Lenguajes y Autómatas finitos

1. Lenguajes. Conceptos fundamentales

Sea Σ una colección finita de símbolos. Este conjunto de símbolos se denomina **alfabeto** y los elementos **letras**. Una palabra sobre Σ es una cadena de longitud finita de elementos de Σ. La *cadena vacía* o *cadena nula*, denotada por ε, es una cadena que no contiene símbolos. El conjunto de todas las palabras sobre Σ se denota Σ^*. ε es una palabra en cualquier alfabeto Σ. Un *lenguaje sobre* Σ es un subconjunto de Σ^*. Si $\mathcal{L}$ denota el conjunto de los lenguajes sobre Σ se tiene que $\mathcal{L} = \{L : L \subset \Sigma^*\}$

Note que ε, la cadena vacía, es la cadena que no contiene símbolos. Es diferente de $\varnothing$ el conjunto vacío (lenguaje vacío). Se sigue que $\{\varepsilon\}$ es el conjunto que contiene exactamente una cadena, de hecho, la cadena vacía.

La longitud de una palabra $u \in \Sigma^*$, denotada por $|u|$, es el número de posiciones que tiene la palabra. Se puede definir recursivamente de esta manera:

$$|\varepsilon| = 0$$
$$|ua| = |u| + 1$$

donde $u \in \Sigma^*$ y $a \in \Sigma$.

De este modo una palabra $u \in \Sigma^*$ se puede definir por una función $u : \{1, 2, \ldots, |u|\} \to \Sigma$ donde $u(i)$ es la letra que se encuentra en la posición i en la palabra u.

Dos palabras $u, v \in \Sigma^*$ son iguales si $|u| = |v|$ y $u(i) = v(i)$ para $1 \leq i \leq |u|$.

Definimos una operación sobre Σ^* denominada **concatenación**. Para $u, v \in \Sigma^*$ la concatenación de u y v se denota uv y se define a través de la función $uv : \{1, 2, \ldots, |u| + |v|\} \to \Sigma$

$$uv(i) = \begin{cases} u(i), & \text{si} \quad 1 \leq i \leq |u| \\ v(i - |u|), & \text{si} \quad |u| + 1 \leq i \leq |u| + |v| \end{cases}$$

Así la palabra uv es la palabra que se obtiene escribiendo las letras de u y luego las letras de v. Si $u = u_1 u_2 \cdots u_k$ y $v = v_1 v_2 \cdots v_s$ entonces $uv = u_1 u_2 \cdots u_k v_1 v_2 \cdots v_s$ donde $u_i, v_j \in \Sigma$. Se deja al lector verificar que esta operación es asociativa.

Ejercicio 1 *Pruebe que* $|uv| = |u| + |v|$

Definición 1 *Dados dos lenguajes* $L, M \in \mathcal{L}$*, la concatenación de los lenguajes* L *y* M *se denota por* LM *y se define*

$$LM = \{vw : v \in L, w \in M\}$$

Es fácil ver que la concatenación de lenguajes es asociativa.

Ejemplo 1 *Sean* $\Sigma = \{0,1\}$ *y* L, M *dos lenguajes sobre* Σ *dados por* $L = \{1,10\}$ *y* $M = \{1,01\}$ *entonces* $LM = \{11,101,1001\}$. *Mientras que* $ML = \{11,110,001,0110\}$.

Definición 2 *Si* Σ *es un alfabeto y* $n \in \mathbb{N}$ *se define las potencias de* Σ *recursivamente de la siguiente manera:*

1) $\Sigma^1 = \Sigma$

2) $\Sigma^{n+1} = \Sigma\Sigma^n$

Por convención se tiene que $\Sigma^0 = \{\varepsilon\}$.

Ejemplo 2 *Si* $\Sigma = \{a,b,c\}$ *entonces* $\Sigma^2 = \{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$

Ejercicio 2 *Pruebe que* $\Sigma^* = \bigcup_{n\geq 0} \Sigma^n$.

