Universidad Nacional Autónoma de México
Facultad de Ciencias

Complejidad Computacional

Tarea 3

Bonilla Reyes Dafne - 319089660
García Ponce José Camilo - 319210536

Máquinas de Turing

Ejercicio 1

Sea M una Máquina de Turing cuyo tiempo de ejecución está dado por:

$$f_M(n) = 2n^3(n+3)(n-4)$$

¿Cuáles de las siguientes afirmaciones son verdaderas? Justifica tu respuesta.

a. $f_M(n) \in O(n)$

Para comenzar, primero revisemos las definiciones formales de O, Θ:

> O
>
> *Una función $f(n)$ es $O(g(n))$ si existen constantes positivas c y n_0 tales que para todo $n \geq n_0$ se cumple que*
>
> $$0 \leq f(n) \leq c \cdot g(n)$$
>
> Esto significa que $f(n)$ crece a lo sumo tan rápido como $g(n)$ para valores grandes de n.

> Θ
>
> *Una función $f(n)$ es $\Theta(g(n))$ si existen constantes positivas c_1, c_2 y n_0 tales que para todo $n \geq n_0$ se cumple que*
>
> $$c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$$
>
> Esto significa que $f(n)$ crece a la misma tasa que $g(n)$ para valores grandes de n.

Ahora, notemos que la función dada se encuentra factorizada, por lo que para conocer su grado, comenzaremos por simplificarla:

$$f_M(n) = 2n^3(n+3)(n-4) \Rightarrow 2n^3(n+3)(n-4) = 2n^3(n^2 - 4n + 3n - 12) = 2n^3(n^2 - n - 12)$$
$$\Rightarrow f_M(n) = 2n^3(n^2) - 2n^3(n) - 2n^3(12) \Rightarrow f_M(n) = 2n^5 - 2n^4 - 24n^3$$

De esta forma, podemos ver el término con mayor exponente de la función es $2n^5$, por lo que podemos decir que $f_M(n)$ crece de manera proporcional a n^5. Ahora bien, revisemos si $f_M(n) \in O(n)$ de manera formal.

A partir de la definición formal de O, notemos que para que $f_M(n) \in O(n)$, tenemos que mostrar que existe una constante $c > 0$ tal que:

$$2n^5 - 2n^4 - 24n^3 \leq c \cdot n, \ \forall n \geq n_0$$

Para ello, supongamos qué $f_M(n) \in O(n)$. Entonces, si simplificamos la desigualdad dividiendo ambos lados por n obtenemos:

$$2n^4 - 2n^3 - 24n^2 \leq c$$

Sin embargo, notemos que esta desigualdad nunca se cumple, ya que del lado izquierdo tenemos términos exponenciales que crecen rápidamente a medida que n aumenta, mientras que el lado derecho solo tiene una constante, lo que implica que no es posible encontrar una constante c tal que cumpla la desigualdad dada.

$\therefore \ f_M(n) \notin O(n)$.

b. $f_M(n) \in O(n^6)$

Dado que en el inciso anterior obtuvimos la forma simplificada de la función $f_M(n)$ y dimos la definición formal de O, solo queda revisar si $f_M(n) \in O(n^6)$ de manera formal, por lo que para ello tendremos que mostrar que existe una constante $c > 0$ tal que:

$$2n^5 - 2n^4 - 24n^3 \leq c \cdot n^6, \ \forall n \geq n_0$$

Simplifiquemos la desigualdad dividiendo ambos lados por n^6:

$$\frac{2n^5}{n^6} - \frac{2n^4}{n^6} - \frac{24n^3}{n^6} \leq c \Rightarrow \frac{2}{n} - \frac{2}{n^2} - \frac{24}{n^3} \leq c$$

Aquí, notemos que a medida que $n \to \infty$, los términos $\frac{2}{n}, \frac{2}{n^2}$ y $\frac{24}{n^3}$ tienden a 0. Esto significa que, para valores suficientemente grandes de n, la desigualdad es satisfecha por algún valor constante c.

$\therefore \ f_M(n) \in O(n^6)$.

c. $f_M(n) \in O(n^5 \log n)$

Dado que en el inciso a) obtuvimos la forma simplificada de la función $f_M(n)$ y dimos la definición formal de O, solo queda revisar si $f_M(n) \in O(n^6)$ de manera formal, por lo que para ello tendremos que mostrar que existe una constante $c > 0$ tal que:

$$2n^5 - 2n^4 - 24n^3 \leq c \cdot n^5 \log n, \ \forall n \geq n_0$$

Simplifiquemos la desigualdad dividiendo ambos lados por n^5:

$$\frac{2n^5}{n^5} - \frac{2n^4}{n^5} - \frac{24n^3}{n^5} \leq c \log n \Rightarrow 2 - \frac{2}{n} - \frac{24}{n^2} \leq c \log n$$

Aquí, notemos que a medida que $n \to \infty$, los términos $\frac{2}{n}$ y $\frac{24}{n^2}$ tienden a 0, por lo que tenemos que:

$$2 \leq c \log n$$

Esto significa que, para valores suficientemente grandes de n, la desigualdad es satisfecha por algún valor constante c.

$\therefore\ f_M(n) \in O(n^5 \log n)$.

d. $f_M(n) \in \Theta(n^6)$

Dado que en el inciso a) obtuvimos la forma simplificada de la función $f_M(n)$ y dimos la definición formal de Θ, solo queda revisar si $f_M(n) \in \Theta(n^6)$ de manera formal, por lo que para ello tendremos que mostrar que existen constantes c_1 y c_2 tales que:

$$c_1 \cdot n^6 \leq 2n^5 - 2n^4 - 24n^3 \leq c_2 \cdot n^6,\ n \geq n_0$$

Simplifiquemos la desigualdad dividiendo ambos lados por n^6:

$$\frac{c_1 n^6}{n^6} \leq \frac{2n^5 - 2n^4 - 24n^3}{n^6} \leq \frac{c_2 n^6}{n^6} \Rightarrow c_1 \leq \frac{2}{n} - \frac{2}{n^2} - \frac{24}{n^3} \leq c_2$$

Aquí, notemos que a medida que $n \to \infty$, los términos $\frac{2}{n}$, $\frac{2}{n^2}$ y $\frac{24}{n^3}$ tienden a 0, lo que implica que el límite de la expresión es 0, es decir, no es posible encontrar una constante c_1 tal que:

$$c_1 \leq \frac{2}{n} - \frac{2}{n^2} - \frac{24}{n^3} \text{ para valores grandes de } n$$

Por lo que podemos decir que no existe una cota inferior positiva para $f_M(n)$ que crezca como n^6.

