DIVERSAS REPRESENTACIONES DE UN AUTÓMATA FINITO DETERMINISTA

DIVERSE REPRESENTATIONS OF A DETERMINISTIC FINITE AUTOMATON

Humberto Robles Guzmán[1]

RESUMEN

En este artículo se describen las diversas representaciones que se pueden realizar para un autómata finito determinista (AFD), además, se describe una nueva representación del autómata y en base a ello, un nuevo procedimiento para saber si una cadena de caracteres *w* es aceptada por el AFD.

Palabras clave: autómata, determinista.

ABSTRACT

This article describes the diverse representations that can be performed for a deterministic finite automaton (DFA), in addition, a new representation of the automaton is described and based on this, a new procedure to know if a string w is accepted by DFA.

Keywords: automaton, deterministic.

1. INTRODUCCIÓN

El tipo de máquina que se aborda en este artículo, es una abstracción matemática que considera únicamente los aspectos referentes a las secuencias de eventos que ocurren con diversas entradas, sin tomar en cuenta la forma de la máquina, ni dimensiones o sus tipos de movimientos.

Un autómata finito o una máquina de estados finitos, es un modelo matemático de un sistema, con entradas y salidas discretas. Consiste de un conjunto finito de estados y un conjunto de transiciones de estado a estado que ocurren sobre los símbolos de entrada elegidos a partir de un alfabeto de entrada Σ.

El estado resume la información actual en la que se encuentra el autómata referente a entradas anteriores, lo cual se necesita para determinar el comportamiento del sistema con las entradas subsiguientes. Por ejemplo, una máquina expendedora de refrescos enlatados comienza con un estado inicial de cero pesos, y a medida que se le insertan diversas monedas, pasa por diversos estados acorde a la cantidad de dinero que se le vaya insertando. La máquina al estar en un estado, por ejemplo $5.00, sabe la cantidad de dinero que tiene, no importando los diversos estados que recorrió. Al seguir insertando monedas, se llega a un estado final donde sabe que tiene el dinero justo para enviar una unidad de refresco, y empezar nuevamente en su estado inicial con cero pesos.

[1] Maestría en Ciencias en Matemáticas y su Didáctica.

Antes de analizar cada una de las representaciones de un AFD, se revisarán algunos conceptos fundamentales para lograr una mejor comprensión del tema.

1.1 CONJUNTOS.

Conjunto: Es cualquier colección bien definida de objetos llamados elementos o miembros del conjunto.

Algunas de las operaciones entre conjuntos son:

Si A y B son conjuntos, se define su ***unión*** como el conjunto que tiene todos los elementos que pertenecen a A o B y se define como:

$$A \cup B = \{x | x \in A \ \ o \ \ x \in B\}$$

Obsérvese que $x \in A \cup B \ \ si \ \ x \in A \ \ \ o \ \ \ x \in B \ \ \ o \ \ \ x \ pertenece \ a \ ambos.$

Si A y B son conjuntos, su ***intersección*** se define como el conjunto que contiene a todos los elementos que pertenecen tanto a A como a B y se define como:

$$A \cap B = \{x | x \in A \ \ \ y \ \ \ x \in B\}$$

A dos conjuntos que no tienen elementos comunes, se les llama conjuntos ***disjuntos.***

Si U es un conjunto universal y contiene a A, entonces U – A se le llama complemento de A y se indica como: $\bar{A} = \{x | x \notin A\}$.

1.2 STRING, ALFABETOS Y LENGUAJES.

Un '***símbolo***' es una entidad abstracta que no se define formalmente, justo como 'punto' no se define en geometría. Un ***string*** (cadena de caracteres o palabra) es una secuencia finita de símbolos yuxtaposicionados. Por ejemplo, a, b, c son símbolos y $abcb$ es un string. La longitud de un string w, representada por $|w|$, es el número de símbolos que componen al string. Un string vacío, se representa por ϵ, el cual es un string que consiste de cero símbolos.

La ***concatenación*** de dos strings es el string formado escribiendo el primero, seguido del segundo, sin ningún espacio en blanco. Por ejemplo, concatenar *perro* y *casa* obtenemos *perrocasa*.

El string vacío es el elemento identidad para el operador concatenación, ya que $\epsilon w = w\epsilon = w$.

