Tarea 4

Profesor: Jesús Rodríguez Viorato

IA & TC

Agosto-Diciembre 2021

Roberto Vásquez Martínez

Problema 1

Este ejercicio considera la TM M_1, cuya descripción y diagrama de estado aparece en el *Ejemplo 3.9*. En cada uno de los incisos, da la secuencia de configuraciones por las que pasa M_1 cuando empieza en las siguientes cadenas de entrada.

a) 11.

b) 1#1.

c) 1##1.

d) 10#11.

e) 10#10.

Solución:

a) La sucesión de configuraciones con entrada 11 en la TM del *Ejemplo 3.9* es la siguiente:

$$q_1 11 \quad \mathbf{x} q_3 1 \quad \mathbf{x}1 q_{3\sqcup} \quad \mathbf{x}1 \sqcup q_{\text{reject}}.$$

b) Para el input 1#1 la sucesión de configuraciones en orden de izquierda a derecha en las filas y luego de arriba a abajo entre filas es como sigue

$$\begin{array}{lll} q_1 1\#1 & \mathbf{x} q_3 \#1 & \mathbf{x}\#q_5 1 \\ \mathbf{x} q_6 \#\mathbf{x} & q_7 \mathbf{x}\#\mathbf{x} & \mathbf{x} q_1 \#\mathbf{x} \\ \mathbf{x}\#q_8 \mathbf{x} & \mathbf{x}\#\mathbf{x} q_{8\sqcup} & \mathbf{x}\#\mathbf{x} \sqcup q_{\text{accept}} \end{array}$$

c) De manera similar al inciso anterior, presentamos la sucesión de configuraciones correspondiente al input 1##1 recorriendo primero las filas de izquierda a derecha y luego de arriba a abajo entre filas. La sucesión de configuraciones en este caso es

$$q_1 1\#\#1 \quad \mathbf{x} q_3 \#\#1 \quad \mathbf{x}\#q_5 \#1 \quad \mathbf{x}\#\#q_{\text{reject}} 1$$

d) Ahora presentamos la sucesión de configuraciones correspondiente a la cadena de entrada 10#11 en el orden que hemos estado trabajando en los incisos anteriores.

La sucesión de configuraciones es la siguiente

$$\begin{array}{llll} q_1 10\#11 & \mathbf{x} q_3 0\#11 & \mathbf{x}0 q_3 \#11 & \mathbf{x}0\#q_5 11 \\ \mathbf{x}0 q_6 \#\mathbf{x}1 & \mathbf{x} q_7 0\#\mathbf{x}1 & q_7 \mathbf{x}0\#\mathbf{x}1 & \mathbf{x} q_1 0\#\mathbf{x}1 \\ \mathbf{xx} q_2 \mathbf{x}1 & \mathbf{xx}\#q_4 \mathbf{x}1 & \mathbf{xx}\#\mathbf{x} q_4 1 & \mathbf{xx}\#\mathbf{x}1 q_{\text{reject}} \end{array}$$

e) Finalmente, presentamos a continuación la sucesión de configuraciones correspondientes a la cadena de entrada 10#10.

$$
\begin{array}{lll}
q_1 10\#10 & \mathbf{x}q_3 0\#10 & \mathbf{x}0q_3\#10 \\
\mathbf{x}0\#q_5 10 & \mathbf{x}0q_6\#\mathbf{x}0 & \mathbf{x}q_7 0\#\mathbf{x}0 \\
q_7\mathbf{x}0\#\mathbf{x}0 & \mathbf{x}q_1 0\#\mathbf{x}0 & \mathbf{xx}q_2\#\mathbf{x}0 \\
\mathbf{xx}\#q_4\mathbf{x}0 & \mathbf{xx}\#\mathbf{x}q_4 0 & \mathbf{xx}\#q_6\mathbf{xx} \\
\mathbf{xx}q_6\#\mathbf{xx} & \mathbf{x}q_7\mathbf{x}\#\mathbf{xx} & \mathbf{xx}q_1\#\mathbf{xx} \\
\mathbf{xx}\#q_8\mathbf{xx} & \mathbf{xx}\#\mathbf{x}q_8\mathbf{x} & \mathbf{xx}\#\mathbf{xx}q_{8\sqcup} \\
\multicolumn{3}{c}{\mathbf{xx}\#\mathbf{xx}\sqcup q_{\text{accept}}}
\end{array}
$$

□

Problema 2

Da una definición formal de *enumerador*. Considera que es un tipo máquina de Turing de dos cintas que usa la segunda como impresora. Incluye una definición de *lenguaje enumerado*

Solución: En primer lugar, daremos la definición de enumerador que será la siguiente.

