayores benefactores. Como un pequeño reconocimiento a su labor la ACM *(Association for Computer Machine)* estableció desde 1966 el *Turing Award,* un premio que se otorga cada año a la(s) persona(s) que haya realizado un aporte(s) significativo(s) a la INFORMÁTICA.

Por otra parte y desde un punto de vista más general, estamos convencidos que una "verdadera" revolución informática se producirá no a partir de nuevos desarrollos tecnológicos (sin desconocer por esto su importancia), sino por el contrario, germinará como resultado del desarrollo de nuevos conceptos y modelos teóricos. Ésta es para nosotros, una de las razones principales por lo cual el estudio y desarrollo del área del saber que llamamos INFORMÁTICA TEÓRICA presenta importancia y vigencia actuales [29], y en particular, es una de las razones por la cual ofrecemos (modestamente) esta área a nuestros estudiantes de Ingeniería de Sistemas, convencidos en la necesidad que tiene el país de contar con profesionales que "no solamente" tengan la capacidad de adquirir nuevos conocimientos, sino que además cuenten con la capacidad de producirlos.

10. BIBLIOGRAFÍA [30]

[Cai90] Caicedo, Xavier. Elementos de lógica y calculabilidad. 2ed. Santafé de Bogotá: Una empresa docente, Universidad de Los Andes, págs. 254-321, 1990.

[Hop84] Hopcroft, Jhon. Máquinas de Turing. En: Investigación y Ciencia. Julio 1984. págs. 8-19, 1984.

[Kle74] Kleene, Stephen C. Introducción a la metamatemática. Madrid: Editorial Tecnos. págs.495, 1974.

[Pen95] Penrose, Roger. La nueva mente del emperador. Barcelona: Grijalbo Mondadori, págs. 56-106, 1995.

[TCWG96] Theory of Computing Working Group. Strategic Direction for Research in Theory of Computing: Preliminary Report.
En: http://geisel.csl.uiuc.edu/~loui/sdcr.html

[Tur36] Turing, Alan. On computable numbers, with an application to the Entscheidungsproblem. En: Proc. London Math. Soc. ser. 2, vol. 42. 1936-1937. págs. 230 - 265. A correction, ibid, vol 43. 1937. págs. 544-546, 1936.

---

(29) [TCWG96] ofrece una perspectiva muy actual, acerca de la importancia de la informática teórica.

(30) En Internert se encuentran muy buenas biografías de Alan M. Turing y también algunos simuladores (share y free software) de máquinas de Turing, para diferentes plataformas. En razón de que con algún "motor de búsqueda" es posible encontrarlos, no decidimos incluir sus direcciones en esta bibliografía (además éstas cambian frecuentemente).