Autómatas y lenguajes formales. Tarea 3

Fabián Romero Jiménez

Problema 1 Diseña una máquina de Turing que reconozca el lenguaje $\{\alpha\alpha\alpha | \alpha \in \{a,b\}^\star\}$

	$\vdash$	a	b	a_1	b_1	a_2	b_2	$\dashv$
s	$s, \vdash, \rightarrow$	$s_1, a_1, \rightarrow$	$s_1, b_1, \rightarrow$	$s, a_1, \rightarrow$	$s, b_1, \rightarrow$			Acepta
s_1		$s_1, a, \rightarrow$	$s_1, b, \rightarrow$	$s_1, a_1, \rightarrow$	$s_1, b_1, \rightarrow$	$s_2, a_2, \leftarrow$	$s_2, b_2, \leftarrow$	$s_2, \dashv, \leftarrow$
s_2		$s_3, a_2, \leftarrow$	$s_3, b_2, \leftarrow$					
s_3		$s_4, a_2, \leftarrow$	$s_4, b_2, \leftarrow$					
s_4		$s_5, a, \leftarrow$	$s_5, b, \leftarrow$	$v_1, a_1, \leftarrow$	$v_1, b_1, \leftarrow$			
s_5		$s_5, a, \leftarrow$	$s_5, b, \leftarrow$	$s, a_1, \rightarrow$	$s, b_1, \rightarrow$			
v_1	$v_2, \_, \rightarrow$			$v_1, a_1, \leftarrow$	$v_1, b_1, \leftarrow$			
v_2				$v_{a1}, \vdash, \rightarrow$	$v_{b1}, \vdash, \rightarrow$			
v_a				$v_a, a_1, \rightarrow$	$v_a, b_1, \rightarrow$	$v_1, a_1, \leftarrow$		Acepta
v_b				$v_{a1}, a_1, \rightarrow$	$v_{a1}, b_1, \rightarrow$		$v_1, b_1, \leftarrow$	Acepta

Explicación Se marca un carácter al inicio (le ponemos un subindice 1) y dos al final (con subindice 2) y se repite el proceso hasta que no haya carácteres sin marcar. Si no acaba exactamente la máquina se detiene en no aceptación, si acaba exactamente la cadena es de longitud $3k, k \in \mathbb{N}$ y hay k carácteres al principio marcados (1) y $2k$ carácteres marcados al final (2). Los estados que hacen la operación de marcar son los estados s_i.

Una vez marcados todos borramos el primer carácter en la cadena y vamos a un estado donde buscamos el primer carácter con subindice 2, si es el mismo carácter base que el borrado, cambiamos de subindice (2) a subindice (1) y regresamos al principio de la cadena y repetimos, así al final de $2k$ operaciones si se eliminaron todas los subindices 2 y aceptamos la cadena.

Problema 2 Diseña una máquina de Turing que acepte el conjunto $a^{2^m}, m \in \mathbb{Z}^+$.

	$\vdash$	a	a_1	a_2	$\dashv$
s	$s, \vdash, \rightarrow$	$s_1, a, \rightarrow$			
s_1		$s_2, a, \leftarrow$		Acepta	Acepta
s_2		$s_3, a_1, \rightarrow$			
s_3		$s_3, a, \rightarrow$		$s_4, a_2, \leftarrow$	$s_4, \dashv, \leftarrow$
s_4		$s_5, a_2, \leftarrow$			
s_5		$s_6, a, \leftarrow$	$s_7, a, \leftarrow$		
s_6		$s_6, a, \leftarrow$	$s_2, a_1, \rightarrow$		
s_7	$s, \vdash, \rightarrow$		$s_7, a, \leftarrow$		

Explicación Se verifica si la cadena no marcada es de longitud 1, en este caso se acepta (Estados s, s_1), luego se marca el primer carácter no marcado con subindice 1 y el ultimo carácter no marcado con subindice 2 y se repite el proceso hasta que no hay más carácteres por marcar, partiendo el conjunto inicial de carácteres no marcados en carácteres marcados con la primera mitad marcada con subindice 1 y la segunda marcada con subindice 2, luego se borran las marcas a todos los elementos con subindice 1 y ser repite el procedimiento hasta que termine de verificar la cadena.

Problema 3 Demuestra que el conjunto $TOT = \{M|M \text{ se detiene con todas las entradas}\}$ no es recursivamente enumerable y tampoco lo es su complemento.

Demostración Por diagonalización:

Supongamos que si, que $TOT = \{M|M \text{ se detiene con todas las entradas}\}$ es recursivamente enumerable, por lo que hay una máquina de Turing M_{TOT} que puede generar todas las máquinas en TOT, creemos una matriz cuadrada con M_i la i-ésima máquina total generada por M_{TOT} poniendo en (i, j) si la máquina M_i acepta o rechaza M_j, esto es posible por que M_i es total.

	M_1	M_2	M_3	...
M_1	[0]	1	0	...
M_2	1	[1]	1	...
M_3	0	1	[1]	...
...	...	...	...	[...]

Por lo tanto hay una maquina de turing TM_{TOT} que acepta el lenguage TOT y podemos crear una maquina $TM_{NO} = \{M|M \in TOT \wedge M$ no acepta $M\}$ simplemente poniendo la maquina que evalua a una maquina con ella misma como entrada después de pasarla por TOT, como TOT regresa solo maquinas totales cuando evaluamos M en M termina eventualmente por lo que TM_{NO} es entonces recursivente enumerable. ahora TM_{NO} se detiene con TM_{NO}?, si esto pasa entonces por definición no debería deternese y si se detiene tendría que no deterse. Contradiccción a la hipotesis de que sea recursivamente enumerable.

Problema 4 Demuestra la siguiente extensión del teorema de Rice: toda propiedad no trivial de pares de conjuntos recursivamente enumerables es indecidible. Utilízalo para demostrar que, dadas dos máquinas $N, M \in MT$, los siguientes problemas son indecidibles:

Extensión Podemos construir una maquina de turing que simule un paso la primera propiedad y un paso la segunda, así, la propiedad no trivial de pares es una propiedad no trivial de la maquina que hace la composición, y se aplica el teorema de Rice en su forma normal.

Aplicaciones

(a) $L(M) = L(N)$ La máquina composicion: acepta x si ambos M y N aceptan x, nos da la equivalencia con saber si acepta un lenguage no vacío que es indecidible.

(b) $L(M) \cap L(N) = \emptyset$ Exactamente la misma maquina, pero ahora rechaza si ambos aceptan, tambien se reduce al problema del lenguage vacío.

(c) $L(M) \subseteq L(N)$ Misma máquina, condicionando la evaluacion de N si M ya acepto.

Problema 5 Ubica a TOT en la jerarquía aritmética por medio de un predicado.

Respuesta considerando la descripcion de TOT como
$TOT = \{M | \forall x \exists t. M$ se detiene en la entrada x en n pasos $\}$
Sabemos que TOT esta en Π_2^0