Un ***alfabeto*** es un conjunto finito de símbolos. Un ***lenguaje*** es un conjunto de strings compuestos por símbolos de algún alfabeto.

Otro lenguaje es el conjunto de todos los **strings sobre algún alfabeto fijo Σ**. Este lenguaje se denota por $\boldsymbol{\Sigma}^*$. Por ejemplo, si $\Sigma = \{a, b\}$, entonces $\Sigma^* = \{\epsilon, a, \ b, \ aa, \ ab, \ \ ba, \ \ bb, \ \ \cdots\}$

1.3 MATRICES.

Una matriz es un arreglo rectangular de números o símbolos dispuestos en *m renglones* y *n columnas*.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Se dirá que A es una matriz $m \times n$. Si $m = n$, se dirá que A es una matriz cuadrada de orden 'n' y que los elementos $a_{11}, a_{22}, \cdots, a_{nn}$ forman la diagonal principal de A.

Si $A = [a_{ij}]$ es una matriz $m \times p$ y $B = [b_{ij}]$ es una matriz $p \times n$, entonces el producto de A y B, que se escribe AB, es la matriz $C = [c_{ij}]$ $m \times n$ definida por:

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{ip}b_{pj} \qquad (1 \leq i \leq m, \quad 1 \leq j \leq n)$$

Se ve que los elementos $i, j - \acute{e}simos$ en la matriz producto se calcula sumando todos los productos obtenidos de la multiplicación de cada componente en el renglón $i - \acute{e}simo$ de A por la componente correspondiente de la $j - \acute{e}sima$ columna de B.

1.4 PRODUCTO CARTESIANO.

Si A y B son dos conjuntos no vacíos, se define el producto cartesiano $A \times B$ como el conjunto de pares ordenados (a, b) donde $a \in A \;\; y \;\; b \in B$. Por tanto,

$$A \times B = \{(a, b) |\, a \in A \quad y \quad b \in B\}$$

Por ejemplo, sea $A = \{1, 2, 3\} \quad y \quad B = \{r, s\}$. Entonces

$$A \times B = \{(1, r), (1, s), (2, r), (2, s), (3, r), (3, s)\}$$

Se puede ver que los elementos de $A \times B$ se pueden acomodar adecuadamente en un arreglo tabular:

B

A	r	s
1	(1, r)	(1, s)
2	(2, r)	(2, s)
3	(3, r)	(3, s)

1.5 RELACIÓN.

Sean A y B conjuntos no vacíos. Una relación $\mathcal{R}$ de A en B es un subconjunto de $A \times B$. Si $\mathcal{R} \subseteq A \times B \ y \ (a,b) \in \mathcal{R}$, se dice que a está relacionada con b por $\mathcal{R}$ y se escribirá $a\mathcal{R}b$. Si a no está relacionada con b, se escribirá $a\overline{\mathcal{R}}b$.

Si A y B son conjuntos finitos, con m y n elementos respectivamente, y si $\mathcal{R}$ es una relación de A en B, es posible representar a $\mathcal{R}$ con una matriz $M_{\mathcal{R}} = [m_{ij}]$, cuando se define como sigue:

$$m_{ij} = \begin{cases} 1 & si \quad (a_i, b_j) \in \mathcal{R} \\ 0 & si \quad (a_i, b_j) \notin \mathcal{R} \end{cases}$$

Por ejemplo, la matriz $M_{\mathcal{R}}$ para los conjuntos $A = \{1,2,3\} \ y \ B = \{r,s\}$ y la relación $\mathcal{R} = \{(1,r),(2,s),(3,r)\}$ es:

$$M_{\mathcal{R}} = \begin{matrix} \\ 1 \\ 2 \\ 3 \end{matrix} \begin{matrix} r & s \end{matrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

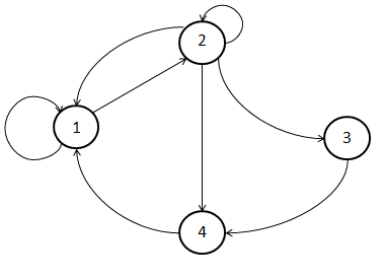
1.6 GRAFOS DIRIGIDOS.

