UNIVERSIDAD SIMON BOLIVAR **2 de febrero de 2010**
DEPARTAMENTO DE COMPUTACION
MAESTRIA EN CIENCIAS DE LA COMPUTACION **Ing. Angarita Arocha Rafael**
ASIGNATURA: CI-7541 - Teoria de la computación. **Lic. Claro Mosqueda Alejandro**

TAREA N^o 1

Ejercicio 3.6: En el teorema 3.21 [1, p. 153] se mostró que un lenguaje es Turing-reconocible sii algún enumerador lo enumera. Por qué no se utilizó el siguiente algoritmo para la demostración directa? Como antes, $s_1, s_2, s_3, \ldots$ es una lista de todas las posibles palabras en Σ^*.

$E =$ "Ignora la entrada.

1. Repite lo siguiente para $i = 1, 2, 3, \ldots$
2. Ejecuta M con s_i como entrada.
3. Si M acepta, imprime como salida s_i."

Solución:

Con este algoritmo es posible que E no enumere el lenguaje $L(M)$. Esto se debe a que M no es un "decider", y puede que no acepte ni rechace alguna palabra s_i en el paso 2. Si esto sucede, E no imprimirá mas palabras pertenecientes a $L(M)$ despues de s_i.

Problema 3.12: Una ***máquina de Turing con reinicio a la izquierda*** (LRTM) es similar a una máquina de Turing ordinaria (TM), pero la función de transición tiene la forma:

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{R, RESET\} \tag{1}$$

Si $\delta(q, a) = (r, b, RESET)$, cuando la máquina esta en el estado q leyendo el símbolo a, la cabeza de la máquina salta al extremo izquierdo de la cinta después de escribir el símbolo b sobre la cinta y entrar en el estado r. Observe que esta máquina no tiene la habilidad usual de mover la cabeza un símbolo a la izquierda. Muestre que la máquina de Turing con reinicio a la izquierda reconoce la clase de lenguajes Turing-reconocibles.

Solución:

Para demostrar que LRTM reconoce la clase de lenguajes Turing-reconocibles es suficiente demostrar que ésta es capaz de simular una máquina de Turing ordinaria. El funcionamiento de estas dos máquinas se diferencia únicamente en la movilidad de la cabeza de la cinta. Debido a esto, el único punto que vale la pena tratar en la demostración es como LRTM es capaz de alcanzar configuraciones equivalentes a las que producen las funciones de transición de TM durante cada paso de la simulación.

Demostración. Considérese alguna función de transición de TM de la forma $\delta(q_l, x) = (q_m, y, R)$. La simulación de esta transición en LRTM es trivial,

$$\delta_{LRTM}(q_l, x) = (q_m, y, R). \tag{2}$$

Esto es, cuando TM mueve el cabezal a la derecha entonces LRTM hace lo mismo.

La otra posibilidad, $\delta(q_i, a) = (q_j, b, L)$, no es tan sencilla. La idea para simular esta transición consiste en reemplazar el símbolo a por b con una marca sobre este símbolo (llamemos este símbolo, sin pérdida de generalidad, *). Luego, se reinicia ($RESET$) y se procede a buscar de izquierda a derecha el símbolo inmediatamente antes del marcado con *. Esto se puede lograr, marcando (con otro símbolo, digamos $^\#$) y reiniciado ($RESET$) de forma sucesiva hasta llegar de nuevo al símbolo marcado con *. A continuación se describe con mas detalle, paso a paso, el procedimiento que simula el movimiento del cabezal una celda a la izquierda:

1. Marcar con * la celda actual y reinicia a la izquierda el cabezal ($RESET$).
2. Marcar con $^\#$ la primera celda de la cinta y mover una celda a la derecha (R).
3. Si la celda actual esta marcada con * salta al paso 9.
4. Marcar con $^\#$ la celda actual y reinicia a la izquierda el cabezal ($RESET$).
5. Recorrer la cinta con pasos a la derecha hasta encontrar la primera celda marcada con $^\#$.
6. Remover la marca $^\#$ de la celda actual y mover una celda a la derecha el cabezal (R).
7. Mover una celda a la derecha el cabezal (R).
8. Saltar al paso 3.
9. Remover la marca * de la celda actual y reinicia a la izquierda el cabezal ($RESET$).
10. Recorre la cinta con pasos a la derecha hasta encontrar la celda marcada con $^\#$. Esta celda es la celda junto a la izquierda de la celda inicial.

Observe que la marca $^\#$ desaparece en la proxima transición, sea por qué se sustituye por un símbolo sin marca en el movimiento a la derecha o por un símbolo marcado con * al comienzo del movimiento a la izquierda. Con esto se concluye que es posible simular todas las transiciones de cualquier TM en LRTM y por lo tanto LRTM también reconoce la clase de lenguajes Turing-reconocibles.

□

Problema 3.18: Muestre que un lenguaje es decidible sii algún enumerador enumera el lenguaje en orden lexicográfico.

