Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC2223 – Teoría de Autómatas y Lenguajes Formales
Segundo semestre de 2024
Profesor: Cristian Riveros
Ayudante: Amaranta Salas

Ayudantia 2

Construcción de Autómatas y No Determinismo

Problema 1

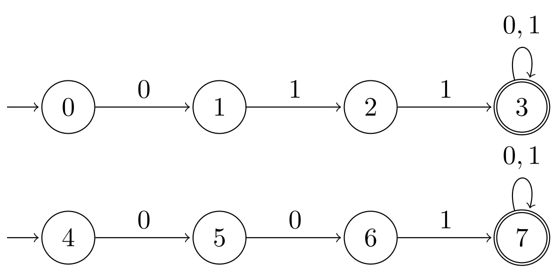
Considere $\Sigma = \{0, 1\}$ construya un NFA para cada uno de los siguientes lenguajes:

- $L = \{w = a_1a_2 \ldots a_{n-1}a_n \mid a_1a_2a_3 = 011 \vee a_1a_2a_3 = 001 \}$
- $L = \{w = a_1a_2 \ldots a_{n-1}a_n \mid a_1a_2 = 01 \vee a_{n-1}a_n = 01 \}$
- $L = \{w = a_1a_2 \ldots a_{n-1}a_n \mid a_{n-1} = a_n \}$

Solución

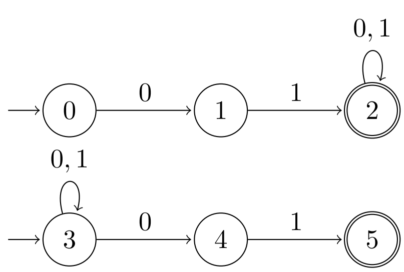
- $L = \{w = a_1a_2 \ldots a_{n-1}a_n \mid a_1a_2a_3 = 011 \vee a_1a_2a_3 = 001 \}$

  Intuitivamente, podemos crear dos autómatas, uno para cada una de las condiciones del *or*, y, dado que buscamos un autómata no determinista, el autómata formado por estos dos definirá el lenguaje que buscamos.



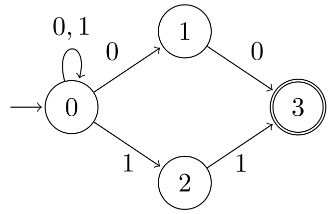
- $L = \{w = a_1a_2 \ldots a_{n-1}a_n \mid a_1a_2 = 01 \vee a_{n-1}a_n = 01 \}$

  Podemos seguir la misma estrategia en este caso, obteniendo:



- $L = \{w = a_1 a_2 \dots a_{n-1} a_n \mid a_{n-1} = a_n\}$

  Para este lenguaje, podemos crear el siguiente autómata:

Problema 2

Sea Σ un alfabeto finito y $L_1, L_2 \subseteq \Sigma^*$ dos lenguajes. Se define el lenguaje

$$L_1 \mid L_2 = \{w \in L_1 \mid \exists u, v \in \Sigma^* . \, w = u \cdot v \wedge v \in L_2\}$$

Demuestre que si L_1 y L_2 son lenguajes regulares, entonces $L_1 \mid L_2$ es regular.

Solución

Como L_1 y L_2 son lenguajes regulares, sabemos que existirán los DFA $\mathcal{A}_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$ y $\mathcal{A}_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$ tales que $\mathcal{L}(\mathcal{A}_1) = L_1$ y $\mathcal{L}(\mathcal{A}_2) = L_2$. Luego, para demostrar que $L_1|L_2$ es regular, encontraremos un autómata que lo defina y luego demostraremos que este es el caso.

Sea $\mathcal{A} = (Q_1 \cup (Q_1 \times Q_2), \Sigma, \Delta, \{q_{01}\}, F_1 \times F_2)$, con Δ dado por:

$$\begin{aligned}\Delta \; = & \{(p_1, a, q_1) \mid \delta_1(p_1, a) = q_1\} \cup \\ & \{(p_1, a, (q_1, q_{02})) \mid \delta_1(p_1, a) = q_1\} \cup \\ & \{((p_1, p_2), a, (q_1, q_2)) \mid \delta_1(p_1, a) = q_1 \wedge \delta_2(p_2, a) = q_2\}\end{aligned}$$

La idea de este autómata es simular la ejecución de $\mathcal{A}_1$ (primer conjunto de Δ), luego, de forma no determinista, iniciar la ejecución de $\mathcal{A}_2$ (segundo conjunto de Δ) y finalmente termina ejecutando ambos autómatas en paralelo (tercera parte de Δ).

Ahora debemos demostrar que $\mathcal{L}(\mathcal{A}) = L_1|L_2$.