Si A es un conjunto finito y $\mathcal{R}$ es una relación en A, se podrá representar a $\mathcal{R}$ mediante una gráfica. Dibújese un pequeño círculo para cada elemento de A y etiquétese el círculo con el elemento correspondiente de A. Estos círculos son llamados vértices. Dibújese una línea con dirección, llamada arista, del vértice a_i al vértice a_j si y sólo si $a_i \ \mathcal{R} \ a_j$. La representación gráfica resultante de $\mathcal{R}$ se llama un **grafo dirigido** de $\mathcal{R}$.

Por ejemplo, sea

$A = \{1,2,3,4\}$ y $\mathcal{R} = \{(1,1),(1,2),(2,1),(2,2),(2,3),(2,4),(3,4),(4,1)\}$

El grafo dirigido de $\mathcal{R}$ es:



2. REPRESENTACIONES DE UN AFD

Para realizar el diseño o análisis de un autómata finito determinista, es conveniente comprender y llevar a cabo distintas formas de representarlo y de esta forma tener una mayor comprensión del mismo.

Para tal fin, se analizan las diversas formas de representar un autómata finito determinista.

2.1 REPRESENTACIÓN FORMAL.

Formalmente denotamos a un autómata finito determinista mediante una 5-upla $(Q,\ \Sigma,\ \delta,\ q_0,\ F)$, donde Q es el conjunto finito de estados, Σ es el alfabeto finito de entrada, $q_0 \in Q$ es el estado inicial, $F \subseteq Q$ es el conjunto de estados finales, y δ es la función de transición de transformar $Q \times \Sigma\ a\ Q$. Esto es, $\delta(q, \sigma)$ es un estado para cada estado q y el símbolo de entrada $\sigma \in \Sigma$.

Por ejemplo, diseñar un autómata finito determinista M que acepte el lenguaje con el alfabeto $\Sigma = \{a, b\}$, tal que únicamente acepte cadenas de caracteres que no contengan tres 'b' consecutivas, es decir:

$$L(M) = \{w \in \{a,b\}^* |\ w\ no\ contenga\ 3\ b\ (bbb) consecutivas\}.$$

Sea $M = (Q, \Sigma, \delta, q_0, F)$, donde:

$Q = \{q_0, q_1, q_2, q_3\}$,

$\Sigma = \{a, b\}$,

$q_0\ es\ el\ estado\ inicial$,

$F = \{q_0, q_1, q_2\}$,

y δ está dada por la siguiente tabla:

Q	σ	$\delta(q, \sigma)$
q_0	a	q_0
q_0	b	q_1
q_1	a	q_0
q_1	b	q_2
q_2	a	q_0

q_2	b	q_3
q_3	a	q_3
q_3	b	q_3

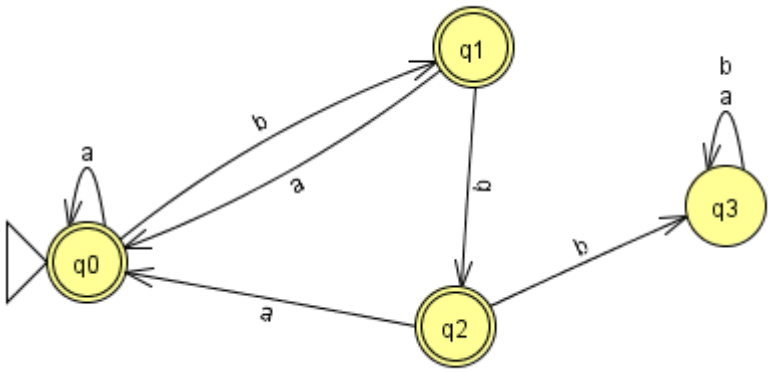
Esta tabla nos indica las transiciones de estado a estado según la entrada al sistema, por ejemplo, $(q_0, b) \rightarrow q_1$, indica que si el autómata está en el estado q_0 y llega como entrada un carácter b, entonces se cambia al estado q_1 .

2.2 REPRESENTACIÓN A TRAVÉS DE UN DIAGRAMA DE ESTADOS O DE TRANSICIÓN.

Una manera de representar gráficamente a un autómata, es a través de un diagrama de estados. Es importante señalar que en el diseño de un autómata finito determinista es preferible iniciarlo con el desarrollo de un diagrama de estados y posteriormente, encontrar su representación formal.