Definición 1

Un enumerador E es una 7-tupla $E = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{print}}, q_{\text{reject}})$ que representa a una máquina de Turing de 2-capas, donde Q, Σ, Γ son conjuntos finitos tales que

(i) Q es el conjunto de estados.

(ii) Γ es el alfabeto de trabajo.

(iii) Σ es el alfabeto que se imprimirá en la segunda cinta.

(iv) $\delta : Q \times \Gamma^2 \to Q \times \Gamma^2 \times \{L, R, S\}^2$ la función de transición.

(v) $q_0 \in Q$ es el estado inicial.

(vi) $q_{\text{print}} \in Q$ es el estado aceptor en en el cual se acepta la cadena escrita en la primer cinta.

(vii) $q_{\text{reject}} \in Q$ es el estado de rechazo, donde $q_{\text{reject}} \neq q_{\text{print}}$.

El cómputo en el *enumerador* es como sigue.

1. La máquina tiene dos cintas, la primer es la de trabajo y la segunda es la de impresión. Ambas están inicialmente en blanco, es decir, el enumerador empieza con una entrada vacía.

2. En cada paso, la máquina escribirá un símbolo del alfabeto Σ en la segunda cinta, que es la de impresión, o bien no escribirá nada, todo determinado por la función de transición δ. Por ejemplo, la función de transición δ tiene la forma

$$\delta(q_i, a, \sqcup) = \begin{cases} (q_{i+1}, b, c, L, R) & \text{si } c \in \Sigma \\ (q_{i+1}, b, \sqcup, L, S) & \text{en otro caso} , \end{cases}$$

   esto quiere decir que en el estado q_i al leer a el enumerador E llega al estado q_{i+1}, escribe b en lugar de a y desplaza la cabeza de la primera cinta a la izquierda según L.

   En la segunda cita, se imprime c y desplaza la cabeza a la derecha si c está en el alfabeto Σ, en otro caso se queda donde estaba. Cabe mencionar que la cabeza en la segunda cinta siempre apunta al final de la cadena impresa en la segunda cinta, por lo que en cada transición la cabeza de la segunda cinta apunta al símbolo en blanco $\sqcup$ al final de la cadena y sólo se puede desplazar a la derecha o quedarse ahí dependiendo de si se imprimirá algo en la segunda cinta o no.

3. Cuando se entra al estado q_{print} entonces lo segunda cinta se reinicia dejándola en blanco y la cabeza de la cinta de trabajo se pone al inicio.

   Por lo que después de entrar al estado q_{print} se vuelve al estado q_0 y se continua el proceso de la máquina de Turing con la segunda cinta vacía aunque en la cinta de trabajo después de la iteración anterior no este vacía.

El estado q_{print} es el único estado a partir del cual se puede ir a la izquierda reemplazando por caracteres blancos todo lo que se había escrito en la segunda cinta. Por otra parte, la máquina se detiene cuando llega al estado q_{reject}.

El lenguaje reconocido por el enumerador lo denotamos por $L(E)$ y viene dado por

$$L(E) = \{\omega \in \Sigma^* \mid \omega \text{ aparece en la segunda cita al llegar al estado } q_{\text{print}}\}$$

Por lo tanto un lenguaje L' se dice *lenguaje enumerado* si existe un enumerador E tal que $L' = L(E)$, y con esto completamos las definiciones solicitadas en el problema. □

Problema 3

En el *Teorema 3.21* demostramos que un lenguaje es reconocido por una máquina de Turing si y sólo si un enumerador lo enumera. ¿Por qué no usamos el siguiente simple algoritmo para probar la primer implicación? Como antes, $s_1, s_2, \ldots$ es una lista de todos las cadenas de Σ^*.

$E =$ "Ignora la entrada.

1. Repetir los siguiente para $i = 1, 2, 3, \ldots$.
2. Correr M en s_i.
3. Si lo acepta, imprimir s_i."

Solución: En la fase 2 de este algoritmo, si M itera indefinidamente al ingresar s_i para alguna cadena $s_i \in \Sigma^*$ entonces E no podrá escribir ninguna cadena después de s_i. Si esto ocurre y existe $s_j \in L(M)$ con $j > i$ entonces E no podra imprimir la cadena s_j.

