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

IIC2223 - Teoría de Autómatas y Lenguajes Formales

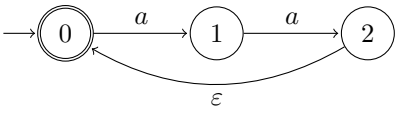
Ayudantía 3

Franco Bruña y Dante Pinto
10 de Septiembre, 2021

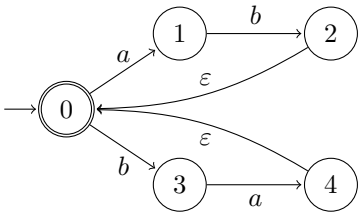
Pregunta 1

Para los siguientes lenguajes, $L \subseteq \{a, b\}^*$, defina su clausura de Kleene y construya un autómata que la acepte:

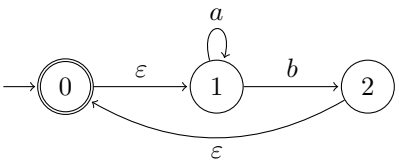
- $L = \{aa\} \Rightarrow L^* = (aa)^*$



- $L = \{ab,\, ba\} \Rightarrow L^* = (ab + ba)^*$



- $L = \{a^n b \mid n \geq 0\} \Rightarrow L^* = (a^*b)^*$



Pregunta 2

La *distancia de Hamming* $H(u, v)$ entre dos palabras u y v sobre el alfabeto $\{0, 1\}$ es el número de posiciones en que sus letras difieren. Por ejemplo, $H(01, 00) = 1$ y $H(011, 110) = 2$. En el caso en que $|u| \neq |v|$ la distancia de Hamming se define como infinita.
Por otra parte, definimos la distancia de Hamming entre una palabra u y un lenguaje L como la distancia desde u a la palabra más cercana en L, es decir, la menor de las distancias de Hamming entre u y cada una de las palabras de L:

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

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

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

Este lenguaje representa 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 $k \geq 0$.
Podemos interpretar el lenguaje $N_k(L)$ como las palabras de L para las que el autómata comete a lo más "k errores" en su ejecución de la palabra.
Siguiendo esta idea, si podemos crear un autómata que defina el mismo lenguaje que el original, pero se equivoque en a lo más una letra, podemos repetir esta construcción y construir de manera inductiva el autómata para cualquier k.
Dado que L es regular, sabemos que existirá un DFA $A = (Q, \Sigma, \Delta, q_0, F)$ que lo defina y podemos construir nuestro autómata que cometa un error más que A como $A_1(A) = (Q \times \{0, 1\}, \Sigma, \Delta_1, I_1, F_1)$ dado por:

- $I_1 = (q_0, 0), (q_0, 1)$
- $F_1 = (f, n) \mid f \in F \wedge n \in \{0, 1\}$
- $$\begin{aligned}\Delta_1 = &((p, n), a, (q, n)) \mid n \in \{0, 1\} \wedge (p, a, q) \in \Delta \\ &\cup ((p, 0), a, (q, 1)) \mid (p, a, q) \in \Delta\end{aligned}$$

La idea de este autómata es copiar el original dos veces y mantener la cuenta del número de errores en los estados. En cualquier estado, el autómata puede tomar, de forma no determinista, una transición hacia el estado copia, aumentando en 1 la cuenta de los errores cometidos. Cabe señalar que el autómata puede nunca tomar esta transición, llegando a un estado final con 0 errores, aceptando de esta manera al lenguaje orignial (palabras con distancia de Hamming 0) y además que no existirán transiciones desde el autómata con cuenta 1 hacia el autómata con cuenta 0, por lo que no podrá cometer más de un error.

Teniendo lo anterior, podemos ver que si bien nuestro comete solamente un error, lo que hicimos en realidad fue construir un autómata que agrega un error al autómata que recibe, es decir, si consideramos al autómata original como A_0, podemos redefinir nuestra construcción como $A_{k+1}(A)$. Luego, la operación realizada anteriormente sería $A_1 = A_{k+1}(A_0)$ y podremos repetirla para generar el autómata asociado a cualquier k.