$\therefore f_M(n) \notin \Theta(n^6)$.

Ejercicio 2

Considera la Máquina de Turing definida por:

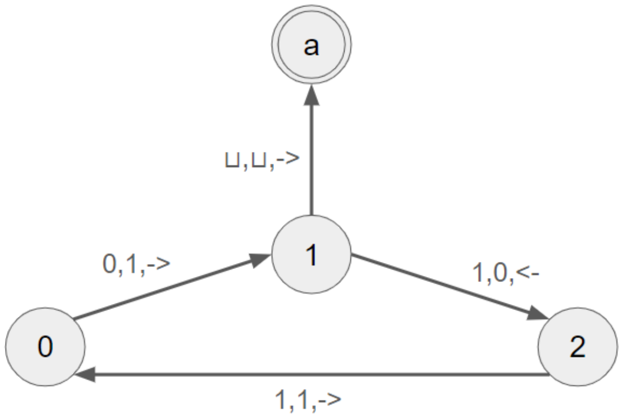
$$M = (\{q_0, q_1, q_2, q_a, q_r\}, \{0, 1\}, \{0, 1, \sqcup\}, \delta, q_0, q_a, q_r)$$

Describe el lenguaje L_M para cada una de las siguientes definiciones de δ. En cada inciso:

- Justifica tu respuesta agregando un par de ejecuciones (al menos una de aceptación y una de rechazo).
- Da la función $f_M(n)$ del tiempo de ejecución de la máquina.

a) $\delta(q_0, 0) = (q_1, 1, \rightarrow)$, $\delta(q_1, 1) = (q_2, 0, \leftarrow)$,
$\delta(q_2, 1) = (q_0, 1, \rightarrow)$, $\delta(q_1, \sqcup) = (q_a, \sqcup, \rightarrow)$.

El lenguaje L_M es $0(1)^*$, las cadenas que empiezan con un 0 y tienen ninguno o varios 1s. Esto lo notamos intentando varias cadenas y viendo esto:



Además, al final deja una cadena con la misma longitud que la original pero con puros 1s.

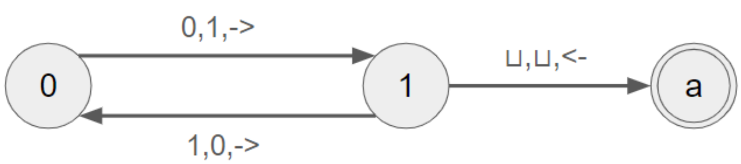
Ahora, veamos algunos ejemplos de ejecuciones:

- Cadena 010, la rechaza
  $q_0 010\sqcup \Rightarrow 1q_1 10\sqcup \Rightarrow q_2 100\sqcup \Rightarrow 1q_0 00\sqcup \Rightarrow 11q_1 0\sqcup \Rightarrow 11q_r 0\sqcup$
- Cadena 0111, la acepta
  $q_0 0111\sqcup \Rightarrow 1q_1 111\sqcup \Rightarrow q_2 1011\sqcup \Rightarrow 1q_0 011\sqcup \Rightarrow 11q_1 11\sqcup \Rightarrow 1q_2 101\sqcup \Rightarrow 11q_0 01\sqcup \Rightarrow 111q_1 1\sqcup \Rightarrow 11q_2 10\sqcup \Rightarrow 111q_0 0\sqcup \Rightarrow 1111q_1 \sqcup \Rightarrow 1111 \sqcup q_a$

Y $f_M(n)$ es $3(n-1)+2 = 3n-1$, esto debido a que para procesar el primer 0 toma dos transiciones la máquina y para procesar cada 1 de la cadena de entrada toma tres transiciones, y la cadena de entrada solo tiene $n-1$ cantidad de 1s, con n la longitud de la cadena de entrada.

b) $\delta(q_0, 0) = (q_1, 1, \rightarrow)$, $\delta(q_1, 1) = (q_0, 0, \rightarrow)$,
$\delta(q_1, \sqcup) = (q_a, \sqcup, \leftarrow)$.

El lenguaje L_M es $(01)^*0$, las cadenas que empiezan con un 0, terminan con un 0 y tienen un solo 1 entre cada dos 0s (es decir, no hay dos 0s juntos y tampoco dos 1s juntos). Esto lo notamos intentando varias cadenas y viendo esto:



Además, al final deja una cadena similar a la de entrada pero con los 0s cambiados por 1s y los 1s cambiados por 0s ($(10)^*1$).

Ahora, veamos algunos ejemplos de ejecuciones:

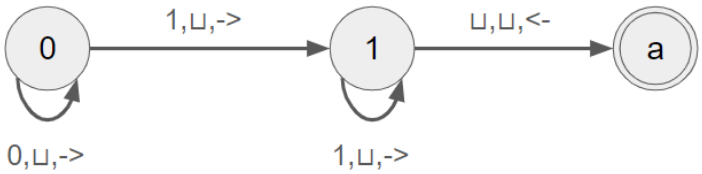
- Cadena 01011, la rechaza
  $q_0 01011\sqcup \Rightarrow 1q_1 1011\sqcup \Rightarrow 10q_0 011\sqcup \Rightarrow 101q_1 11\sqcup \Rightarrow 1010q_0 1\sqcup \Rightarrow 1010q_r 1\sqcup$

- Cadena 01010, la acepta
  $q_0 01010\sqcup \Rightarrow 1q_1 1010\sqcup \Rightarrow 10q_0 010\sqcup \Rightarrow 101q_1 10\sqcup \Rightarrow 1010q_0 0\sqcup \Rightarrow 10101q_1\sqcup \Rightarrow 1010q_a 1\sqcup$

Y $f_M(n)$ es $n+1$, esto debido a que se recorre toda la entrada de entrada y toma una transición extra para aceptar, con n la longitud de la cadena de entrada.

c) $\delta(q_0, 0) = (q_0, \sqcup, \rightarrow)$, $\delta(q_0, 1) = (q_1, \sqcup, \rightarrow)$,
$\delta(q_1, 1) = (q_1, \sqcup, \rightarrow)$, $\delta(q_1, \sqcup) = (q_a, \sqcup, \leftarrow)$.

El lenguaje L_M es $(0)^*(1)^+$, las cadenas con ninguno o varios 0s seguido de uno o varios 1s. Esto lo notamos intentando varias cadenas y viendo esto:



Además, al final deja una cadena de puros espacios vacíos (borra la cadena de entrada).