- $L_1|L_2 \subseteq \mathcal{L}(\mathcal{A})$:

  Sea $w = uv \in L_1|L_2$. Sabemos entonces que $w \in L_1 \wedge v \in L_2$, con $w = a_1 ... a_n$ y $v = a_i ... a_n$. Luego, sabemos que existirán las siguientes ejecuciones de aceptación:

  $$\rho_1 : \; q_{01} \xrightarrow{a_1} p_1 \xrightarrow{a_2} \dots \xrightarrow{a_n} p_n \wedge p_n \in F_1$$

  $$\rho_2 : \; q_{02} \xrightarrow{a_i} q_i \xrightarrow{a_{i+1}} \dots \xrightarrow{a_n} q_n \wedge q_n \in F_2$$

  Con ρ_1 una ejecución de aceptación de $\mathcal{A}_1$ sobre w y ρ_2 una ejecución de aceptación de $\mathcal{A}_2$ sobre v.

  Luego, podemos construir:

  $$\rho : \; q_{01} \xrightarrow{a_1} p_1 \xrightarrow{a_2} \dots \xrightarrow{a_{i-2}} p_{i-2} \xrightarrow{a_{i-1}} (p_{i-1}, q_{02}) \xrightarrow{a_i} (p_i, q_i) \xrightarrow{a_{i+1}} \dots \xrightarrow{a_n} (p_n, q_n)$$

  Es claro que ρ es una ejecución de aceptación de $\mathcal{A}$ sobre w, pues q_{01} es el estado inicial de $\mathcal{A}$, es claro que $p_n \in F_1 \wedge q_n \in F_2$ y cada una de las transiciones de ρ sigue las reglas especificadas en Δ. Por tanto, $w \in \mathcal{L}(\mathcal{A})$ y entonces $L_1|L_2 \subseteq \mathcal{L}(\mathcal{A})$.

- $\mathcal{L}(\mathcal{A}) \subseteq L_1|L_2$:

  Sea $w \in \mathcal{L}(\mathcal{A})$. Por la construcción de $\mathcal{A}$, sabemos que existe una ejecución de aceptación de $\mathcal{A}$ sobre w que tendrá la siguiente forma:

  $$\rho:\ q_{01} \xrightarrow{a_1} p_1 \xrightarrow{a_2} ... \xrightarrow{a_{i-2}} p_{i-2} \xrightarrow{a_{i-1}} (p_{i-1}, q_{02}) \xrightarrow{a_i} (p_i, q_i) \xrightarrow{a_{i+1}} ... \xrightarrow{a_n} (p_n, q_n)\ \wedge\ p_n \in F_1\ \wedge\ q_n \in F_2$$

  Además, por definición de Δ, sabemos que para toda transición de la forma $p_{k-1} \xrightarrow{a_k} p_k$ se cumplirá que $\delta_1(p_{k-1}, a_k) = p_k$ y para toda transición de la forma $(p_{k-1}, q_{k-1}) \xrightarrow{a_k} (p_k, q_k)$ se cumplirá $\delta_1(p_{k-1}, a_k) = p_k$ y $\delta_2(q_{k-1}, a_k) = q_k$.

  Utilizando esto, podemos generar las siguientes ejecuciones de $\mathcal{A}_1$ y $\mathcal{A}_2$:

  $$\rho_1:\ q_{01} \xrightarrow{a_1} p_1 \xrightarrow{a_2} ... \xrightarrow{a_n} p_n\ \wedge\ p_n \in F_1$$

  $$\rho_2:\ q_{02} \xrightarrow{a_i} q_i \xrightarrow{a_{i+1}} ... \xrightarrow{a_n} q_n\ \wedge\ q_n \in F_2$$

  Es claro que ρ_1 y ρ_2 son ejecuciones de aceptación, lo que significa que $w = a_1...a_n \in L_1$, $v = a_i...a_n \in L_2$, y por tanto $\exists u.\ uv \in L_1\ \wedge\ v \in L_2$ y, es decir, $w \in L_1|L_2$.

Por lo anterior $\mathcal{L}(\mathcal{A}) = L_1|L_2$ y $L_1|L_2$ es un lenguaje regular.