El diagrama de estados es una gráfica dirigida, con cierta información adicional incorporada en el gráfico. La gráfica está compuesta de vértices o nodos, los cuales corresponden a los estados del autómata. Cada estado se representa con un círculo, y si es un estado de aceptación, se representa con doble círculo. Al estado inicial se le antepone el símbolo $\triangleright$. Si existe una transición del estado q_i al estado q_k sobre una entrada $\sigma \in \Sigma$, entonces se traza una flecha etiquetada como σ que va del nodo q_i al nodo q_k. El autómata finito determinista acepta una palabra $w \in \Sigma^*$ si la secuencia de transiciones correspondiente a los símbolos de la palabra empieza desde el estado inicial y se llega a un estado de aceptación.

Por ejemplo, continuando con el autómata finito determinista M que acepte el lenguaje $L(M) = \{w \in \{a, b\}^* | \, w \, no \, contenga \, 3 \, b \, (bbb) consecutivas\}$, construimos su diagrama de estados a partir de las reglas de transición presentadas en la tabla anterior:



Supongamos que $w = aabbaaabaab$, siguiendo la entrada w a través de nuestro autómata, partiendo del estado inicial q_0 observamos que el autómata va cambiando o conmutando por los estados $q_0q_0q_1q_2q_0q_0q_0q_1q_0q_0q_1$, llegando al final a un estado de aceptación, por lo que la palabra w es aceptada por el autómata.

En el diseño de un autómata finito determinista, lo difícil es el crear su diagrama de estados, ya que el diagrama es el algoritmo que permite dar solución al problema planteado. Recordar que un algoritmo es una secuencia lógica y ordenada de acciones para resolver un problema.

2.3 REPRESENTACIÓN A TRAVÉS DE UNA TABLA DE TRANSICIÓN.

Una manera alternativa de representar un autómata finito determinista es mediante su tabla de transición, representada de la siguiente manera:

	Entradas (elementos del alfabeto)			
Estados	σ_1	σ_2	$\cdots$	σ_n
q_1	q_r	q_s	$\cdots$	q_t
q_2	q_w	q_x	$\cdots$	q_y
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
q_m	q_i	q_j	$\cdots$	q_k

En esta representación, se tendrán *n+1* columnas y *m* renglones, en donde la primera columna representa los estados y las *n* columnas siguientes estarán etiquetadas por cada uno de los símbolos de nuestro alfabeto de entrada. Los *m* renglones contendrán en la columna de estados, los *m* estados que conforman el autómata. El elemento $q_{i,k}$ de la matriz es uno de los elementos del conjunto de estados del autómata $q_{i,k} \in$ Q, lo que indica que si se parte del estado i, y se lee un carácter del alfabeto $\sigma_j \in \Sigma$, propicia pasar del estado de entrada i al estado de salida k.

Considerando el ejemplo del autómata finito determinista M que acepte cadenas pertenecientes al lenguaje

$$(M) = \{w \in \{a, b\}^* | \, w \; no \; contenga \; 3 \; b \; (bbb) consecutivas\},$$

se construye su tabla de transición:

Estados	a	b
q_0	q_0	q_1
q_1	q_0	q_2
q_2	q_0	q_3
q_3	q_3	q_3

A través de la tabla se puede observar que si se inicia en el estado q_2 y el elemento de entrada es el carácter a, entonces el siguiente estado del autómata es q_0.

Además, esta representación facilita el diseño de un algoritmo para programar el autómata por medio de algún lenguaje de programación como Java, C++, Python, entre otros.