Finalmente, podemos demostrar que la construcción es correcta usando inducción. Trivialmente tenemos que $N_0(L) = L \ \wedge \ A = A_0 \Rightarrow N_0(L) = \mathcal{L}(A_0)$, por lo que bastará demostrar que $N_k(L) = \mathcal{L}(A_k)$.

- $N_k(L) \subseteq \mathcal{L}(A_k)$

Sabemos que $\forall j < k . N_j(L) = \mathcal{L}(A_j)$ y, en particular $N_{k-1}(L) = \mathcal{L}(A_{k-1})$, luego, tomando $A_k = A_{k+1}(A_{k-1})$, es claro que $\mathcal{L}(A_{k-1}) \subseteq \mathcal{L}(A_k)$, por lo que bastará con demostrar que el autómata puede generar las palabras con distancia de Hamming exactamente k.

$H(w, L) = k \Rightarrow w \in \mathcal{L}(A_k)$

Si $H(w, L) = k$ tenemos que $|w| \geq k$ y que existe una palabra $w_0 \in L$ tal que $H(w, w_0) = k$. Luego, existirá una ejecución de aceptación de $A = A_0$ sobre w_0 dada por:

$$\rho_0 : \ q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} ... \xrightarrow{a_n} q_n \ \wedge \ q_n \in F$$

Considerando ahora, sin pérdida de generalidad, que las diferencias entre las palabras se encuentran en las primeras k letras de w, tendremos la siguiente ejecución de A_k sobre w:

$$\rho_k : \ (q_0, 0, ..., 0) \xrightarrow{b_1} (q_1, 1, ..., 0) \xrightarrow{b_2} ... \xrightarrow{b_k} (q_k, 1, ..., 1) \xrightarrow{a_{k+1}} ... \xrightarrow{a_n} (q_n, 1, ..., 1) \ \wedge \ q_n \in F$$

Donde $a_i \neq b_i \, \forall i$ y tenemos k partimos con k números 0 en junto al estado q_0 y terminamos con k números 1 al lado de los estados, desde q_k en adelante. Es claro que esta ejecución existirá, pues cambiaremos de copia en el autómata con cada b_i, cambiando exactamente k veces, el estado q_n será de aceptación (y podemos generalizar lo anterior para hacer los k cambios en cualquier momento de la ejcución). Por tanto, $w \in \mathcal{L}(A_k)$

- $\mathcal{L}(A_k) \subseteq N_k(L)$

  Podemos demostrar esta dirección por contradicción. Sea w una palabra tal que $H(w, L) > k \ \wedge \ w \in \mathcal{L}(A_k)$. Lo anterior significa que tenemos una ejecución de aceptación de la siguiente manera:

$$\rho_w : \ (q_0, 0, ..., 0) \xrightarrow{b_1} (q_1, 1, ..., 0) \xrightarrow{b_2} ... \xrightarrow{b_k} (q_k, 1, ..., 1) \xrightarrow{b_{k+1}} (q_{k+1}, 1, ..., 1) \xrightarrow{b_{k+2}} ... \xrightarrow{a_n} (q_n, 1, ..., 1) \wedge q_n \in F$$

  Donde $a_i \neq b_i \, \forall i$. Sin embargo, si miramos el estado $(q_k, 1, ..., 1)$ y nuestra definición de Δ, podemos ver que toda transición entre este y $(q_{k+1}, 1, ..., 1)$ debe estar definida en el autómata original, pues ya no hay más copias a las cuales cambiarse). Esto significa entonces que existirá una palabra w' en L que difiera en las k letras anteriores, pero que sea idéntica a esta desde la letra $k + 1$, lo que a su vez implica que $H(w, w') = k \rightarrow H(w, L) = k$; sin embargo, esto se contradice con nuestro supuesto inicial de $H(w, L) > k$, por lo que esta palabra no puede pertenecer al lenguaje.