Definimos $\Sigma^+ = \bigcup_{n\geq 1} \Sigma^n$.
Este es el conjunto de todas las palabras de longitud mayor que 1. Puesto que ε nunca es elemento de nuestro alfabeto, es decir, $\varepsilon \notin \Sigma$ entonces $\Sigma^* \neq \Sigma^+$ y $\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$

Definición 3 *Para* L *un lenguaje se define de manera recursiva* L^n *así:*

$L^0 = \{\varepsilon\}$
$L^{n+1} = LL^n$

L^* *se define*

$$L^* = \bigcup_{n\geq 0} L^n$$

L^* *se denomina la* **Cerradura de Kleene** *o* **Cerradura estrella** *de* L.

Ejemplo 3 *Sea* $\Sigma = \{0,1\}$ *y* $L = \{01,1\}$, *entonces*

$$L^3 = \{010101, 01011, 01101, 0111, 10101, 1011, 1101, 111\}$$

2. Lenguajes y Expresiones Regulares

En aritmética, usamos las operaciones + y × para construir expresiones tales como

$$(4+1)\times 5$$

De manera similar, usamos operaciones regulares para construir expresiones que describen lenguajes, las cuales se denominan **expresiones regulares**. Un ejemplo es:

$$(0 \cup 1)0^*$$

El valor de la expresión aritmética es 25. El valor de una expresión regular es un lenguaje. En este caso el lenguaje consiste en todas las palabras que empiezan con 1 o 0 seguido por cualquier número (finito) de $0\,s$. Otro ejemplo de una expresión regular es

$$(0 \cup 1)^*$$

Empieza con el lenguaje $(0 \cup 1)$ y se aplica la operación $*$. El valor de esta expresión son todas las palabras posibles de $0\,s$ y $1s$.

Lenguajes Regulares

Para un alfabeto Σ dado, los lenguajes regulares constituyen el menor conjunto de lenguajes sobre Σ que es cerrado respecto a las operaciones de concatenación, la cerradura de Kleene y la unión de lenguajes y además contiene el lenguaje $\varnothing$ y los lenguajes unitarios $\{a\}$ para $a \in \Sigma$.

Definición 4 *Sea Σ un alfabeto. El conjunto de lenguajes regulares sobre Σ se define recursivamente como sigue:*

1. *$\varnothing$ y $\{\varepsilon\}$ son lenguajes regulares*
2. *Para toda $a \in \Sigma$, $\{a\}$ es un lenguaje regular.*
3. *Si L y M son lenguajes regulares, entonces $L \cup M$, LM, L^* son lenguajes regulares.*
4. *Ningún otro lenguaje sobre Σ es regular.*

Ejemplo 4 *Dado $\Sigma = \{a, b\}$, las siguientes afirmaciones son verdaderas:*

$\varnothing$ y $\{\varepsilon\}$ son lenguajes regulares.

$\{a\}$ y $\{b\}$ son lenguajes regulares.

$\{a, b\}$ es un lenguaje regular.

$\{ab\}$ es regular.

$\{a^i | i \geq 0\}$ es regular.

Podemos simplificar la representación de un lenguaje regular por medio de una abreviatura llamada *expresión regular*. Convenimos en escribir a en lugar del lenguaje unitario $\{a\}$. Por tanto

$$\begin{array}{rll} a \cup b & \text{denota} & \{a, b\} = \{a\} \cup \{b\} \\ ab & \text{denota} & \{ab\} \\ a^* & \text{denota} & \{a\}^* \\ a^+ & \text{denota} & \{a\}^+ \end{array}$$

Además se establece un orden en los operadores, primero la clausura de Kleene ($*$), luego la concatenación y por último la unión. En algunos casos esto simplifica la expresión. Por ejemplo, una expresión $\{a\} \cup (\{b\}^*\{c\})$, se reduce a la expresión regular $a \cup b^*c$. De este modo, si se realiza la unión previa a la concatenación, se deben usar paréntesis; así $(a \cup b)c$ indica que se realiza la unión de $\{a\}$ y $\{b\}$ y el resultado se concatena con $\{c\}$ a la derecha. Si escribimos $a \cup bc$ debe ser leído: se concatena $\{b\}$ y $\{c\}$ en ese orden y el resultado se une con $\{a\}$.