Solución:

Demostración. A continuación se describe una máquina de Turing que decide un lenguaje A si un enumerador E enumera A en orden lexicográfico. Por conveniencia y sin pérdida de generalidad, en caso de ser A finito, el enumerador E, al terminar de imprimir todas las palabras en A repite desde el comienzo. [1]

[1] La idea detrás de esta demostración esta basada en la demostración del Teorema 3.21 [1, p. 153].

$M =$ "En la entrada w:

1. Ejecuta E.
2. Cada vez que E imprime una palabra x_i, se realizan las siguientes acciones,
3. Si x_i es lexicográficamente menor que x_{i-1}, para y ***rechaza***.
4. Se compara x_i con w.
5. Si x_i es lexicográficamente mayor que w, para y ***rechaza***.
6. Si x_i es igual a w, para y ***acepta***.

Claramente M acepta aquellas cadenas que aparecen impresas por E. Además, el hecho de que las palabras aparezcan en orden lexicográfico permite que se garantice que la máquina M termina, sea porque se imprimen todas las palabras en A, porque se alcanza una palabra lexicográficamente mayor que la entrada o porque se acepta una palabra impresa.

Ahora, en la otra dirección. Si M decide A, se puede construir el siguiente enumerador E para A. Sea $s_1, s_2, s_3, \ldots$ una lista de todas las posibles palabras en Σ^* en orden lexicográfico, entonces,

$E =$ "Ignora la entrada.

1. Repite lo siguiente para $i = 1, 2, 3, \ldots$
2. Ejecuta M con s_i como entrada.
3. Si M acepta, imprime como salida s_i."

como M es un "decider", eventualmente E imprimirá todas las palabras en A. □

Ejercicio 4.6: Sea $\mathcal{B}$ el conjunto de todas las secuencias infinitas sobre $\{0, 1\}$. Muestre que $\mathcal{B}$ es incontable, utilizando una demostración por diagonalización.

Solución:

Demostración. Suponga que $\mathcal{B}$ es contable. Esto es, existe una correspondencia entre $\mathbb{N}$ y $\mathcal{B}$ dada por cierto mapa $f : \mathbb{N} \longrightarrow \mathcal{B}$.

n	$f(n)$
0	000000...
1	100000...
2	010000...
⋮	⋮
i	$f(i)$
⋮	⋮

Cuadro 1: Correspondencia entre $\mathbb{N}$ y $\mathcal{B}$.

Si se considera una palabra, $z = z_1 z_2 z_3 \ldots$, perteneciente a $\mathcal{B}$ que cumple la siguiente condición,

$$z_i = \begin{cases} 0 & \text{si } [f(i)]_i = 1 \\ 1 & \text{si } [f(i)]_i = 0 \end{cases} \tag{3}$$

donde z_i es el i-esimo símbolo en la palabra z y $[f(i)]_i$ es el i-esimo símbolo en la palabra $f(i)$.

La palabra z es la palabra que difiere de cada $f(i)$ en su i-esimo símbolo. Esto indica que z no puede ser igual a ningún $f(i)$, y por lo tanto z esta fuera de la imagen de f, lo cual contradice la suposición inicial. En conclusión, queda demostrado por contradicción que $\mathcal{B}$ es incontable.

□

Problema 4.12: Sea $A = \{\langle R, S\rangle \mid R \text{ y } S \text{ son expresiones regulares y } L(R) \subseteq L(S)\}$. Muestre que A es decidible.

Solución:

Demostración. A continuación se describe una máquina de Turing M que decide el lenguaje A:

$M =$ "En la entrada $\langle R, S\rangle$ donde R y S son expresiones regulares:

1. Convertir R y S a su forma equivalente NFA R_{NFA} y S_{NFA}. [1, Teorema 1.54 p. 66]
2. Convertir R_{NFA} y S_{NFA} a su forma equivalente DFA R_{DFA} y S_{DFA}. [1, Teorema 1.39 p. 55]
3. Construir un DFA E tal que $L(E) = L(S_{DFA}) \cap L(R_{DFA})$.
4. Ejecutar la TM F con $\langle E, R_{DFA}\rangle$ como entrada. Donde, F decide EQ_{DFA}. [1, p. 169]
5. Si F acepta, para y ***acepta***.
6. Si F rechaza, para y ***rechaza***."

□

Problema 4.18: Sean A y B dos lenguajes disjuntos. Se dice que un lenguaje C ***separa*** a A de B si $A \subseteq C$ y $B \subseteq \overline{C}$. Muestre que cualquier par de lenguajes co-Turing-reconocibles son separables por algún lenguaje decidible.