Problema 3

La *distancia de Hamming* $H(u, v)$ entre dos palabras u y v sobre el alfabeto $\{0, 1\}$ es el nuúmero de posiciones en que difieren. Por ejemplo, $H(01, 00) = 1$ y $H(011, 110) = 2$. En el caso especial que $|u| \neq |v|$ la distancia de Hamming se define como infinita, esto es, $H(u, v) = \infty$. Si u es una palabra y L es un conjunto de palabras, la distancia de Hamming entre u y L se define como la distancia desde u a la palabra mas cercana en L:

$$H(u, L) = \min_{v \in L} H(u, v)$$

Para cualquier $L \subseteq \{0, 1\}^*$ y $k \geq 0$, considere el lenguaje:

$$N_k(L) = \{u \in \{0, 1\}^* \mid H(u, L) \leq k\}$$

esto es, el conjunto de palabras que están a distancia de Hamming a lo más k de L. Por ejemplo, $N_0(\{000\}) = \{000\}$, $N_1(\{000\}) = \{000, 100, 010, 001\}$ y $N_2(\{000\}) = \{0, 1\}^3 - \{111\}$.
Demuestre que si $L \subseteq \{0, 1\}^*$ es regular, entonces $N_k(L)$ es regular para todo $k \geq 0$.
(Hint: piense el caso para $k = 1$ y $k = 2$ y después generalice la construcción para cualquier k.)

Solución

Para este problema primero definiremos $\bar{a}$ como el 'opuesto', es decir, si $a = 0$ entonces $\bar{a} = 1$ y viceversa, y pensaremos en los casos $k = 0$ y $k = 1$. Se puede ver que si $k = 0$, no se aceptan errores, es decir, $N_0(L)$ es el mismo lenguaje que L. Después si $k = 1$, solo se acepta un error, por lo que podemos construir un NFA que tenga un contador de 0 o 1 dependiendo de cuantos errores ha visto hasta el momento, para esto sea $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ el autómata que define a L y sean $\mathcal{A}_0$ y $\mathcal{A}_1$ 2 copias de $\mathcal{A}$. Necesitamos 2 copias, ya que se ejecutará $\mathcal{A}_0$ cuando no se hayan visto errores, guardando el numero 0 en los estados, y al ver el primer error se pasará a ejecutar $\mathcal{A}_1$ con estados que guardan el numero 1 para indicar que ya se vió un error. Entonces definimos el NFA $\mathcal{A}^1 = (Q^1, \Sigma, \Delta^1, I^1, F^1)$ como

- $Q^1 = Q\ \times\ \{0, 1\}$
- $I^1 = \{(q_0, 0)\}$

- $F^1 = \{(q,l) \mid q \in F \wedge l \in \{0,1\}\}$ (la segunda condición no es necesaria, pero se deja para que se entienda mejor)
- $\Delta^1 = \{((p,0),a,(q,0)) \mid \delta(p,a) = q\} \cup \{((p,1),a,(q,1)) \mid \delta(p,a) = q\} \cup \{((p,0),a,(q,1)) \mid \delta(p,\bar{a}) = q\}$

Esta noción la podemos generalizar para cualquier k, creando $k+1$ copias del autómata $\mathcal{A}$ (que define el mismo lenguaje que L) y pasando al siguiente cuando se ve un error, aceptando a lo más k errores y siempre partiendo en $\mathcal{A}_0$. Por lo tanto, el NFA $\mathcal{A}^k = (Q^k, \Sigma, \Delta^k, I^k, F^k)$ nos queda como:

- $Q^k = Q \times \{0, \ldots, k\}$
- $I^k = \{(q_0, 0)\}$
- $F^k = \{(q,l) \mid q \in F \wedge l \in \{0, \ldots, k\}\}$ (la segunda condición no es necesaria, pero se deja para que se entienda mejor)
- $\Delta^k = \{((p,i),a,(q,i)) \mid \delta(p,a) = q \wedge i \leq k\} \cup \{((p,i),a,(q,i+1)) \mid \delta(p,\bar{a}) = q \wedge i < k\}$

Ahora debemos demostrar que $L(\mathcal{A}^k) = N_k(L)$, lo cual haremos por inducción sobre k.