Definición Formal de una Expresión Regular

Definición 5 *Definimos las expresiones regulares de manera recursiva:*

1. *$\varnothing$ y ε son expresiones regulares.*
2. *a es una expresión regular para toda $a \in \Sigma$.*
3. *Si r y s son expresiones regulares, entonces $r \cup s$, rs, r^* son expresiones regulares.*

4. *Ninguna otra secuencia de símbolos es una expresión regular.*

Comparando las definiciones de lenguajes regulares y expresiones regulares se deduce que toda expresión regular sobre Σ denota un lenguaje regular sobre Σ.

Por ejemplo el lenguaje de todas las palabras sobre $\{a, b, c\}$ que no tienen ninguna subcadena ac se denota mediante la expresión regular $c^*(a \cup bc^*)^*$.

Cuando sea necesario distinguir entre una expresión regular r y el lenguaje denotado por la misma, usaremos $L(r)$ para identificar dicho lenguaje. Así, si se afirma $w \in r$, esto significa que $w \in L(r)$. Si r y s son expresiones sobre el mismo alfabeto y si $L(r) = L(s)$, entonces se dice que r y s son *equivalentes*. en el caso que r y s sean equivalentes se puede escribir $r = s$. también se puede usar $r \subseteq s$ en el caso que $L(r) \subseteq L(s)$.

En la definición de la cerradura estrella se obtiene que $\varnothing^* = \{\varepsilon\}$, y en términos de expresiones regulares se tiene que $\varnothing^* = \varepsilon$.

Obsérvese que hay muchas expresiones regulares que denotan el mismo lenguaje. Por ejemplo $(a^*b)^*$ y $\varepsilon \cup (a \cup b)^*b$ denotan el mismo lenguaje: el lenguaje de todas las palabras con 0 o más $a\,s$ y $b\,s$, que son la palabra vacía o las que tienen una b al final. De este modo $(a^*b)^* = \varepsilon \cup (a \cup b)^*b$. Se pueden simplificar expresiones regulares reemplazándolas por otras equivalentes pero menos complejas. Existen muchas equivalencias en relación a expresiones regulares. Las resumimos en el siguiente teorema.

Teorema 1 *Sean r, s y t expresiones regulares sobre el mismo alfabeto Σ. Entonces:*

1. $r \cup s = s \cup r$
2. $r \cup \varnothing = r = \varnothing \cup r$
3. $r \cup r = r$
4. $(r \cup s) \cup t = r \cup (s \cup t)$
5. $r\varepsilon = r = \varepsilon r$
6. $r\varnothing = \varnothing r = \varnothing$
7. $r(st) = (rs)t$
8. $(r \cup s)t = rt \cup st \quad y \quad r(s \cup t) = rs \cup rt$
9. $r^* = r^{**} = r^*r^* = (\varepsilon \cup r)^* = r^*(\varepsilon \cup r) = (\varepsilon \cup r)r^* = \varepsilon \cup rr^*$
10. $(r \cup s)^* = (r^* \cup s^*)^* = (r^*s^*)^* = (r^*s)^*r^* = r^*(sr^*)^*$
11. $r(sr)^* = (rs)^*r$
12. $(r^*s)^* = \varepsilon \cup (r \cup s)^*s$
13. $(rs^*)^* = \varepsilon \cup r(r \cup s)^*$
14. $s(r \cup \varepsilon)^*(r \cup \varepsilon) \cup s = sr^*$
15. $rr^* = r^*r$

Ejercicio 3

1. *Verificar, aplicando la definición de lenguaje regular, que los siguientes son lenguajes regulares sobre $\Sigma = \{a, b\}$:*
   (a) $\{a^i \mid i > 0\}$.
   (b) $\{a^i \mid i > n\}$ *para un* $n \geq 0$ *fijado.*
   (c) $\{w \in \Sigma^* \mid w$ *termina con* $a\}$.
   (d) $\{w \in \Sigma^* \mid w$ *tiene un número par de* $aes\}$.