Ahora, veamos algunos ejemplos de ejecuciones:

- Cadena 00110, la rechaza
  $q_0 00110\sqcup \Rightarrow \sqcup q_0 0110\sqcup \Rightarrow \sqcup\sqcup q_0 110\sqcup \Rightarrow \sqcup\sqcup\sqcup q_1 10\sqcup \Rightarrow \sqcup\sqcup\sqcup\sqcup q_1 0\sqcup \Rightarrow \sqcup\sqcup\sqcup\sqcup q_r 0\sqcup$
- Cadena 00111, la acepta
  $q_0 00111\sqcup \Rightarrow \sqcup q_0 0111\sqcup \Rightarrow \sqcup\sqcup q_0 111\sqcup \Rightarrow \sqcup\sqcup\sqcup q_1 11\sqcup \Rightarrow \sqcup\sqcup\sqcup\sqcup q_1 1\sqcup \Rightarrow \sqcup\sqcup\sqcup\sqcup\sqcup q_1\sqcup \Rightarrow \sqcup\sqcup\sqcup\sqcup q_a\sqcup\sqcup$

Y $f_M(n)$ es $n+1$, esto debido a que se recorre toda la entrada de entrada y toma una transición extra para aceptar, con n la longitud de la cadena de entrada.

Ejercicio 3

Para cada uno de los siguientes lenguajes, da una descripción general de una máquina de Turing que decide el lenguaje sobre el alfabeto $\Sigma = a, b$ y determina el tiempo de ejecución de dicha máquina.

a) $\{w \in \Sigma^* \mid w$ contiene el triple de b's que de a's$\}$

Para este ejercicio daremos la siguiente máquina de Turing:

- **Estados:**
  La máquina de Turing llevará un conteo del número de a's y b's en la cadena de manera que pueda determinar si hay exactamente el triple de b's que de a's.
- **Algoritmo:**
  1. Recorrer la cadena buscando una a sin marcar, al encontrarla marcarla, si no se encuentra alguna, regresamos al inicio de la cadena y pasamos al paso 10.

2. Regresar al inicio de la cadena.
3. Recorrer la cadena buscando una b sin marcar, al encontrarla marcarla (sería la primera b), si no se encuentra alguna rechazamos.
4. Regresar al inicio de la cadena.
5. Recorrer la cadena buscando una b sin marcar, al encontrarla marcarla (sería la segunda b), si no se encuentra alguna rechazamos.
6. Regresar al inicio de la cadena.
7. Recorrer la cadena buscando una b sin marcar, al encontrarla marcarla (sería la tercera b), si no se encuentra alguna rechazamos.
8. Regresar al inicio de la cadena.
9. Repetir pasos 1 al 8 hasta que no se encuentren a's sin marcar.
10. Recorrer toda la cadena, si encontramos una b sin marcar, rechazar, de otro modo aceptar.

- **Aceptación:**
  La máquina acepta si puede emparejar exactamente una a con tres b's en la entrada. En caso contrario, la máquina rechaza.

- **Tiempo de ejecución:**
  Para calcular el tiempo de ejecución, el peor de los casos, la máquina tiene que recorrer toda la cinta varias veces. Digamos que la longitud de la cadena será n, entonces en cada recorrido de ida y regreso tendremos $2n$ pasos.
  Ahora, notemos que haremos $8n$ pasos para cada a (ya que buscamos una a y tres b, por lo cual hacemos 8 veces el recorrido que nos toma $2n$ pasos) y finalmente un recorrido más para ver si hay b's sin marcar. Por lo tanto, la complejidad total será $8n \cdot c + n$, en donde c es el número de a's en la cadena, por lo cual la complejidad de todo sería $O(nc)$.

b) $\{w \in \{a, b, c\}^* \mid$ el total de a's en $w \geq$ el total de b's en $w >$ el total de c's en $w\}$

Para este ejercicio daremos la siguiente máquina de Turing:

- **Estados:**
  La máquina llevará un conteo del número de a's, b's y c's. Para ello, la máquina puede marcar las letras que ya ha contado y seguir recorriendo la cadena, para no repetir letras contadas.

- **Algoritmo:**
  1. Recorrer la cadena buscando una c sin marcar, al encontrarla marcarla, si no se encuentra alguna, regresamos al inicio de la cadena y pasamos al paso 8.
  2. Regresar al inicio de la cadena.
  3. Recorrer la cadena buscando una b sin marcar, al encontrarla marcarla, si no se encuentra alguna rechazamos.
  4. Regresar al inicio de la cadena.
  5. Recorrer la cadena buscando una a sin marcar, al encontrarla marcarla, si no se encuentra alguna rechazamos.
  6. Regresar al inicio de la cadena.
  7. Repetir pasos del 1 al 6 hasta que no existan c's sin marcar.

8. Recorrer la cadena buscando una b sin marcar, al encontrarla marcarla, si no se encuentra alguna rechazamos.
9. Regresar al inicio de la cadena.
10. Recorrer la cadena buscando una a sin marcar, al encontrarla marcarla, si no se encuentra alguna rechazamos.
11. Regresar al inicio de la cadena.
12. Recorrer la cadena buscando una b sin marcar, al encontrarla marcarla, si no se encuentra alguna aceptamos.
13. Regresar al inicio de la cadena.
14. Recorrer la cadena buscando una a sin marcar, al encontrarla marcarla, si no se encuentra alguna rechazamos.
15. Regresar al inicio de la cadena.
16. Repetir pasos 12 a 15 hasta que no queden b's sin marcar, cuando ya no queden b's sin marcar aceptamos.

- **Aceptación:**
  La máquina acepta si se cumplen ambas condiciones: $a \geq b > c$.

- **Tiempo de ejecución:**
  Para calcular el tiempo de ejecución, tendremos que contar el número de veces que se recorre la cadena. En este caso, se realizan $6n \cdot d$ pasos para marcar todas las c's (pasos 1 al 7), en donde d es la cantidad de c's, luego para marcar una b y una a (pasos 8 al 11) nos toma toma $4n$ pasos y para marcar el resto de b's (pasos 12 a 16) toma $(e-d-1)4n$, en donde e es la cantidad de bs, por lo cual la complejidad de todo sería $6nd + 4nn + (e-d-1)4n = 2dn + 4en$, lo cual es $O(dn + en)$.

c) $\{w \in \Sigma^* \mid a^n b^m a^{n+2m} \text{ y}(n,m) \geq 1\}$

Para este ejercicio daremos la siguiente máquina de Turing:

- **Estados:**
  La máquina primero revisara si el formato de la cadena es valida (algo de la forma $a^+b^+a^+$), para esto se recorre la cadena completa pasando entre 3 estados dependiendo si encuentra una a o una b. Después se llevará un conteo del número de a's y b's, para esto, la máquina puede marcar las letras que ya ha contado y seguir recorriendo la cadena, para no repetir letras contadas.