- **CB:** $k = 0$
  - $\exists u \in L.\ H(w,u) = 0 \longrightarrow w \in L(\mathcal{A}^0)$:
    Sea $w \in N_0(L)$, por definición sabemos que existe $u \in L$ tal que $H(w,u) = 0$ por lo tanto $N_0(L)$ es el mismo lenguaje que L ya que no acepta errores. Luego sea ρ_0 una ejecución de w sobre $\mathcal{A}$ de la forma
    $$\rho_0 : (q_0, 0) \xrightarrow{a_1} (q_1, 0) \xrightarrow{a_2} \ldots \xrightarrow{a_n} (q_n, 0)$$
    Como no se aceptan errores, no se ejecutan las transiciones de errores 'manteniendose' en solo una máquina y como $w \in L$, necesariamente $q_n \in F$, por lo tanto $w \in \mathcal{A}^0$.
  - $w \in L(\mathcal{A}^0) \longrightarrow \exists u \in L.\ H(w,u) \leq 0$:
    Sea $w \in \mathcal{A}^0$, por definición existe una ejecución de aceptación
    $$\rho_0 : (q_0, 0) \xrightarrow{a_1} (q_1, 0) \xrightarrow{a_2} \ldots \xrightarrow{a_n} (q_n, 0)$$
    Donde $q_n \in F$ y como $k = 0$ solo se ejecutan las transiciones que simulan las de $\mathcal{A}$, es decir, las de la máquina que acepta el lenguaje original (sin errores). Por lo tanto como no acepta errores, $H(w,u) = 0$ para toda $u \in L$ y por lo tanto $w \in N_0(L)$.
- **HI:** $w \in L(\mathcal{A}^k) \longleftrightarrow \exists u \in L.\ H(w,u) \leq k$
- **TI:** PD: $w \in L(\mathcal{A}^{k+1}) \longleftrightarrow \exists u \in L.\ H(w,u) \leq k+1$
  - $\exists u \in L.\ H(w,u) \leq k+1 \longrightarrow w \in L(\mathcal{A}^{k+1})$:
    Sea $w = a_1 \ldots a_n \in N_k(L)$ y sea $v = b_1 \ldots b_n \in L$ tal que $H(w,v) = k+1$. Podemos separar ambas palabras en 3 trozos, desde las posiciones 1 a $i-1$, la posición i y desde $i+1$ a n como w_s, w_i, w_e y v_s, v_i, v_e respectivamente. SPDG consideramos que $H(w_s, v_s) = k$, $w_e = v_e$ y tenemos 2 casos para w_i y v_i, uno es que sean iguales y el otro distintos, pero solo nos interesa el caso en que sean distintos ya que sino habrían k errores y ya sabemos que se aceptan. Por lo tanto, podemos ejecutar w en $\mathcal{A}^{k+1}$
    $$\rho_0 : (q_0, 0) \xrightarrow{a_1} \ldots \xrightarrow{a_{i-1}} (q_{i-1}, k) \xrightarrow{a_i} (q_i, k+1) \xrightarrow{a_{i+1}} \ldots \xrightarrow{a_n} (q_n, k+1)$$
    Como sabemos que $w_i \neq v_i$, ahí se hace una transición del tipo $\delta(q_{i-1}, \bar{w_i}) = q_i$ y como $w_e = v_e$, sabemos que $q_n \in F$ ya que se siguen ejecutando las transiciones del autómata original pero el la siguiente copia y por lo tanto $w \in \mathcal{A}^{k+1}$.

- $w \in L(\mathcal{A}^{k+1}) \longrightarrow \exists u \in L.\ H(w,u) \leq k+1$:
  Sea $w = a_1 \dots a_n \in \mathcal{A}^k$, sea $v = b_1 \dots b_n \in L$, las cuales podemos escribir como $w_s w_i w_e$ y $v_s v_i v_e$, y sea ρ_k una ejecución de aceptación

$$\rho_0 : (q_0, 0) \xrightarrow{a_1} \dots \xrightarrow{a_{i-1}} (q_{i-1}, k) \xrightarrow{a_i} (q_i, k+1) \xrightarrow{a_{i+1}} \dots \xrightarrow{a_n} (q_n, k+1)$$

  Esto quiere decir que al ejecutar w se diferenció con v en $k+1$ oportunidades, llegando a $q_n \in F$ (por construcción), es decir, hasta a_{i-1} se habian encontrado k errores ($H(w_s, v_s)$), luego al leer a_i se ejecutó una transición del tipo $\delta(q_{i-1}, \bar{a}_i) = q_i$ es decir $a_i \neq b_i$ y luego $w_e = v_e$, por lo tanto $H(w,v) = k+1$. Como la distancia entre esas palabras es $k+1$ $w \in N_{k+1}(L)$, ya que existe una palabra con la máxima distancia permitida.

Por lo anterior $L(\mathcal{A}^k) = N_k(L)$ y $N_k(L)$ es regular para todo $k \geq 0$

Problema 4 (propuesto)

Sea Σ un alfabeto finito. Para dos lenguajes $L_1, L_2 \subseteq \Sigma^*$ se define la operación de "barajar" L_1 y L_2 como:

$$barajar(L_1, L_2) = \{a_1 a_2 \dots a_{2n} \in \Sigma^* \mid a_1 a_3 \dots a_{2n-3} a_{2n-1} \in L_1 \ \wedge \ a_2 a_4 \dots a_{2n-2} a_{2n} \in L_2\}$$

Demuestre que si $\mathcal{A}_1$ y $\mathcal{A}_2$ son autómatas finitos no-deterministas, entonces existe un autómata finito no-determinista $\mathcal{A}$ tal que:

$$L(\mathcal{A}) = barajar(L(\mathcal{A}_1), L(\mathcal{A}_2)).$$

Demuestre como construir $\mathcal{A}$ a partir de $\mathcal{A}_1$ y $\mathcal{A}_2$, como también demuestre la correctitud de su construcción.