Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2223 — Teoría de Autómatas y Lenguajes Formales — 2' 2021

PAUTA TAREA 3

Pregunta 1

Una posible solución para esta pregunta es construir un algortimo a partir del siguiente lema que se obtiene de la definición de separabilidad.

Lema 1 *La palabra w es separable por L si, y solo si, la palabra se puede descomponer de la forma $w = u \cdot v$ tal que $u \in L$ y para todo prefijo u', donde $u' \neq \epsilon$ y $u' \neq u$, se tiene que $u' \notin L$, y se cumple que $v = \epsilon$ o v es separable.*

Demostraremos las dos direcciones del lema.

($\Rightarrow$) Sea w separable por L.

Por demostrar: La palabra w se puede descomponer de la forma $w = u \cdot v$ tal que $u \in L$ y para todo prefijo u', donde $u' \neq \epsilon$ y $u' \neq u$, se tiene que $u' \notin L$, y se cumple que $v = \epsilon$ o v es separable.

Se demostrará por inducción sobre el largo de w.

- Para $|w| = 1$, como w es separable por L, entonces necesariamente se cumple el caso 1 de la defición de separabilidad, es decir, para todo prefijo u', con $u' \neq u$, se tiene que $u' \notin L$ y, además, $v = \epsilon$.
- Suponemos que la propiedad se cumple para $|w| = i$, con $i \leq n-1$ y se analiza $|w| = n$. Como w es separable por L, se cumple alguno de los dos casos de la definición. Se revisan ambos casos.
  1. Si se cumple el caso 1, entonces $v = \epsilon$ y para todo prefijo de u' de w, con $u' \neq w$ y $u' \neq \epsilon$, se tiene que $u' \notin L$. De esta manera, se cumple la propiedad.
  2. Si se cumple el caso 2, esto quiere decir que $u \neq \epsilon$ y $v \neq \epsilon$, donde u y v son separables. Representando w de la siguiente manera:

     $$w = \underbrace{u_1 ... u_j}_{u'_1} \underbrace{u_{j+1} ... u_i}_{u'_2} \cdot v$$

     Como u es separable y la propiedad se cumple para $|w| = i$, con $i = 1, .., n-1$, entonces la palabra u se puede descomponer en $u = u'_1 \cdot u'_2$, donde $u'_1 \in L$ y todos los prefijos u'' de u'_1, tal que $u'' \neq u'_1$ y $u'' \neq \epsilon$, cumplirán que $u'' \notin L$.
     Ahora, si $u'_2 = \epsilon$, entonces se cumple la propiedad. Por otro lado, si $u'_2 \neq \epsilon$, entonces u'_2 debe ser separable dado que u es separable, teniendo así que $u'_2 \cdot v$ es separable, cumpliendose.

($\Rightarrow$) Supongamos que la palabra w se puede descomponer de la forma $w = u \cdot v$ tal que $u \in L$ y para todo prefijo u', donde $u' \neq \epsilon$ y $u' \neq u$, se tiene que $u' \notin L$, y se cumple que $v = \epsilon$ o v es separable.

Por demostrar: La palabra w es separable por L.

Se demuestra a partir de la definición. Se tiene que $w = u \cdot v$. Se sabe que $u \in L$ y todos sus prefijos u', con $u' \neq u$ y $u' \neq \epsilon$, cumplen que $u' \notin L$. Ahora, si:

- $v = \epsilon$: se cumple el caso 1 de la definición, por lo que w es separable por L.
- $v \neq \epsilon$: ya se sabe que v es separable. Se tiene que u también es separable porque u cumple el caso 1 de la definición. Como u y v son separables, se cumple el caso 2 de la definición, por lo que w es separable por L.