- **Algoritmo:**
  1. Revisar el primer símbolo de la cadena entrada, si es una b rechazamos y si es una a seguimos.
  2. Recorrer la cadena buscando la primera b, si llegamos al final de la cadena de entrada sin encontrar una b rechazamos.
  3. Recorrer la cadena de entrada a partir de la b encontrada en el paso 2, recorrer la cadena hasta encontrar una a, si llegamos al final de la cadena de entrada sin encontrar una a rechazamos.
  4. Recorrer la cadena de entrada a partir de la a encontrada en el paso 3, recorrer la cadena hasta encontrar el final de la cadena, si encontramos alguna b durante este recorrido rechazamos.

5. Regresamos al inicio de la cadena.
6. Recorrer la cadena buscando la primera a sin marcar antes de una b, al encontrarla marcarla, si no se encuentra pasar al paso 11.
7. Recorrer la cadena de entrada a partir de la a marcada en el paso 6, recorrer la cadena hasta encontrar la primera b sin marcar.
8. Recorrer la cadena de entrada a partir de la b encontrada en el paso 7, recorrer la cadena buscando la primera a sin marcar después de la b de donde partimos, al encontrarla marcarla, si no se encuentra una a rechazamos.
9. Regresamos al inicio de la cadena.
10. Repetir pasos 6 a 9 hasta que no queden a's sin marcar antes de la primera b.
11. Recorrer la cadena buscando la primera b sin marcar, al encontrarla marcarla, si no se encuentra pasar al paso 16.
12. Recorrer la cadena de entrada a partir de la b marcada en el paso 11, recorrer la cadena hasta encontrar la primera a sin marcar después de la b marcada de donde partimos, al encontrarla marcarla, si no se encuentra una a rechazamos.
13. Recorrer la cadena de entrada a partir de la a marcada en el paso 12, recorrer la cadena hasta encontrar la primera a sin marcar después de la a marcada de donde partimos, al encontrarla marcarla, si no se encuentra una a rechazamos.
14. Regresamos al inicio de la cadena.
15. Repetir pasos 11 a 14 hasta que no queden b's sin marcar.
16. Regresamos al inicio de la cadena.
17. Recorremos toda la cadena, si encontramos alguna a o b sin marcar rechazamos, pero si llegamos al final de la cadena sin encontrar una a o b sin marcar, aceptamos.

- **Aceptación:**
  La máquina acepta si tiene el formato correcto y al final del algoritmo no existen b's o a's sin marcar.

- **Tiempo de ejecución:**
  Para calcular el tiempo de ejecución, primero veamos cual seria el tamaño de una cadena que si aceptamos, como hay n a's primero luego m b's y por ultimo $n+2m$ a's por lo tanto en total hay $2n+3m$ símbolos en la cadena, entonces recorrer la cadena de ida y regreso nos tomaría $4n+6m$ pasos. Por lo tanto para los pasos 1 a 5 nos toma $4n+6m$ pasos (solo recorremos de ida y vuelta la cadena una vez), después los pasos 6 a 9 toman $4n+6m$ pasos (recorremos una vez toda la cadena, ida y regreso), los pasos 6 a 9 se repiten hasta que no hay a's sin marcar antes de las b's por lo tanto los pasos 6 a 9 se repiten n veces (por el a^n), por lo cual los pasos 6 a 10 nos tomara en total $4n^2+6nm$, luego los pasos 11 a 14 nos toman $4n+6m$ pasos (solo recorremos de ida y vuelta la cadena una vez), los pasos 11 a 14 se repiten hasta que no hay b's sin marcar por lo tanto los pasos 11 a 14 se repiten m veces (por el b^m), por lo cual los pasos 11 a 15 nos tomara en total $4nm+6m^2$, y por ultimo los pasos 16 y 17 solo recorren la cadena una vez, primero de regreso y luego de iba, entonces estos dos pasos nos toma $4n+6m$ pasos. Juntando todo lo que nos toman los pasos tenemos que todo toma $4n+6m+4n^2+6nm+4nm+6m^2+4n+6m$, lo cual es $8n+12m+10nm+4n^2+6m^2$,

entonces seria algo de la complejidad $O(nm+n^2+m^2)$ o si queremos simplificar las cosas seria algo de la forma $O(p^2)$ con $p = max(n, m)$.

Ejercicio 4

Sea $L = \{a\#b \mid a,\ b$ codifican números enteros positivos en binario y $min(a, b) = a\}$.

Da la descripción completa de una máquina de Turing (incluyendo estados, alfabeto y función de transición) que acepte L. Ejemplifica la ejecución de la máquina propuesta mostrando las configuraciones que se producen para las siguientes entradas (indica claramente el total de pasos en cada caso):

- $\lfloor 10 \rfloor \# \lfloor 6 \rfloor \ \rightarrow 1010\#110$
- $\lfloor 2 \rfloor \# \lfloor 5 \rfloor \ \rightarrow 10\#101$
- $\lfloor 6 \rfloor \# \lfloor 6 \rfloor \ \rightarrow 110\#110$

Calcula la complejidad de tiempo, en el peor caso, en función del tamaño de la entrada.

La máquina M será $M =< Q, \Sigma, \Gamma, \delta, q_s, q_a, q_r >$, M será una máquina de Turing de cinta infinita hacia ambos lados (es equivalente a una máquina de Turing de cinta infinita a la derecha, visto en clase). Esto para poder facilitar encontrar el inicio de la cadena de entrada (tendrá un espacio vació $\sqcup$ al inicio y final de la cadena de entrada).
Tenemos que con $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}, q_{11}, q_a, q_r\}$:

- $\Sigma = \{0, 1, \#\}$
- $\Gamma = \{0, 1, \#, \sqcup, \dot{0}, \dot{1}, \ddot{0}, \ddot{1}\}$
- $q_s = q_0$
- $q_a = q_a$
- $q_r = q_r$

Por lo que con δ se da la siguiente tabla:
Nota: Suponemos que las cadenas de entrada tienen el formato correcto y los números binarios no tendrán 0s a la izquierda, pero aun así con esta δ se puede considerar al 0 cómo $\sqcup$, aunque no se recomienda (pasan cosas extrañas).