Pregunta 3

Sea $\Sigma = \{0, 1\}^*$. Determine si los siguientes lenguajes $L \subseteq \Sigma$ son regulares. Si lo son, de una expresión regular o autómata que los defina, si no lo son, demuestre que este es el caso.

- $L_k = \{w \in \{0, 1\}^* \mid w = w^r \ \wedge \ |w| = k\}$, dónde $k \in \mathbb{N}$

  Este lenguaje es regular y lo podemos definir inductivamente con las siguientes expresiones regulares:

  - R_0: ε
  - R_1: $0 + 1$
  - R_k: $0\,(R_{k-2})\,0 + 1\,(R_{k-2})\,1$

  Donde $0(R_1)0$ es el resultado de concatenar 0 con la expresión regular R_1 y luego con 0 nuevamente, es decir, $0(R_1)0 = 0(0 + 1)0 \equiv 000 + 010$.

  Cabe señalar que no estamos definiendo una "Expresión regular recursiva", si no que estamos definiendo una expresión regular para cada valor de k.

- $L = \{w \in \{0,1\}^* \mid w = w^r\}$

  Este lenguaje **no es regular**, y podemos demostrarlo usando el lema de bombeo.

  Sea $w = 0^N 1^N 1^N 0^N$, es claro que $w \in L$ para todo $N > 0$ y que podemos tomar $x = 0^N$, $y = 1^N$ $z = 1^N 0^N$. Luego, toda división de y tendrá la forma:

  $y = u \cdot v \cdot w = 1^a 1^b 1^c$, con $a + b + c = N$ y tomando cualquier $i > 1$ tendremos:

  $x \cdot u \cdot v^i \cdot w \cdot z = 0^N 1^{N+(i-1)b} 1^N 0^N \notin L$, pues claramente no es un palíndromo y, por tanto, el lenguaje no es regular.

- $L = \{w = 0^n 10^m 1 \mid n \geq 0 \wedge m \geq 0\}$

  Para este lenguaje podemos definir la siguiente expresión regular:

  $$0^* 10^* 1$$

- $L = \{w = 0^n 10^n 1 \mid n \geq 0\}$

  Este lenguaje **no es regular**, y podemos demostrarlo usando el lema de bombeo.

  Sea $w = 0^N 10^N 1$, es claro que $w \in L$ para todo $N > 0$ y que podemos tomar $x = \varepsilon$, $y = 0^N$ $z = 10^N 1$. Luego, toda división de y tendrá la forma:

  $y = u \cdot v \cdot w = 0^a 0^b 0^c$, con $a + b + c = N$ y tomando cualquier $i > 1$ tendremos:

  $x \cdot u \cdot v^i \cdot w \cdot z = 0^{N+(i-1)b} 10^N 1 \notin L$, pues la cantidad de 0s no será la misma en ambos lados de la palabra.

- $L = \{w \in \{0,1\}^* \mid |w| \mod 2 \equiv 0\}$

  Para este lenguaje podemos definir la siguiente expresión regular:

  $$(00 + 01 + 10 + 11)^*$$

- $L = \{w \in \{0,1\}^* \mid |w|_0 = |w|_1 \}$, dónde $|w|_a$ representa el número de símbolos a en w.

  Este lenguaje **no es regular**, y podemos demostrarlo usando el lema de bombeo.

  Sea $w = 0^N 1^N$, es claro que $w \in L$ para todo $N > 0$ y que podemos tomar $x = 0^N$, $y = 1^N$ $z = \varepsilon$. Luego, toda división de y tendrá la forma:

  $y = u \cdot v \cdot w = 1^a 1^b 1^c$, con $a + b + c = N$ y tomando cualquier $i > 1$ tendremos:

  $x \cdot u \cdot v^i \cdot w \cdot z = 0^N 1^{N+(i-1)b} \notin L$, pues la cantidad de 1s será mayor que la de 0s.