2. *Verificar que el lenguaje de todas las palabras de unos y ceros que tienen al menos dos ceros consecutivos, es un lenguaje regular.*

3. *Sobre* $\Sigma = \{a, b, c\}$*, determinar cuales parejas de expresiones regulares son equivalentes.*

   (a) $(a \cup b)^* a^*$ *y* $((a \cup b)a)^*$.

   (b) $\varnothing^{**}$ *y* ε.

   (c) $((a \cup b)c)^*$ *y* $(ac \cup bc)^*$

   (d) $b(ab \cup ac)$ *y* $(ba \cup ba)(b \cup c)$.

4. *Simplificar:*

   (a) $\varnothing^* \cup a^* \cup b^* \cup (a \cup b)^*$.

   (b) $((a^*b^*)^*(b^*a^*)^*)^*$.

   (c) $(a^*b)^* \cup (b^*a)^*$.

   (d) $(a \cup b)^* a (a \cup b)^*$.

5. *Probar que* $(aa)^*a = a(aa)^*$.

6. *Simplificar las siguientes expresiones regulares:*

   (a) $(\varepsilon \cup aa)^*$.

   (b) $(\varepsilon \cup aa)(\varepsilon \cup aa)^*$.

   (c) $a(\varepsilon \cup aa)^* a \cup \varepsilon$.

   (d) $a(\varepsilon \cup aa)^*(\varepsilon \cup aa) \cup a$.

   (e) $(a \cup \varepsilon)a^*b$.

   (f) $(\varepsilon \cup aa)^*(\varepsilon \cup aa)a \cup a$.

   (g) $(\varepsilon \cup aa)(\varepsilon \cup aa)^*(\varepsilon \cup aa)(\varepsilon \cup aa)$.

   (h) $(\varepsilon \cup aa)(\varepsilon \cup aa)^*(ab \cup b)(ab \cup b)$.

   (i) $(a \cup b)(\varepsilon \cup aa)^*(\varepsilon \cup aa) \cup (a \cup b)$.

   (j) $(aa)^*a \cup (aa)^*$.

   (k) $a^*b((a \cup b)a^*b)^* \cup a^*b$.

   (l) $a^*b((a \cup b)a^*b)^*(a \cup b)(aa)^* \cup a(aa)^* \cup a^*b((a \cup b)a^*b)^*$.

3. Autómata Finito Determinista

Consideremos el lenguaje regular L representado por $c^*(a \cup bc^*)^*$. Dada una palabra w ¿cómo saber si w pertenece al lenguaje L? Debemos analizar no sólo los caracteres que aparecen en w, sino también sus posiciones relativas. Por ejemplo, la cadena abc^3ab está en L, pero *cabacbc* no lo está. Podemos construir un diagrama que nos ayude a determinar los distintos miembros del lenguaje. Tal diagrama tiene la forma de un grafo dirigido son información adicional añadida, y se llama *diagrama de transición*. Los vértices del grafo se llaman *estados* y se usan para señalar, en ese momento, hasta que lugar se ha realizado la cadena. Las aristas del grafo se etiquetan con caracteres del alfabeto y se llaman *transiciones*. Si el siguiente carácter a reconocer concuerda con la etiqueta de alguna transición que parte del estado actual, nos desplazamos al estado al que nos lleve la arista correspondiente. Se comienza en un *estado inicial*, y cuando se hayan tratado todos los caracteres de la palabra (cadena) correspondiente, necesitamos saber si la cadena es "legal". Para ello se marcan ciertos estados

como *estados de aceptación* o *estados finales* (doble círculo). Toda cadena que surja de una transición desde el estado inicial a un estado final de aceptación es "legal". Marcamos el estado inicial con una flecha ($\rightarrow$) y alrededor de los estados de aceptación trazamos un círculo.

Por ejemplo el diagrama de la Figura 1 acepta todas las cadenas que están formadas por 0 o más $a\,s$ seguidas por una única b. Obsérvese que para toda cadena de la forma $a^k b$, para $k \geq 0$, el recorrido del diagrama termina en un estado de aceptación. El recorrido del mismo con cualquier otra cadena de $a\,s$ y $b\,s$ (incluida la cadena vacía) termina en cualquier otro estado, pero este no es de aceptación.