El **algoritmo** en seudocódigo de cualquier autómata finito determinista, lo podemos generalizar como sigue:

Inicio
    Se inicializa el estado actual (q= 0)
    Leer cadena de entrada (*w*)
    long=tamaño de la cadena de entrada (*w*)
    For (i=0; i<long; i++)
        Leer elemento *w(i)*
        Si el estado actual es:
        q=0
            Si el carácter de entrada es:
            $w(i) = \sigma_1; \quad (actualiza\ el\ estado) \quad q = k_i$
            $w(i) = \sigma_2; \quad (actualiza\ el\ estado) \quad q = k_j$
            ……
            $w(i) = \sigma_n; \quad (actualiza\ el\ estado) \quad q = k_m$
            Sino error (carácter no pertenece al alfabeto)
        q=1
            Si el carácter de entrada es:
            $w(i) = \sigma_1; \quad (actualiza\ el\ estado) \quad q = k_i$
            $w(i) = \sigma_2; \quad (actualiza\ el\ estado) \quad q = k_j$
            ……
            $w(i) = \sigma_n; \quad (actualiza\ el\ estado) \quad q = k_m$
            Sino error (carácter no pertenece al alfabeto)
        ……
        q=m
            Si el carácter de entrada es:
            $w(i) = \sigma_1; \quad (actualiza\ el\ estado) \quad q = k_i$
            $w(i) = \sigma_2; \quad (actualiza\ el\ estado) \quad q = k_j$
            ……
            $w(i) = \sigma_n; \quad (actualiza\ el\ estado) \quad q = k_m$
            Sino error (carácter no pertenece al alfabeto)

        Sino error (estado no valido)
    Siguiente i
    Si (q esta entre los estados de aceptación)
        Escribir: Palabra aceptada por el autómata
    Sino
        Escribir: Palabra no aceptada por el autómata
    Fin-Si

Fin del Algoritmo

2.4 REPRESENTACIÓN A TRAVÉS DE UNA MATRIZ DE TRANSICIÓN.

Otra forma interesante de representar un autómata finito determinista, es mediante un matriz de transición, en la cual cada renglón se considera un estado de entrada y cada columna un estado de salida, por lo que es una matriz cuadrada de orden igual al número de estados del autómata. El elemento $a_{i,k}$ de la matriz es un carácter del alfabeto de entrada $a_{i,k} \in \Sigma$, si propicia pasar del estado de entrada i al estado de salida k, en caso contrario, su valor será 0.

Entrada/Salida	q_0	q_1	$\cdots$	q_n
q_0	$a_{0,0}$	$a_{0,1}$	$\cdots$	$a_{0,n}$
q_1	$a_{1,0}$	$a_{1,1}$	$\cdots$	$a_{1,n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
q_n	$a_{n,0}$	$a_{n,1}$	$\cdots$	$a_{n,n}$

Para el ejemplo propuesto del autómata finito determinista M que acepte cadenas pertenecientes al lenguaje

$$L(M) = \{w \in \{a,b\}^* | \, w \; no \; contenga \; 3 \; b \; (bbb) consecutivas\},$$

se construye su matriz de transición:

$$A = \begin{bmatrix} a & b & 0 & 0 \\ a & 0 & b & 0 \\ a & 0 & 0 & b \\ 0 & 0 & 0 & a,b \end{bmatrix}$$

Se observa en esta matriz de transición, por ejemplo, considerando el primer renglón, que si se inicia en el estado q_0 y se lee el carácter a, la transición llevará al autómata al estado q_0, si se inicia en el estado q_0 y se lee el carácter b, la transición enviará al autómata al estado q_1, los demás elementos del renglón son cero, ya que no existe transición alguna de q_0 a los demás estados.

Con esta representación matricial, podemos utilizar la operación de multiplicación entre matrices para encontrar la matriz de $A^n = A * A^{n-1} = A^{n-1} * A$, la cual dará la cantidad y la ruta de los caminos que llegan de q_i a q_k con n elementos validos de la cadena de entrada. Es importante aclarar que debido a que los elementos de la matriz no son números, el producto entre los elementos de la matriz es un producto cartesiano, el cual da como resultado un conjunto y el operador '+' es la operación de unión entre conjuntos.

Por ejemplo, $A^2 = A * A = \begin{bmatrix} a & b & 0 & 0 \\ a & 0 & b & 0 \\ a & 0 & 0 & b \\ 0 & 0 & 0 & a,b \end{bmatrix} * \begin{bmatrix} a & b & 0 & 0 \\ a & 0 & b & 0 \\ a & 0 & 0 & b \\ 0 & 0 & 0 & a,b \end{bmatrix} =$

$aa + ba$	ab	bb	0
$aa + ba$	ab	0	bb
aa	ab	0	$ba + bb$
0	0	0	$aa + ab + ba + bb$

En esta matriz resultante, considerando el segundo renglón, se observa que para llegar de q_1 a q_0 con dos elementos de entrada, lo podemos hacer a través de dos caminos, los cuales son aa o bien ba; para llegar de q_1 a q_3 solo existe un camino, el cual es bb. También se observa en el cuarto renglón que si se inicia en q_3 no se puede llegar a ningún otro estado que no sea q_3. Todo ello se puede constatar revisando el diagrama de estados del autómata.