Por lo tanto, con este algoritmo E falla al enumerar $L(M)$, y por esta razón no lo utilizamos en la prueba de la primer dirección del *Teorema 3.21*. □

Problema 4

Da una implementación-descripción de nivel de una máquina de Turing que decida el siguiente lenguaje sobre el alfabeto $\{0,1\}$.

$$L = \{\omega \mid \omega \text{ contiene el doble de 0's que de 1's}\}$$

Solución: La idea para diseñar una máquina de Turing M que reconozca L será utilizar un alfabeto de trabajo $\Gamma = \{0, 1, \mathbf{x}, \sqcup\}$ y recorrer la cadena iterativamente reemplazando un 1 por una **x** y luego dos 0's por **x**'s.

Para evitar confusiones y por simplicidad en la descripción del algoritmo decimos que marcamos un 1 (o un 0) si lo reemplazamos por **x** en la cinta de trabajo y movemos la cabeza a la derecha.

La implementación descrita por niveles de la máquina de Turing M será la siguiente, para la cadena de entrada $\omega \in \{0,1\}^*$ realizar los siguientes pasos:

1. Recorrer la cinta para encontrar el primer 1 y marcarlo.
   - Si no hay 1's para marcar ir al paso 5.
   - En otro caso, regresar la cabeza al inicio de la cinta de trabajo
2. Recorrer la cinta hasta encontrar el primer 0 y marcarlo.
   - Si no hay 0's para marcar se *rechaza* la cadena.
3. Recorrer la cita hasta que se encuentre otro 0 y luego marcarlo.
   - Si no hay 0's para marcar se *rechaza* la cadena.
4. Mover la cabeza al principio de la cinta y repetir el paso 1.
5. Colocar la cabeza al principio de la cinta y recorrerla para ver si hay 0's.
   - Si no se encuentra ningún 0 entonces se *acepta* la cadena.
   - En otro caso, se *rechaza* la cadena.

   Esta es la descripción buscada que nos sirve para implementar una máquina de Turing M de forma que $L = L(M)$. □

Problema 5

Una *máquina de Turing con reset izquierdo* es similar a una máquina de Turing ordinaria, pero con la función de transición de la siguiente forma

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{R, RESET\}.$$

Si $\delta(q, a) = (r, b, RESET)$, cuando la máquina está en el estado q al leer a, la máquina mueve la cabeza hacia la izquierda al inicio de la cinta después de escribir b y entrar al estado r.

Muestra que las máquinas de Turing con reset izquierdo reconocen los misma clase de lenguajes que las máquinas de Turing ordinarias.

Solución: En primer lugar, si tenemos una máquina de Turing con reset izquierdo se puede construir una máquina de Turing estándar que reconozca el mismo lenguaje observado que el reset izquierdo es una sucesión finita de desplazamientos de la cabeza a la izquierda.

Ahora para una máquina de Turing estándar construiremos una máquina con reset izquierdo que reconozca el mismo lenguaje.

Sea $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ una máquina de Turing ordinaria.

Denotaremos como $M_L = (Q, \Sigma, \Gamma_L, \delta', q_0, q_{\text{accept}}, q_{\text{reject}})$ a la máquina de Turing con reset izquierdo que reconocerá $L(M)$. La construcción la describiremos en los párrafos subsecuentes.

Para construir está máquina M_L consideraremos para cada $a \in \Sigma$ el caractér marcado $\dot{a} \in \Gamma_L$.

Consideraremos que si estamos en el estado $q \in Q$, en M_L tendremos que

$$(q, a) \to (r, b) \text{ y } (q, \dot{a}) \to (r, b),$$

es decir, tendremos la misma sustitución de estado y caracter en M_L si leemos a como el caracter marcado $\dot{a}$ correspondiente.

El algoritmo de funcionamiento de M_L es el siguiente:

1. Si M se mueve a la derecha entonces M_L hará lo mismo que M.
2. Si M se mueve a la izquierda entonces M_L hará lo siguiente:
   a) Marcar el caracter actual al que apunta la cabeza hacer y hacer la sustitución de estados que indique M. RESET al inicio de la cinta de trabajo.
   b) Marcar el primer caracter de la cinta al que se apunto. RESET.
   c) Recorrer la cinta a la derecha hasta hallar un caracter marcado. Moverse a la derecha una vez más.
   d) Si la celda a la que se apunta tiene ya un caracter marcado con punto, debe ser el caracter marcado al inicio del algoritmo.
      - Borrar el punto.
      - RESET.
      - Moverse hacia la derecha hasta encontrar un caracter marcado con punto.