- **Descripción de cada estado:**
  - q_0: buscar un símbolo 0 o 1 del lado izquierdo de # para ponerle una marca ($\dot{0}$ o $\dot{1}$, respectivamente).
  - q_1: ir al símbolo #
  - q_2: buscar un símbolo 0 o 1 del lado derecho de # para ponerle una marca ($\dot{0}$ o $\dot{1}$, respectivamente).
  - q_3: regresar al inicio de la cadena.
  - q_4: ver si queda algo sin marcar a la derecha de #.
  - q_5: regresar al inicio de la cadena.
  - q_6: buscar un símbolo $\dot{0}$ o $\dot{1}$ del lado izquierdo de #, guardarlo y marcarlo ($\ddot{0}$ o $\ddot{1}$, respectivamente).

	0	1	$\dot{0}$	$\dot{1}$	$\ddot{0}$	$\ddot{1}$	#	$\sqcup$
q_0	$q_1, \dot{0}, \rightarrow$	$q_1, \dot{1}, \rightarrow$	$q_0, \dot{0}, \rightarrow$	$q_0, \dot{1}, \rightarrow$	q_r	q_r	$q_4, \#, \rightarrow$	q_r
q_1	$q_1, 0, \rightarrow$	$q_1, 1, \rightarrow$	q_r	q_r	q_r	q_r	$q_2, \#, \rightarrow$	q_r
q_2	$q_3, \dot{0}, \leftarrow$	$q_3, \dot{1}, \leftarrow$	$q_2, \dot{0}, \rightarrow$	$q_2, \dot{1}, \rightarrow$	q_r	q_r	q_r	q_r
q_3	$q_3, 0, \leftarrow$	$q_3, 1, \leftarrow$	$q_3, \dot{0}, \leftarrow$	$q_3, \dot{1}, \leftarrow$	q_r	q_r	$q_3, \#, \leftarrow$	$q_0, \sqcup, \rightarrow$
q_4	q_a	q_a	$q_4, \dot{0}, \rightarrow$	$q_4, \dot{1}, \rightarrow$	q_r	q_r	q_r	$q_5, \sqcup, \leftarrow$
q_5	q_r	q_r	$q_5, \dot{0}, \leftarrow$	$q_5, \dot{1}, \leftarrow$	q_r	q_r	$q_5, \#, \leftarrow$	$q_6, \sqcup, \rightarrow$
q_6	q_r	q_r	$q_7, \ddot{0}, \rightarrow$	$q_8, \ddot{1}, \rightarrow$	$q_6, \ddot{0}, \rightarrow$	$q_6, \ddot{1}, \rightarrow$	q_a	q_r
q_7	q_r	q_r	$q_7, \dot{0}, \rightarrow$	$q_7, \dot{1}, \rightarrow$	q_r	q_r	$q_9, \#, \rightarrow$	q_r
q_8	q_r	q_r	$q_8, \dot{0}, \rightarrow$	$q_8, \dot{1}, \rightarrow$	q_r	q_r	$q_{10}, \#, \rightarrow$	q_r
q_9	q_r	q_r	$q_{11}, \ddot{0}, \leftarrow$	q_a	$q_9, \ddot{0}, \rightarrow$	$q_9, \ddot{1}, \rightarrow$	q_r	q_r
q_{10}	q_r	q_r	q_r	$q_{11}, \ddot{1}, \leftarrow$	$q_{10}, \ddot{0}, \rightarrow$	$q_{10}, \ddot{1}, \rightarrow$	q_r	q_r
q_{11}	q_r	q_r	$q_{11}, \dot{0}, \leftarrow$	$q_{11}, \dot{1}, \leftarrow$	$q_{11}, \ddot{0}, \leftarrow$	$q_{11}, \ddot{1}, \leftarrow$	$q_{11}, \#, \leftarrow$	$q_6, \sqcup, \rightarrow$

- q_7: ir al símbolo # teniendo un $\dot{0}$ guardado.
- q_8: ir al símbolo # teniendo un $\dot{1}$ guardado.
- q_9: buscar un símbolo $\dot{0}$ o $\dot{1}$ a la derecha de #, marcarlo ($\ddot{0}$ o $\ddot{1}$, respectivamente) y compararlo con $\dot{0}$.
- q_{10}: buscar un símbolo $\dot{0}$ o $\dot{1}$ a la derecha de #, marcarlo ($\ddot{0}$ o $\ddot{1}$, respectivamente) y compararlo con $\dot{1}$.
- q_{11}: regresar al inicio de la cadena.

- **Ejemplos de ejecuciones:**
  - *Cadena* 1010#110*:*

$$
\begin{aligned}
& \sqcup q_0 1010\#110\sqcup \\
\Rightarrow\ & \sqcup \dot{1} q_1 010\#110\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0} q_1 10\#110\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0}\dot{1} q_1 0\#110\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0}\dot{1}\dot{0} q_1 \#110\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0}\dot{1}\dot{0}\# q_2 110\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0}\dot{1}\dot{0} q_3 \#\dot{1}10\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0}\dot{1} q_3 0\#\dot{1}10\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0} q_3 10\#\dot{1}10\sqcup \\
\Rightarrow\ & \sqcup \dot{1} q_3 \dot{0}10\#\dot{1}10\sqcup \\
\Rightarrow\ & \sqcup q_3 \dot{1}\dot{0}\dot{1}0\#\dot{1}10\sqcup \\
\Rightarrow\ & q_3 \sqcup \dot{1}\dot{0}\dot{1}0\#\dot{1}10\sqcup \\
\Rightarrow\ & \sqcup q_0 \dot{1}\dot{0}\dot{1}0\#\dot{1}10\sqcup \\
\Rightarrow\ & \sqcup \dot{1} q_0 \dot{0}\dot{1}0\#\dot{1}10\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0} q_1 10\#\dot{1}10\sqcup \\
\Rightarrow\ & \sqcup \dot{1}\dot{0}\dot{1} q_1 0\#\dot{1}10\sqcup
\end{aligned}
$$

$$
\begin{aligned}
&\Rightarrow \sqcup \dot{1}\dot{0}10q_1\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}10\#q_2\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}10\#\dot{1}q_2 10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}10\#q_3\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}10q_3\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}1q_3 0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}q_3 10\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}q_3\dot{0}10\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup q_3\dot{1}\dot{0}10\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow q_3 \sqcup \dot{1}\dot{0}10\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup q_0\dot{1}\dot{0}10\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}q_0\dot{0}10\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}q_0 10\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}q_1 0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}0q_1\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}0\#q_2\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}0\#\dot{1}q_2\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}0\#\dot{1}\dot{1}q_2 0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}0\#\dot{1}q_3\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}0\#q_3\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}0q_3\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}q_3 0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}q_3\dot{1}0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}q_3\dot{0}\dot{1}0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup q_3\dot{1}\dot{0}\dot{1}0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow q_3 \sqcup \dot{1}\dot{0}\dot{1}0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup q_0\dot{1}\dot{0}\dot{1}0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}q_0\dot{0}\dot{1}0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}q_0\dot{1}0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}q_0 0\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}\dot{0}q_1\#\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}\dot{0}\#q_2\dot{1}\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}\dot{0}\#\dot{1}q_2\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}\dot{0}\#\dot{1}\dot{1}q_2\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}\dot{0}\#\dot{1}\dot{1}\dot{0}q_2\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\dot{1}\dot{0}\#\dot{1}\dot{1}\dot{0}q_r\sqcup
\end{aligned}
$$

La rechaza, lo cual es correcto y toma 51 transiciones.