De esta manera, a partir del lema se construye el siguiente algoritmo:

```
Input: w ≠ ϵ and A = (Q, Σ, Δ, I, F)
Output: True si y solo si w es separable por ℒ(A)

Separable(w = a_1...a_n, A):
    S = I;
    i = 0;
    while (i < n)
        i++
        S' = ∅
        for each q ∈ S
            S' = S' ∪ {p / (q, a_i, p) ∈ Δ}
        if S' ∩ F ≠ ∅ then S = I
        else S = S'
    return S' ∩ F ≠ ∅
```

La intuición detrás del algoritmo corresponde a leer letra por letra la palabra w. Si en algún momento se llega a un estado final, es decir, se encuentra una subpalabra de w que es parte del lenguaje L, entonces desde la siguiente letra se sigue leyendo la palabra, pero como si fuera una nueva palabra, es decir, se reemplaza el conjunto de estados actuales S por el conjunto de estados iniciales. Esto se repite cada vez que se encuentra un estado final. El algoritmo retorna verdadero si y solo si al terminar de recorrer la palabra hay un estado final en el conjunto S. La complejidad del algoritmo es $O(|A| \cdot |w|)$.

Dado lo anterior, la distribución de puntaje es la siguiente:

- **(1 punto)** Por definir el lema y demostrarlo correctamente.
- **(3 puntos)** Por la construcción del algoritmo. En específico:
  - **(1 punto)** Por leer la palabra manteniendo un conjunto de estados actuales.
  - **(1 punto)** Por redefinir el conjunto de estados actuales como el conjunto de estados iniciales al encontrar un estado final.
  - **(1 punto)** Por retornar correctamente el resultado del algoritmo.

Pregunta 2

Pregunta 2.1

La respuesta es *Verdadero*.

Si R es una relación racional, entonces existe un transductor $\mathcal{T} = (Q, \Sigma, \Omega, \Delta, I, F)$ tal que $R = [\![\mathcal{T}]\!]$. Defina $\mathcal{T}^{-1} = (Q, \Omega, \Sigma, \Delta^{-1}, I, F)$ tal que $\Delta^{-1} = \{(p, b, a, q) \mid (p, a, b, q) \in \Delta.\}$.

Sean $u \in \Sigma^*$ y $v \in \Omega^*$. Demostramos utilizando el principio de inducción fuerte sobre el largo de u y v, que R^{-1} es un relación racional. En particular, $R^{-1} = [\![\mathcal{T}]\!]^{-1}$ y basta demostrar que $(u, v) \in [\![\mathcal{T}]\!]$ si, y solo si, $(v, u) \in [\![\mathcal{T}^{-1}]\!]$.

Hipótesis de inducción: $(p, u, \epsilon) \vdash^*_{\mathcal{T}} (q, \epsilon, v)$ si, y solo si, $(p, v, \epsilon) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, u)$.

- Caso base: Sean $a \in \Sigma$ y $b \in \Omega$. Tenemos dos casos:
  1. Si $(p, a, \epsilon) \vdash^*_{\mathcal{T}} (q, \epsilon, b)$, entonces $(p, b, \epsilon) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, a)$ porque en la ejecución de $\mathcal{T}^{-1}$ se utilizarán las transiciones utilizadas por $\mathcal{T}$ en el mismo orden, pero la escritura invertida con la lectura. Por ejemplo, si en un instante $\mathcal{T}$ lee a y escribe b, entonces $\mathcal{T}^{-1}$ leerá b y escribirá a. O si $\mathcal{T}$ lee a sin escribir y escribe b sin leer en algún orden, entonces $\mathcal{T}^{-1}$ leerá b sin escribir y escribirá a sin leer en el mismo orden.
  2. Si $(p, b, \epsilon) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, a)$, entonces $(p, a, \epsilon) \vdash^*_{\mathcal{T}} (q, \epsilon, b)$ por un argumento simétrico al anterior.
- Caso Inductivo: Sean $u, w \in \Sigma^*$, $a \in \Sigma$, $v, w' \in \Omega^*$ y $b \in \Omega$ tal que $u = aw$ y $v = bw'$. Tenemos dos direcciones que demostrar.

  ($\Rightarrow$) Demostraremos que si $(p, u, \epsilon) \vdash^*_{\mathcal{T}} (q, \epsilon, v)$, entonces $(p, v, \epsilon) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, u)$. Sabemos que $(p, aw, \epsilon) \vdash^*_{\mathcal{T}} (q, \epsilon, bw')$ si, y solo si, $(p, aw, \epsilon) \vdash^*_{\mathcal{T}} (p', w, b) \vdash^*_{\mathcal{T}} (q, \epsilon, bw')$ por el caso base. Luego, $(p', w, \epsilon) \vdash^*_{\mathcal{T}} (q, \epsilon, w')$ por definición de ejecución de un transductor.
  Por hipótesis de inducción se tiene que $(p', w', \epsilon) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, w)$. Y por definición de ejecución se deduce que $(p', w', a) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, aw)$. Además, $(p, bw', \epsilon) \vdash^*_{\mathcal{T}^{-1}} (p', w', a)$ por la definición de $\mathcal{T}^{-1}$. Uniendo las últimas 2 ejecuciones mencionadas se obtiene $(p, bw', \epsilon) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, aw)$, es decir, $(p, v, \epsilon) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, u)$.