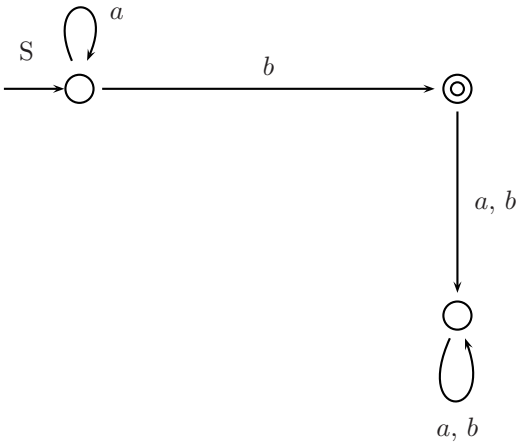


Figura 1:

Considérese el lenguaje $A = \{(ab)^i \mid i \geq 1\}$, el cual está representado por la expresión regular $(ab)^+$. La palabra más corta de este lenguaje es ab. El estado inicial no es un estado de aceptación, porque entonces aceptaría la palabra vacía. Hay dos transiciones una para a y una para b; ninguna de ellas puede llevar a un estado de aceptación. Esto se debe a que ni a ni b son palabras del lenguaje A. Las cadenas legales se obtienen al llegar a un estado de aceptación y se debe hacer esto después de un número finito de pares de transiciones ab. Un diagrama de transición para A es el que muestra la Figura 2. Este diagrama tiene un único

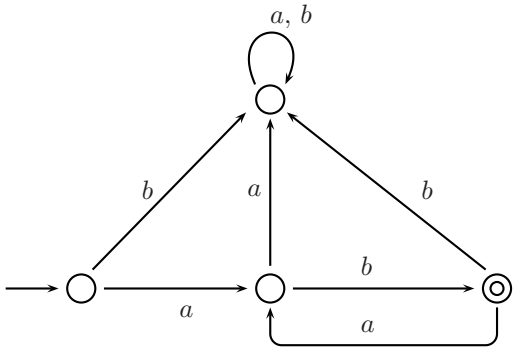


Figura 2:

estado de aceptación. Si el análisis termina en cualquier otro estado, la cadena no está correctamente construida. Asimismo se puede ver que una vez que se identifica un prefijo incorrecto, se realiza un desplazamiento a un estado que no es de aceptación y se permanece en el mismo.

Consideremos el lenguaje $(ab)^*$. En este caso se acepta la cadena vacía. Por lo tanto, el estado inicial es también de aceptación. El diagrama de transición se muestra en la Figura 3

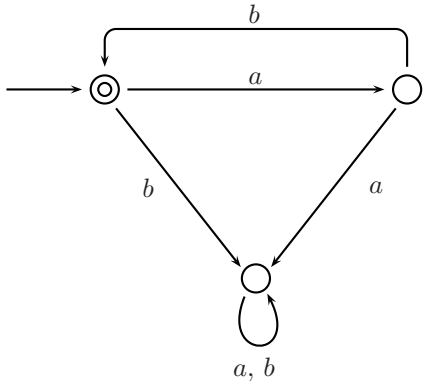


Figura 3:

Podemos representar el diagrama por medio de una tabla que indica el siguiente estado al cual desplazarse, dado un estado y un símbolo de entrada (Figura 4).

Obsérvese que la tabla para nuestro diagrama de transición tiene, para cada par estado-entrada, un único estado siguiente. De este modo para cada estado y símbolo de entrada, se puede determinar el siguiente estado. Se puede pensar que sean acciones de una máquina. dicha máquina se denomina *autómata finito*, una computadora ideal. El autómata finito se define en términos de sus estados, la entrada que acepta y su reacción ante la misma. Hay autómatas de dos tipos, deterministas y no deterministas. El autómata que corresponde a la figura 4 es determinista.

Entrada\ Salida	a	b
q_0	q_1	q_2
q_1	q_2	q_0
q_2	q_2	q_2

Figura 4:

Formalmente, un *autómata finito determinista* M es una colección de cinco elementos.