      En este punto del algoritmo, la cabeza de la máquina apunta un lugar a la izquierda del lugar al que apuntaba al inicio del algoritmo, por lo que hemos terminado.
   e) Si la celda a la que se apunta no tiene un caracter marcado

- Marcar el caracter al que se apunta.
- RESET.
- Moverse a la derecha hasta encontrar un caracter marcado con punto.
- Borrar el punto.
- RESET.
- Ir al paso c).

La iteración de este algortimo el paso d) garantiza que en algún momento lleguemos a d) y d) se tiene la posición de la cabeza deseada cuando en M hay un transición a la izquierda. Y como consideramos la acción del control en M_L indistinta para los caracteres y los correspondientes marcados, entonces hay una relación biyectiva entre las configuraciones de M y M_L por lo se aceptan las mismas cadenas de entrada en cada máquina y así se tiene que M_L reconoce el lenguaje $L(M)$.

Para ilustrar como funciona este algoritmo, en primer lugar mostramos una sucesión de configuraciones en M

$$q_0abcd \to a'q_1bcd \to a'b'q_2cd \to a'q_3b'c'd.$$

La correpondiente configuración en M_L es la siguiente

$$\begin{aligned} q_0abcd \to a'q_1bcd \to & a'b'q_2cd \to q_3a'b'\dot{c}'d \\ \to \dot{a}'q_3b'\dot{c}'d \to q_3a'\dot{b}'\dot{c}'d \to & a'\dot{b}'q_3\dot{c}'d \to a'q_3\dot{b}'c'd, \end{aligned}$$

y esto ilustra el algoritmo y como ambas sucesiones de configuraciones tienen el mismo resultado final. □

Problema 6

Una *máquina de Turing con permanecer en vez de izquierda* es similar a una máquina de Turing ordinaria, pero la función de transición tiene la forma

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{R, S\}.$$

En este punto, la máquina puede mover su cabeza a la derecha o quedarse en el lugar al que apuntoa. Muestra que esta variante de máquina de Turing no es equivalente a la versión usual. ¿Qué clase de lenguaje reconoce esta máquina?

Solución: Sea $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ la máquina de Turing con permanecer en vez de ir a la izquierda.

Vamos a probar que esta máquina sólo puede reconocer lenguajes regulares. La razón de esto es que este tipo de máquina puede escribir en la cita pero no puede usar lo que ya escribió una vez que se mueve de la celda en la que escribió.

Probaremos esto construyendo un NFA N tal que reconozca el mismo lenguaje de M.

La idea detrás de la construcción es que el *NFA* almacenará el símbolo que leyó de la cinta en el control de la máquina, es decir, en los estados, y cuando M haga cambios en la celda a la que apunta N registrará esos cambios en los estados a través de transiciónes vacías ε. Y una vez que M se mueva a la derecha, N leerá el siguiente símbolo.

Escribiendo formalmente la idea anterior, sea $N = (Q', \Sigma, \delta', q_I, F)$ donde

- $Q' = Q \times (\Gamma \cup \{\text{rd}\})$. Entonces los estados son parejas (q, X) dibde q es un estado de M que es registrado y X es un símbolo que se lee en la cinta de M. Si $X = \text{rd}$ significa que M se movió a la derecha en el paso anterior por lo que N debe leer el siguiente símbolo de la cadena de entrada.
- $q_I = (q_0, \text{rd})$. Inicialmente, M está en q_0 y debemos leer el primer símbolo.
- $F = \{(q_{\text{accept}}, a) \mid (q_{\text{accept}}, a) \in Q'\}$. Por tanto aceptamos una cadena cuando en la simúlación de M este termina en un estado aceptor.
- La función de transición δ' queda definida por

$$\delta'((q, X), a). = \begin{cases} \{(q', X')\} & \text{si } X \neq \text{rd y } a = \varepsilon \text{ y } \delta(q, X) = (q', X', S) \\ \{(q', \text{rd})\} & \text{si } X \neq \text{rd y } a = \varepsilon \text{ y } \delta(q, X) = (q', X', R) \\ \{(q, a)\} & \text{si } X = \text{rd y } a \neq \varepsilon \end{cases}$$

Para probar que en efecto la construcción basta con verificar que para $\omega_i \in \Sigma$[1]

$$q_0\omega_1\omega_2 \ldots \omega_n \to a_1 \ldots a_{k-1} q b \omega_{k+1} \ldots \omega_n \Leftrightarrow (q, b) \in (\delta')^*(q_I, \omega_1\omega_2 \ldots \omega_k) \tag{1}$$

Lo anterior nos dice que, si M está leyendo k-ésima celda de la cinta de trabajo que contiene a b y su estado de control es q entonces N debe alcanzar el estado (q, b) después de leer $\omega_1, \ldots, \omega_{k-1}$, el inverso también se cumple.