En el ejercicio desarrollado, el elemento $a_{4,4}$ de la matriz A^2 se obtiene realizando el producto cartesiano $\{a,b\} \times \{a,b\} = \{aa, ab, ba, bb\}$, el cual indica que para llegar al estado q_3 partiendo del estado q_3, considerando únicamente dos elementos del alfabeto de entrada, se puede realizar a través de las cadenas $aa, ab, ba,$ ó bb.

2.5 REPRESENTACIÓN A TRAVÉS DE UNA MATRIZ DE TRANSICIÓN MODIFICADA.

Una nueva forma dinámica de representar a un autómata determinista, es lo que he llamado matriz de transición modificada. Esta forma de representar al autómata, no la he encontrado en ningún documento o libro sobre autómatas, por lo que la he denominado matriz de transición modificada.

En esta nueva representación se construyen 'n' matrices cuadradas, una matriz por cada elemento del alfabeto del autómata.

Cada matriz es de orden m, en donde m es la cantidad de estados que forman el autómata. Cada renglón representa un estado de entrada y cada columna, representa un estado de salida, por lo que la forma general de cada matriz es la siguiente:

Entrada/Salida	q_0	q_1	$\cdots$	q_n
q_0	$a_{0,0}$	$a_{0,1}$	$\cdots$	$a_{0,n}$
q_1	$a_{1,0}$	$a_{1,1}$	$\cdots$	$a_{1,n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
q_n	$a_{n,0}$	$a_{n,1}$	$\cdots$	$a_{n,n}$

El elemento $a_{i,k}$ de la matriz es un 1 si con el elemento de entrada $\sigma \in \Sigma$ propicia pasar del estado de entrada i al estado de salida k, en caso contrario, su valor será 0.

$$a_{i,k} = \begin{cases} 1 & si \quad (q_i, \sigma) \implies q_k \\ 0 & si \quad (q_i, \sigma) \nRightarrow q_k \end{cases}$$

En este tipo de matriz, cada renglón solo contiene un elemento con el valor de uno y los demás elementos contienen el valor cero, ya que al ser un autómata determinista, cada estado q_i con una entrada σ únicamente puede cambiarse a un estado q_k.

Al mismo tiempo, como esta representación matricial tiene por renglones los estados de entrada y como columnas los estados de salida, se puede efectuar la multiplicación de matrices, cuya operación permite transformar los estados de salida de la primer matriz en estados de entrada de la segunda matriz, y el resultado del producto da la ventaja de ver la transición de iniciar en el estado q_i y llegar al estado de salida q_k a través de las entradas $\sigma_j \; y \; \sigma_n$. Por supuesto, la matriz resultante del producto, también tiene un único uno en cada renglón, ya que las transiciones llegan a un determinado y único estado.

Este producto de matrices es lo que hace que esta representación sea muy dinámica, ya que con esta simple operación se puede saber el estado final al que llega el autómata al analizar una determinada palabra de entrada y con ello saber si dicha palabra pertenece al lenguaje que acepta el autómata finito determinista M.

Para el ejemplo propuesto del autómata finito determinista M que acepte el lenguaje $L(M) = \{w \in \{a,b\}^* \mid w \; no \; contenga \; 3 \; b \; (bbb) consecutivas\}$, se construyen dos matrices de transición modificada, una para cada elemento del alfabeto de entrada $\Sigma = \{a, b\}$. Se llamará A a la matriz de transición modificada para la entrada a, y B a la matriz para la entrada b:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Obsérvese que cada renglón de la matriz A y de la matriz B, tiene un solo elemento con valor de uno y los demás elementos tienen el valor de cero.

Con esta representación, se puede determinar si una cadena de entrada w pertenece al lenguaje $L(M)$, únicamente multiplicando las matrices $A \; o \; B$, según corresponda a la entrada de la cadena.