1. Un alfabeto de entrada Σ.
2. Una colección finita de estados Q.
3. Un estado inicial S.
4. Una colección de estados finales o de aceptación.

5. Una función $\delta : Q \times \Sigma \to Q$ que determina el único estado siguiente para el par(q_i, σ) correspondiente al estado actual y la entrada.

Generalmente el término *autómata finito determinista* se abrevia AFD.

Ejercicio 4

(a) *Obtener la expresión regular que representa al lenguaje formado por todas las cadenas sobre* $\{a, b\}$ *que tienen un número par de bes. Construir el diagrama de transición para este lenguaje.*

(b) *Construir el diagrama de transición para el lenguaje dado por* $c^*(a \cup bc^*)^*$*.Convertir el diagrama en una tabla, etiquetando los estados* q_0, $q_1, \ldots$.

3.1. AFD y Lenguajes

Si M es un AFD, entonces *el lenguaje aceptado por* M es

$$L(M) = \{w \in \Sigma^* \mid w \text{ es aceptada por } M\}$$

Así, $L(M)$ es el conjunto de cadenas (palabras) que hacen que M pase de su estado inicial a un estado de aceptación.

Diremos que dos AFD M_1 y M_2 son *equivalentes* si $L(M_1) = L(M_2)$. Por ejemplo, sean M_1 y M_2 sobre el alfabeto $\Sigma = \{a, b\}$, representados por los siguiente diagramas de transiciones. Ambos aceptan el lenguaje a^+ y, en conse-

Figura 5:

cuencia son equivalentes.

Ejercicio 5 *Construir los AFD que aceptan cada uno de estos lenguajes sobre* $\{a, b\}$*:*

(a) $\{w \mid$ *toda a de w está entre dos bes*$\}$

(b) $\{w \mid$ *w contiene la subpalabra abab*$\}$

(c) $\{w \mid$ *w no contiene ninguna de las subpalabras aa o bb*$\}$

(d) $\{w \mid$ *w contiene un número impar de aes y un número par de bes*$\}$

(e) $\{w \mid$ *w tiene ab y ba como subpalabras*$\}$

3.2. Autómata Finito No Determinista

Si se permite que desde un estado se realicen cero, una o más transiciones mediante el mismo símbolo de entrada, se dice que el autómata finito es *no determinista*. A veces es más conveniente diseñar autómatas finitos no determinista (AFN) en lugar de deterministas.

Un *autómata finito no determinista* es una colección de cinco objetos $(Q, \Sigma, S, F, \Delta)$, donde

1. Una colección finita de estados Q.
2. Un alfabeto de entrada Σ.
3. Un estado inicial S.
4. Una colección de estados finales o de aceptación F.
5. Una función $\Delta : Q \times \Sigma \to \mathcal{P}(\mathcal{Q})$ que determina un subconjunto de Q para el par(q_i, σ) correspondiente al estado actual y la entrada. $\mathcal{P}(\mathcal{Q})$ son los subconjuntos de Q.

Decimos que una cadena w es aceptada por un autómata finito no determinista M, sin M pasa de su estado inicial a un estado de aceptación o final al recorrer w. Definimos el lenguaje aceptado por M por medio de:

$$L(M) = \{w \mid w \text{ es una cadena aceptada por } M\}$$

Ejercicio 6

(a) *Construir un AFN que acepte el lenguaje* $(ab \cup aba)^*$.

(b) *Obtener un AFN (que no sea AFD) que acepte el lenguaje* $ab^* \cup ab^*a$.

3.3. Equivalencia entre AFN y AFD

Decimos que dos autómatas M y M'son equivalentes si $L(M) = L(M')$.