Procederemos por inducción sobre el numéro de pasos en el cómputo de M.

[1] Ver Definición 2.12 John Martin, Introduction to Languages and Theory of Computation, *2010*

La base de inducción es clara por la definición de δ'. Supongamos el resultado se cumple para k, probaremos la proposición en el cómputo $k+1$ de M.

Sabemos que se cumple (1) en el cómputo k-ésimo, queremos probar

$$a_1 \dots a_k q' c \omega_{k+2} \dots \omega_n \Leftrightarrow (q', c) \in (\delta')^*(q_I, \omega_1 \omega_2 \dots \omega_{k+1}).$$

Notamos que según la notación de (1) $c = \omega_{k+1}$ pues M sólo se puede mover a la derecha o quedarse en el mismo lugar, luego observamos que

$$(\delta')^*(q_I, \omega_1 \omega_2 \dots \omega_{k+1}) = \delta'((\delta')^*(q_I, \omega_1 \dots \omega_k), c) \tag{2}$$

Además, en este caso suponemos que

$$\delta(q, b) = (q', a_k, R)$$

Por lo tanto

$$\delta'((q, b), \varepsilon) = \{(q', \mathrm{rd})\},$$

por lo que

$$\delta'((q, b), c) = \{(q', c)\},$$

y así se se cumple la primer dirección de 2.

Para la otra dirección notamos primero que

$$(q', c) \in \delta'((\delta')^*(q_I, \omega_1 \dots \omega_k), \omega_{k+1}),$$

luego existe $(q_0, b_0) \in (\delta')^*(q_I, \omega_1 \dots \omega_k)$ tal que

$$\{(q', c)\} = \delta'((q_0, b_0), \omega_{k+1}). \tag{3}$$

como $(q_0, b_0) \in (\delta')^*(q_I, \omega_1 \dots \omega_k)$ por hipótesis de inducción

$$q_0 \omega_1 \omega_2 \dots \omega_n \to a_1 \dots a_{k-1} q_0 b_0 \omega_{k+1} \dots \omega_n$$

Sin pérdida de generaidad supongamos que $b_0, \omega_{k+1} \neq \varepsilon$ de manera similar a la dirección anterior tenemos suponiendo que nos hemos desplazado a la derecha y de (3) que

$$\delta(q_0, b_0) = (q', a_k, R) \text{ para algún } a_k \in \Sigma,$$

así

$$a_1 \dots a_{k-1} q_0 b_0 \omega_{k+1} \dots \omega_n \to a_1 \dots a_k q' \omega_{k+1} \dots \omega_n$$

Si $\omega_{k+1} \neq \varepsilon$ y por como definimos δ' y recordando (3) se tiene que $\omega_{k+1} = c$, por tanto

$$a_1 \dots a_{k-1} q_0 b_0 \omega_{k+1} \dots \omega_n \to a_1 \dots a_k q' c \omega_{k+2} \dots \omega_n,$$

que es lo que queríamos probar. El caso suponiendo no nos movemos a la derecha es análogo, lo que concluye la inducción.

Por tanto hemos construido un NFA que reconoce el mismo lenguaje que M.

Ahora notamos que si un lenguaje es regular puede ser reconocido por un DFA, que se puede ver como una máquina de Turing en la cual la cinta es de sólo lectura y la cabeza sólo se mueve a la derecha, que es precisamente es el tipo de variante de máquina de Turing que estamos trabajando.

Por lo tanto, las máquinas de Turing con permanecer en vez de izquierda reconocen los lenguajes regulares, y ya hemos visto que las máquinas de Turing ordinarias reconocen una conjunto de lenguajes que contiene propiamente a los lenguajes regulares.

Concluimos así que la variante de máquina de Turing que sustituye el movimiento de la cabeza hacia la izquierda por poder dejar fija la cabeza no es equivalente a la versión usual. □