Solución:

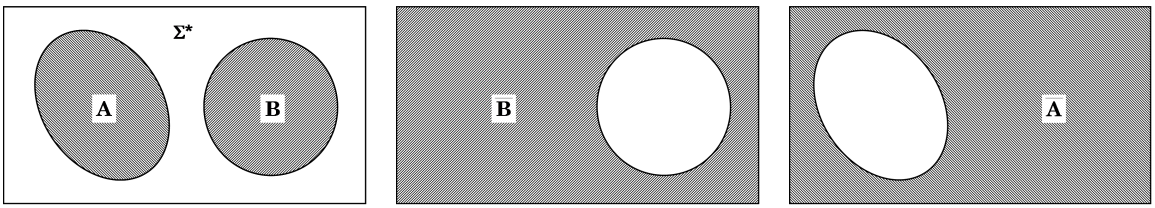
Demostración. Considérese la siguiente descripción de una máquina de Turing M:

$M =$ "En la entrada w:

1. Ejecutar, en paralelo, $M_{\overline{A}}$ y $M_{\overline{B}}$ con w como entrada en cada máquina.
2. Si $M_{\overline{B}}$ acepta primero, para y ***acepta***. Si $M_{\overline{B}}$ rechaza, continúa con la ejecución de $M_{\overline{A}}$.
3. Si $M_{\overline{A}}$ acepta primero, para y ***rechaza***. Si $M_{\overline{A}}$ rechaza, continúa con la ejecución de $M_{\overline{B}}$."

donde $M_{\overline{A}}$ y $M_{\overline{B}}$ son máquinas de Turing que reconocen $\overline{A}$ y $\overline{B}$ respectivamente. Ejecutar las dos máquinas en paralelo significa que M tiene dos cintas, una para simular $M_{\overline{A}}$ y la otra para simular $M_{\overline{B}}$. En este caso M toma turnos simulando un paso de cada máquina, lo cual continúa hasta que una de ellas acepta. [2]

[2] Idea tomada de la demostración del teorema 4.22 [1, p. 181].



(a) Σ^*, A y B (b) $\overline{B}$ (c) $\overline{A}$

Figura 1: Imagenes ilustrativas del los conjutos Σ^*, A, B, $\overline{A}$ y $\overline{B}$

Claramente el conjunto de palabras que $M_{\overline{A}}$ no acepta (A) es aceptado por $M_{\overline{B}}$ y viceversa, el conjunto de palabras que $M_{\overline{B}}$ no acepta (B) es aceptado por $M_{\overline{A}}$, ya que los conjuntos A y B son disjuntos ($A \cap B = \emptyset$) [Figura 1]. Esto implica que M decide $C \equiv L(M)$ puesto que cualquier palabra $w \in \Sigma^*$ es aceptado por $M_{\overline{A}}$ o por $M_{\overline{B}}$. Por otro lado, el hecho de que M acepta cuando $M_{\overline{B}}$ acepta primero (rechaza de lo contrario) implica que $C \subseteq \overline{B}$. Además, si $w \in A$ la máquina $M_{\overline{A}}$ no acepta pero $M_{\overline{B}}$ si lo hace. Esto significa que $A \subseteq C$. En conclusión, el lenguaje C decidido por M separa a A de B. [2]

□

Problema 4.28: Sea A un lenguaje Turing-reconocible que consiste en la descripción de máquinas de Turing, $\{\langle M_1 \rangle, \langle M_2 \rangle, \dots\}$, donde cada M_i es un "decider". Demuestre que algún lenguaje decidible D no es decidido por algún "decider" M_i cuya descripción aparece en A. (Ayuda: quizás encuentre útil considerar un enumerador para A.)

Solución:

Demostración. Considere el siguiente mapa f entre $\mathbb{N}^+$ y A:

n	$f(n)$
1	$\langle M_1 \rangle$
2	$\langle M_2 \rangle$
3	$\langle M_3 \rangle$
$\vdots$	$\vdots$
i	$\langle M_i \rangle$
$\vdots$	$\vdots$

Cuadro 2: Mapa f entre $\mathbb{N}^+$ y A.

Sea E un enumerador que enumera A en orden según f y sea w_i la i-esima palabra de $\Sigma^* \equiv \{w_1, w_2, w_3, \dots\}$ ordenado de alguna manera arbitraria, entonces TM M_D,

$M_D =$ "En la entrada w_i:

1. Ejecutar E.
2. Esperar hasta que E imprima la i-esima palabra $\langle M_i \rangle$.
3. Simular TM M_i con w_i como entrada.
4. Si M_i acepta, para y ***rechaza***.
5. Sino, para y ***acepta***.",

decide cierto lenguaje $D \equiv L(M_D)$ ya que M_i es un "decider". Si D fuese decidido por alguna máquina M_k cuya descripción pertenece a A, esto es $D = L(M_k)$, entonces tanto M_D como M_k deberían aceptar (o rechazar) ante la entrada w_k. Sin embargo, por construcción de M_D, se obtiene una contradicción, cuando M_k acepta w_k entonces M_D rechaza y viceversa. De este argumento se concluye que D es un lenguaje decidible que no puede ser decidido por alguna máquina cuya descripción aparece en A.

□

Referencias

[1] Sipser, Michael. *Introduction to the theory of computation.* International Thomson Publishing, 2nd Edition, 2006.

[2] Conversación con el Profesor Blai Bonet.