Teorema 2 *Para todo autómata finito no determinista M existe un autómata finito determinista M_1 tal que M y M_1 son equivalentes.*

ε-Transiciones

Estas son las transiciones de un estado en otro que no dependen de ninguna entrada. Para todo estado $q \in Q$, definimos la *ε-cerradura de q* como

$$\varepsilon - c(q) = \{p \mid p \text{ es accesible desde } q \text{ sin consumir ninguna entrada}\}$$

Para $q \in Q$ y $\sigma \in \Sigma$ se define

$$d(q, \sigma) = \{p \mid \text{ hay una transición de } q \text{ a } p \text{ etiquetada con } \sigma\}$$

Esta definición se amplia a conjuntos como sigue

$$d(\{q_{i_1}, q_{i_2}, \cdots, q_{i_n}\}, \sigma) = \bigcup_{k=1}^{n} d(q_{i_k}, \sigma)$$

3.4. Autómatas Finitos y Expresiones Regulares

Para un alfabeto Σ se pueden construir los AFN (y los AFD) que acepten palabras unitarias. Si M_1 y M_2 son AFN podemos unir M_1 y M_2 en un nuevo AFN que acepte $L(M_1) \cup L(M_2)$, añadiendo un nuevo estado inicial y dos ε-transiciones, una a cada uno de los estados iniciales anteriores de M_1 y M_2 respectivamente. Se deja al lector formalizar esta idea.

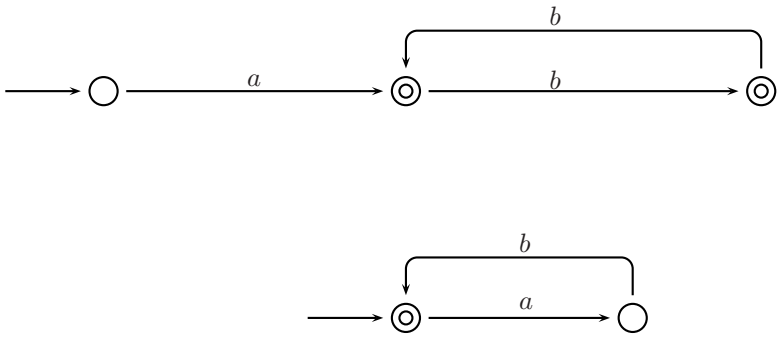


Figura 6:

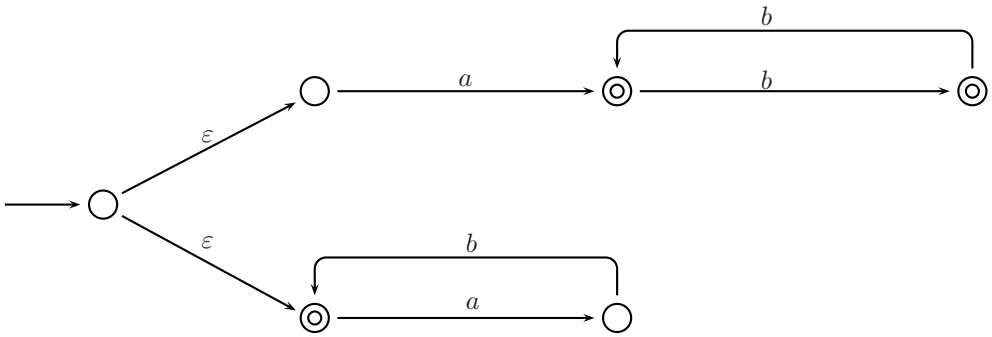


Figura 7:

Como ejemplo, los AFN de la Figura 6 aceptan ab^* y $(ab)^*$ respectivamente y el AFN de la Figura 7 acepta $ab^* \cup (ab)^*$

Si tenemos dos AFN M_1 y M_2, podemos unirlos para formar un AFN que acepte $L(M_1)L(M_2)$. Para eso, el nuevo AFN tiene como estado inicial el estado inicial de M_1 y los estados de aceptación son los estados de aceptación de M_2 y añadimos ε-transiciones de los estados de aceptación de M_1 al estado inicial de M_2. Se deja al lector formalizar esta operación.

El procedimiento para construir una AFN que acepte $L(M)^*$. Dado el AFN M, se añade un nuevo estado inicial que es al mismo tiempo el único estado de aceptación; se añaden ε transiciones del nuevo estado inicial al antiguo estado inicial y de los estados de aceptación de M al nuevo estado inicial. Se deja al lector formalizar esta operación.