- *Cadena* 10#101*:*

$$
\begin{aligned}
&\sqcup q_0 10\#101\sqcup\\
&\Rightarrow \sqcup \dot{1}q_1 0\#101\sqcup\\
&\Rightarrow \sqcup \dot{1}0q_1\#101\sqcup\\
&\Rightarrow \sqcup \dot{1}0\#q_2 101\sqcup\\
&\Rightarrow \sqcup \dot{1}0q_3\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}q_3 0\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup q_3\dot{1}0\#\dot{1}01\sqcup\\
&\Rightarrow q_3 \sqcup \dot{1}0\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup q_0\dot{1}0\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}q_0 0\#\dot{1}01\sqcup
\end{aligned}
$$

$$
\begin{aligned}
&\Rightarrow \sqcup \dot{1}\dot{0}q_1\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\#q_2\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\#\dot{1}q_2 01\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\#q_3\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}q_3\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}q_3\dot{0}\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup q_3\dot{1}\dot{0}\#\dot{1}01\sqcup\\
&\Rightarrow q_3 \sqcup \dot{1}\dot{0}\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup q_0\dot{1}\dot{0}\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}q_0\dot{0}\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}q_0\#\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\#q_4\dot{1}01\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\#\dot{1}q_4\dot{0}1\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\#\dot{1}\dot{0}q_4 1\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{0}\#\dot{1}\dot{0}q_a 1\sqcup
\end{aligned}
$$

La acepta, lo cual es correcto y toma 24 transiciones.

- *Cadena* 110#110*:*

$$
\begin{aligned}
&\sqcup q_0 110\#110\sqcup\\
&\Rightarrow \sqcup \dot{1}q_1 10\#110\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}q_1 0\#110\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}0q_1\#110\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}0\#q_2 110\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}0q_3\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}q_3 0\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}q_3 10\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup q_3\dot{1}10\#\dot{1}10\sqcup\\
&\Rightarrow q_3 \sqcup \dot{1}10\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup q_0\dot{1}10\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}q_0 10\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}q_1 0\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}0q_1\#\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}0\#q_2\dot{1}10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}0\#\dot{1}q_2 10\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}0\#q_3\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}0q_3\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}q_3 0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}q_3\dot{1}0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup q_3\dot{1}\dot{1}0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow q_3 \sqcup \dot{1}\dot{1}0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup q_0\dot{1}\dot{1}0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}q_0\dot{1}0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}q_0 0\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}\dot{0}q_1\#\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}\dot{0}\#q_2\dot{1}\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}\dot{0}\#\dot{1}q_2\dot{1}0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}\dot{0}\#\dot{1}\dot{1}q_2 0\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}\dot{0}\#\dot{1}q_3\dot{1}\dot{0}\sqcup\\
&\Rightarrow \sqcup \dot{1}\dot{1}\dot{0}\#q_3\dot{1}\dot{1}\dot{0}\sqcup
\end{aligned}
$$

$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}q_3\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}q_3\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}q_3\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup q_3\dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow q_3 \sqcup \dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup q_0\dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}q_0\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}q_0\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}q_0\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}\#q_4\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}q_4\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}q_4\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}q_4\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}q_5\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}q_5\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}\#q_5\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}\dot{0}q_5\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}\dot{\imath}q_5\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \dot{\imath}q_5\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup q_5\dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow q_5 \sqcup \dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup q_6\dot{\imath}\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}q_8\dot{\imath}\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\dot{\imath}q_8\dot{0}\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\dot{\imath}\dot{0}q_8\#\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\dot{\imath}\dot{0}\#q_{10}\dot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\dot{\imath}\dot{0}q_{11}\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\dot{\imath}q_{11}\dot{0}\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}q_{11}\dot{\imath}\dot{0}\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup q_{11}\ddot{\imath}\dot{\imath}\dot{0}\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow q_{11} \sqcup \ddot{\imath}\dot{\imath}\dot{0}\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup q_6\ddot{\imath}\dot{\imath}\dot{0}\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}q_6\dot{\imath}\dot{0}\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}q_8\dot{0}\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\dot{0}q_8\#\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\dot{0}\#q_{10}\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\dot{0}\#\ddot{\imath}q_{10}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\dot{0}\#q_{11}\ddot{\imath}\dot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\dot{0}q_{11}\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}q_{11}\dot{0}\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}q_{11}\ddot{\imath}\dot{0}\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup q_{11}\ddot{\imath}\ddot{\imath}\dot{0}\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow q_{11} \sqcup \ddot{\imath}\ddot{\imath}\dot{0}\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup q_6\ddot{\imath}\ddot{\imath}\dot{0}\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}q_6\ddot{\imath}\dot{0}\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}q_6\dot{0}\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\ddot{0}q_7\#\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\ddot{0}\#q_9\ddot{\imath}\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\ddot{0}\#\ddot{\imath}q_9\ddot{\imath}\dot{0}\sqcup$
$\Rightarrow \sqcup \ddot{\imath}\ddot{\imath}\ddot{0}\#\ddot{\imath}\ddot{\imath}q_9\dot{0}\sqcup$

$$
\begin{aligned}
&\Rightarrow \sqcup\ddot{1}\ddot{1}\ddot{0}\#\ddot{1}q_{11}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup\ddot{1}\ddot{1}\ddot{0}\#q_{11}\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup\ddot{1}\ddot{1}\ddot{0}q_{11}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup\ddot{1}\ddot{1}q_{11}\ddot{0}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup\ddot{1}q_{11}\ddot{1}\ddot{0}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup q_{11}\ddot{1}\ddot{1}\ddot{0}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow q_{11}\sqcup\ddot{1}\ddot{1}\ddot{0}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup q_{6}\ddot{1}\ddot{1}\ddot{0}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup\ddot{1}q_{6}\ddot{1}\ddot{0}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup\ddot{1}\ddot{1}q_{6}\ddot{0}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup\ddot{1}\ddot{1}\ddot{0}q_{6}\#\ddot{1}\ddot{1}\ddot{0}\sqcup\\
&\Rightarrow \sqcup\ddot{1}\ddot{1}\ddot{0}q_{a}\#\ddot{1}\ddot{1}\ddot{0}\sqcup
\end{aligned}
$$

La acepta, lo cual es correcto y toma 92 transiciones.