  ($\Leftarrow$) Si $(p, v, \epsilon) \vdash^*_{\mathcal{T}^{-1}} (q, \epsilon, u)$, entonces $(p, u, \epsilon) \vdash^*_{\mathcal{T}} (q, \epsilon, v)$ por un argumento simétrico al anterior.

Dado lo anterior, la distribución de puntaje es la siguiente:

- **(3 puntos)** Por construir T^{-1} correctamente.
- **(1 punto)** Por la demostración.

Pregunta 2.2

La respuesta es *Verdadero*.

Si R y S son relaciones racionales, entonces existen los transductores $\mathcal{T}_1 = (Q_1, \Sigma, \Omega, \Delta_1, I_1, F_1)$ y $\mathcal{T}_2 = (Q_2, \Omega, \Gamma, \Delta_2, I_2, F_2)$ tal que $R = [\![\mathcal{T}_1]\!]$ y $S = [\![\mathcal{T}_2]\!]$. Defina $\mathcal{T} = (Q_1 \times Q_2, \Sigma, \Gamma, \Delta, I_1 \times I_2, F_1 \times F_2)$ tal que:

$$\begin{aligned}\Delta \quad = \quad & \{((p_1, p_2), a, c, (q_1, q_2)) \mid \exists b \in \Omega.\ (p_1, a, b, q_1) \in \Delta_1 \wedge (p_2, b, c, q_2) \in \Delta_2\} \cup \\ & \{((p_1, p_2), \epsilon, c, (p_1, q_2)) \mid \forall p_1 \in Q_1.\ (p_2, \epsilon, c, q_2) \in \Delta_2\} \cup \\ & \{((p_1, p_2), a, \epsilon, (q_1, p_2)) \mid \forall p_2 \in Q_2.\ (p_1, a, \epsilon, q_1) \in \Delta_1\}\end{aligned}$$

Intuitivamente, el transductor anterior hace lo siguiente. Se busca simular 2 ejecuciones. La de un transductor que genera un output y la de otro transductor que lo lee. Para hacer lo anterior en solamente una ejecución se utiliza la idea de una ejecución en paralelo. En cuanto al producto cruz, este permite almacenar la información necesaria para el paralelismo. En cuanto a Δ, la primera parte simula en un paso que T_1 y T_2 ejecuten un paso. La segunda y tercera parte de Δ simulan que un transductor avance sin la necesidad del otro. Si T_1 escribe ϵ esto no afecta el input de T_2 y puede hacerlo libremente. Si T_2 lee ϵ no es necesario que T_1 genere input él.

Sea $u \in \Sigma^*$, $w \in \Omega^*$ y $v \in \Gamma^*$. Por demostrar, a través del principio de inducción fuerte, que $(u, w) \in [\![\mathcal{T}_1]\!]$ y $(w, v) \in [\![\mathcal{T}_2]\!]$ si, y solo si, $(u, v) \in [\![\mathcal{T}]\!]$. Esto es, $(p_1, u, \epsilon) \vdash^*_{\mathcal{T}_1} (q_1, \epsilon, w)$ y $(p_2, w, \epsilon) \vdash^*_{\mathcal{T}_2} (q_2, \epsilon, v)$ si, y solo si, $((p_1, p_2), u, \epsilon) \vdash^*_{\mathcal{T}} ((q_1, q_2), \epsilon, v)$.

Dado lo anterior, la distribución de puntaje es la siguiente:

- **(2 puntos)** Por construir T correctamente hasta la primera parte de Δ.
- **(1 punto)** Por construir las 2 últimas partes de Δ.
- **(1 punto)** Por plantear la demostración por principio de inducción fuerte.