Teorema 3 *El conjunto de lenguajes aceptados por un autómata finito sobre el alfabeto Σ contiene $\varnothing$ y los lenguajes unitarios $\{a\}$ para toda $a \in \Sigma$. Este conjunto es cerrado respecto a la unión, concatenación y cerradura de Kleene.*

Lema 1 *Sea M un autómata finito. Entonces existe una expresión regular r para la cual $L(r) = L(M)$.*

Teorema 4 (Kleene) *Un lenguaje es regular si y sólo si es aceptado por un autómata finito.*

Ejercicio 7

1. *Obtener un AFN que acepte* $(a \cup b)^* \cup (aba)^*$
2. *Obtener un AFN para* $(ab)^*$ *a partir de los AFN que aceptan* $\{a\}$ *y* $\{b\}$.
3. *Obtener un AFN para*$(aa \cup b)^*(bb \cup a)^*$ *a partir de los AFN que aceptan* $\{a\}$ *y* $\{b\}$.
4. *Obtener una expresión regular para el AFD de la Figura 8*

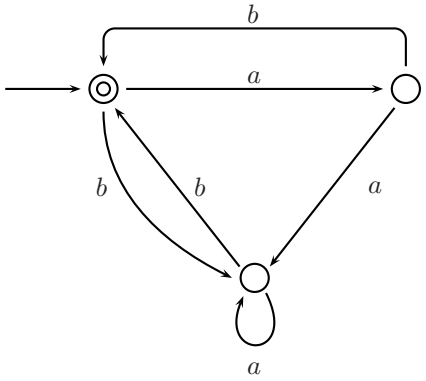
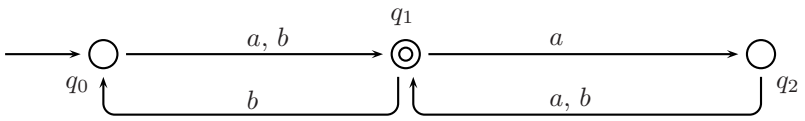


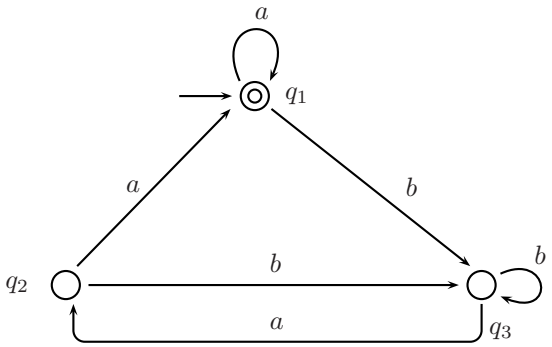
Figura 8:

(5) Obtener una expresión regular para los lenguajes aceptados por cada uno de los autómatas dados y las funciones de transición.

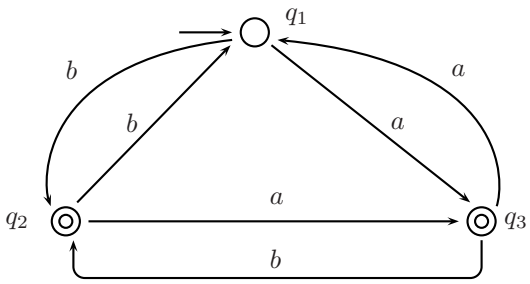
a.)



b.)



c.)



Referencias

[1] J. Glenn Brookshear. Theory of Computation. Formal Languages, Automata, and Complexity The Benjamin/Cummings Publishing Company, Inc. 1989

[2] J. E. Hopcroft, R. Motwani, J. D. Ullman. Introducción a la Teoría de Autómatas, Lenguajes y Computación Addison Wesley Longman, Pearson Education Company, Segunda Edición 2001.

[3] John C. Martin. Introduction to Languages and the theory of Computation *WCB/McGraw-Hill*, Second Edition. 1996.

[4] Michael Sipser. Introduction to the theory of Computation PWS Publishing Company, 1997.

[5] Dean Kelly.Teoría de Autómatas y Lenguajes Formales Prentice-Hall, 1998.

[6] Pedro García, Tomás Perez, etc.Teoría de Autómatas y Lenguajes Formales. Alfaomega Grupo Editor. 2001.