Por último, la complejidad de tiempo en el peor caso es $O(n^2)$ (con n la longitud de la cadena de entrada), esto debido a que el peor caso es cuando a y b son el mismo número, como son el primer número tenemos que primero revisar que tienen la misma longitud (estados q_0 al q_5) y luego ir revisando dígito por dígito (estados q_6 a $q_1 1$), pero como son el mismo número entonces vamos a tener que revisar todos los dígitos. Entonces revisar que tengan la misma longitud nos toma $O(n^2)$, ya que para cada dígito de número a tenemos que marcarlo y marcar uno del número b, por lo tanto, tenemos que recorrer una vez de ida la cadena y una vez de regreso $O(n)$ y como se repite para dígito del número a y $a = b$ entonces se repite $\frac{n}{2} - 1$ veces, por lo cual revisar que tengan la misma longitud nos toma $O(n^2)$.

Después, tenemos que comparar el dígito en la posición i del número a con el dígito en la posición i del número b, esto lo hacemos marcando un dígito del número a y luego recorriendo la cadena para encontrar el dígito correspondiente en el número b y luego volver a regresar, de manera muy similar a como comparábamos las longitudes, entonces cada comparación de dígitos nos toma $O(n^2)$ y como comparamos cada dígito del número a (ya que es igual a b) entonces se repite $\frac{n}{2} - 1$ veces, por lo cual comparar los dígitos nos toma $O(n^2)$. Entonces procesar una cadena en la cual los dos números son iguales nos toma $O(n^2)$.

Ejercicio 5

Investiga en qué consisten los siguientes problemas:

- El décimo problema de Hilbert
- Entscheidungsproblem
- Investiga en qué consiste el Problema de Correspondencia de Post.

Para cada caso, se debe incluir:

- Descripción del problema
- Al menos un ejemplar concreto
- Importancia o relevancia del problema
- ¿Cuándo y por quién fue resuelto?
- ¿Cuál es la respuesta al problema planteado?

El décimo problema de Hilbert

1. Descripción del problema:

Para entender el décimo problema de Hilbert, primero debemos comprender las ecuaciones diofánticas. Estas ecuaciones, caracterizadas por buscar soluciones enteras, pueden variar en su forma y en el tipo de números que consideramos como posibles soluciones. Entonces, definamos una ecuación diofántica en función de las especificaciones que Hilbert utilizó en el enunciado de su problema.

Una ecuación diofántica es una ecuación de la forma:

$$D(x_1, \ldots, x_m) = 0$$

Tal que D es un polinomio con coeficientes enteros y en donde las soluciones para $x_1, \ldots, x_m$ están restringidas a los números enteros.[1]

De esta forma, Hilbert enunció su décimo problema de la siguiente forma:

Dada una ecuación diofántica con cualquier número de incógnitas y con coeficientes numéricos enteros: Dar un algoritmo con el cual pueda determinarse, en un número finito de operaciones, si la ecuación es resoluble en números enteros.

2. Ejemplar:

Consideremos la siguiente ecuación diofántica:

$$x^2 + y^2 = z^3$$

Esta ecuación plantea la pregunta de si existen enteros x, y y z que cumplan la igualdad y, de hecho, veamos que sí existen. Algunas soluciones conocidas para esta ecuación son:

$$x = 1,\ y = 0,\ z = 1 \quad \text{o} \quad x = 3,\ y = 0,\ z = 3$$

En algunos casos específicos como este, es posible encontrar soluciones, pero el problema general es si existe un método algorítmico que pueda determinar de manera sistemática si una ecuación como esta tiene soluciones.

Sin embargo, determinar si todas las ecuaciones de este tipo tienen soluciones es un problema que no puede resolverse de manera general, lo cual quedó demostrado por el décimo problema de Hilbert: no existe un algoritmo universal que pueda decidir si una ecuación diofántica tiene soluciones enteras o no.[2]

3. Importancia del problema:

El décimo problema de Hilbert planteó la posibilidad de encontrar un método único para resolver ecuaciones diofánticas, lo que impulsó el uso de diversas técnicas matemáticas y llevó a pensar en el papel de los algoritmos para resolver problemas de forma general. Este enfoque fue clave para el desarrollo de la teoría de algoritmos, al abrir la puerta para investigar qué problemas pueden automatizarse y cuáles no. Sin embargo, la demostración de que no existe un algoritmo general para resolver todas las ecuaciones diofánticas fue un hito que estableció límites claros en la computación, mostrando que algunos problemas simplemente no son computables, lo que influyó directamente en la teoría de la complejidad y sigue motivando estudios sobre casos específicos que sí pueden resolverse.

4. **¿Cuándo y por quién fue resuelto?:**

   Fue resuelto en 1970 por el matemático Yuri Matiyasevich, en colaboración con Martin Davis, Hilary Putnam y Julia Robinson.[3]

5. **Respuesta al problema planteado:**

   Se demostró que no hay un algoritmo general para decidir la existencia de soluciones enteras a ecuaciones diofánticas, utilizando herramientas de lógica matemática y teoría de números para establecer la indecidibilidad de este tipo de problemas.

Entscheidungsproblem

1. **Descripción del problema:**

   El Entscheidungsproblem, planteado por David Hilbert, trata de encontrar un método universal que permita saber si cualquier proposición matemática es verdadera o falsa dentro de un sistema formal. Estas ideas influyeron profundamente en las matemáticas y las ciencias de la computación. La conclusión de que no se puede formalizar completamente las matemáticas impulsó el estudio de los límites de lo que se puede computar, sentando las bases teóricas para los algoritmos. El concepto de indecidibilidad sigue siendo clave para entender qué problemas pueden resolverse y cuáles no dentro de un contexto computacional.[4]

2. **Ejemplar:**

   Para mostrar un ejemplar que aborde el Entscheidungsproblem, podemos tomar la pregunta de si un conjunto dado es finito o infinito. Consideremos al siguiente conjunto de los números naturales para nuestro ejemplo:

   El conjunto de todos los números naturales que pueden generarse sumando múltiplos de 3 y múltiplos de 5.

   El Entscheidungsproblem preguntaría si existe un algoritmo que, dado cualquier conjunto de números naturales descrito de esta forma, siempre podemos determinar si el conjunto es finito o infinito. En este caso en particular notemos que es infinito, pero para otros conjuntos más complicados, no hay una manera universal de saber si son finitos o infinitos sin analizarlos en detalle.

3. **Importancia del problema:**

   David Hilbert planteó el Entscheidungsproblem y los trabajos de Gödel y Turing demostraron que existen límites fundamentales en lo que se puede conocer o probar matemáticamente. Gödel, a través de sus teoremas de incompletitud, reveló que en sistemas matemáticos complejos siempre habrá proposiciones que no se pueden probar ni refutar, lo que cuestiona la idea de que las matemáticas son completamente formalizables.

   Por otro lado, Alan Turing, con su máquina de Turing y el problema de la detención, no solo resolvió el Entscheidungsproblem, sino que también estableció las bases teóricas de la ciencia computacional. La conclusión de que no existe un algoritmo universal para resolver todos los problemas matemáticos complejos llevó al desarrollo de la teoría de la computabilidad, que clasifica los problemas en decidibles e indecidibles. Esto permitió un análisis más profundo sobre las capacidades y límites de los algoritmos y las máquinas, sentando las bases del campo moderno de la computación.[5]

4. ¿Cuándo y por quién fue resuelto?:

Fue resuelto en la década de 1930 por Alan Turing y Alonzo Church. En 1936, ambos demostraron de manera independiente que no existe un algoritmo general que pueda decidir la verdad o falsedad de todas las proposiciones matemáticas. Turing lo hizo a través de su concepto de la máquina de Turing, mientras que Church utilizó su sistema de cálculo lambda.

5. Respuesta al problema planteado:

La solución al Entscheidungsproblem fue que no hay un algoritmo universal capaz de determinar la verdad o falsedad de todas las proposiciones matemáticas en un sistema formal. Alan Turing y Alonzo Church demostraron que existen problemas matemáticos indecidibles, es decir, que no pueden resolverse mediante ningún método automático o algoritmo. Esto implica que, en términos generales, hay proposiciones que no pueden ser ni demostradas ni refutadas en ciertos sistemas formales, lo que establece límites esenciales sobre lo que se puede conocer y probar en matemáticas.

El problema de Correspondencia de Post

1. Descripción del problema:

Fue planteado por Emil Post en 1946. El Problema de Correspondencia de Post (PCP) es un problema de decisión indecidible conocido. El problema consiste en una colección de piezas de dominó, cada pieza con una cadena en cada lado (arriba y abajo), la siguiente forma: $\{\left[\frac{t_1}{b_1}\right], \left[\frac{t_2}{b_2}\right], ..., \left[\frac{t_k}{b_k}\right]\}$

El objetivo es determinar si existe un "match", el cual es una lista de estas piezas de dominó $i_1, i_2, ..., i_l$ (permitiendo repeticiones) de tal forma que el texto formado por las cadenas en la parte de arriba de las piezas es igual al texto formado por las cadenas en la parte de abajo de las piezas $t_{i_1}t_{i_2}...t_{i_l} = b_{i_1}b_{i_2}...b_{i_l}$.

2. Ejemplar:

Tenemos las siguientes piezas de dominó:

$$\{\left[\frac{a}{ab}\right], \left[\frac{abc}{c}\right], \left[\frac{b}{ca}\right], \left[\frac{ca}{a}\right]\}$$

La pregunta es saber si existe un *match*, y si existe, es el siguiente:

$$\left[\frac{a}{ab}\right]\left[\frac{b}{ca}\right]\left[\frac{ca}{a}\right]\left[\frac{a}{ab}\right]\left[\frac{abc}{c}\right], \text{ ya que } abcaaabc = abcaaabc$$

Otro ejemplo es con las siguientes piezas:

$$\{\left[\frac{abc}{ab}\right], \left[\frac{cb}{c}\right], \left[\frac{abb}{ba}\right]\}$$

La pregunta es saber si existe un "match", y no existe, ya que para cada pieza se cumple que la longitud de la cadena de arriba es mayor a la longitud de la cadena de abajo.

3. Importancia del problema:

Su principal importancia es que es un problema indecidible no tan complicado de entender, por lo cual puede ser muy útil para demostrar que otros problemas no son decididles usando métodos de reducción y otra cosa por la cual es importante es saber un poco más de los límites de la computación debido a que este problema es indecidible.

4. **¿Cuándo y por quién fue resuelto?:**

   Emil Post demostró que este problema es indecidible en un escrito llamado "A variant of a recursively unsolvable problem" que fue publicado en 1946.

5. **Respuesta al problema planteado:**

   Se demostró que el problema es indecidible, Emil en su escrito lo demuestra (lo intente leer para ver si lo entendía, pero sí está muy complejo y técnico). Una forma que se usa para demostrar que es indecidible es reducir el problema A_{TM} ($A_{TM} = \{< M, w > | M$ es una máquina de Turing que acepta la cadena $w\}$) al problema $MPCP$ (es una modificación del problema PCP, $MPCP = \{< P' > | P'$ es una colección de piezas de dominó que tiene un "match.$^{\text{em}}$pezando en la primera pieza $\left[\frac{t_1}{b_1}\right]\}$) y luego reducir el problema $MPCP$ al problema PCP ($PCP = \{< P > | P$ es el conjunto de piezas de dominó que tiene un "match"$\}$)

Referencias

[1] Hilbert's tenth problem. (s.f.). http://www.cs.ecu.edu/karl/6420/spr16/Notes/Reduction/hilbert10.html

[2] Ray, A. (2022). REMARKS ON HILBERT'S TENTH PROBLEM AND THE IWASAWA THEORY OF ELLIPTIC CURVES. Bulletin Of The Australian Mathematical Society, 107(3), 440-450. https://doi.org/10.1017/s000497272200082x

[3] The MIT Press, Massachusetts Institute of Technology. (2024, 18 junio). Book details - MIT Press. MIT Press. https://mitpress.mit.edu/9780262132954/hilberts-10th-problem

[4] Martignon, L. (2001). Algorithms. En Elsevier eBooks (pp. 382-385). https://doi.org/10.1016/b0-08-043076-7/00549-0

[5] The Rise and Fall of the Entscheidungsproblem (Stanford Encyclopedia of Philosophy). (s.f.). https://plato.stanford.edu/entries/church-turing/decision-problem.html

[6] Sipser, M. (2005). Introduction to the theory of competition. Brooks Cole.

[7] Yeon, S., Li, A. (2021). The Post Correspondence Problem. https://math.mit.edu/research/highschool/primes/circle/documents/2021/Chun_Li.pdf

[8] Post, E. L. (1946). A variant of a recursively unsolvable problem. Bulletin of the American Mathematical Society, 52(4), 264–268. https://doi.org/10.1090/s0002-9904-1946-08555-9