Por ejemplo, si $w = abba$, se efectúa el producto de matrices de la siguiente manera: $A * B * B * A$, obteniéndose el siguiente resultado:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

En este ejemplo, la matriz resultante indica que partiendo del estado $q_0, q_1\ o\ q_2$, siempre se llega al estado q_0, el cual es un estado de aceptación, y si se inicia en el estado q_3, se termina en q_3, cuyo estado es de no aceptación.

Lo ventajoso de esta representación es que se puede saber a qué estado llega el autómata determinista conociendo el estado de inicio y la palabra a analizar, únicamente realizando el producto entre las diversas matrices de la representación matricial dinámica del autómata.

Por ejemplo, para el autómata propuesto, si se considera la palabra $w = ababbba$ y efectuando el producto de matrices $A * B * A * B * B * B * A$ se obtiene la matriz

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La cual indica que no importa de qué estado se inicie, siempre se termina en el estado q_3, el cual es un estado de no aceptación, por lo que la palabra $w = ababbba$ no es aceptada por el autómata, ya que al examinar la palabra, se reconoce que tiene la sub-cadena bbb (tres b).

Como se observa, con esta representación únicamente se realiza la multiplicación entre las matrices correspondientes a cada elemento del alfabeto de entrada y la matriz resultante da el estado final a partir de un estado de inicio.

2.6 REPRESENTACIÓN A TRAVÉS DE UNA EXPRESIÓN REGULAR.

Un autómata finito determinista especifica un lenguaje, dando una forma de juzgar si una palabra de entrada w es aceptada o no por el autómata. Una expresión regular se obtiene mediante la realización repetida de tres clases de operaciones realizadas a los símbolos de un alfabeto de entrada, lo que resulta de expresar un lenguaje. Así, una expresión regular expresa directamente la forma de las cadenas de caracteres que pueden ser obtenidas por los símbolos del alfabeto de algún lenguaje.

Una expresión regular tiene tres clases de operaciones expresadas como $+, \cdot, *$, las cuales, respectivamente, representan la unión, la concatenación y la concatenación aplicada repetidamente un número arbitrario de veces. En particular, la operación $*$ es llamada estrella de Kleene.

Definición: Una expresión regular sobre un alfabeto $\Sigma = \{a_1, a_2, \cdots, a_k\}$ es definida como sigue:

a) Cada uno de $a_1, a_2, \cdots, a_k, \in, \Phi$ es una expresión regular.

b) Si r y s son expresiones regulares, también lo son $(r + s), (r \cdot s)\ y\ (r^*)$.

La expresión regular equivalente al autómata finito determinista M que acepte el lenguaje $L(M) = \{w \in \{a, b\}^* | \ w\ no\ contenga\ 3\ b\ (bbb) consecutivas\}$, que es el ejemplo propuesto en el documento, está dada por:

$$(a + ba + bba)^*(\in + b + bb)$$

Se puede reconocer del lado izquierdo de la expresión que podemos aceptar palabras con las cadenas: $\in, a, aa, aaa, \cdots, ba, baba, \cdots, bba, bbabba, \cdots$, y diversas combinaciones de dichas cadenas. Una característica de estas cadenas de caracteres es que todas terminan con el carácter a y ninguna tiene 3 b $(bbb)consecutivas$. Debido a que el lenguaje del autómata también acepta palabras que terminen con el carácter b, empleamos la operación de concatenación para que se acepten palabras que terminen en b o en bb, lo que es representado por el lado derecho de la expresión regular.

3. CONCLUSIÓN

La representación del autómata finito determinista por medio de una matriz de transición modificada nos permite ver de una manera simple si una palabra w es aceptada por el autómata, ya que a través de multiplicación de matrices se puede obtener de manera clara la respuesta. Además, la multiplicación de matrices es de fácil desarrollo en cualquier lenguaje de programación, por lo que su puesta en práctica es accesible para aquellas personas con conocimientos básicos de programación, igualmente, se puede emplear algún software matemático como Matlab, Maple o Sage, entre otros, los cuales realizan la multiplicación de matrices de manera sencilla.

4. APÉNDICE

Programa en el lenguaje Java para el autómata finito determinista M que acepte el lenguaje $L(M) = \{w \in \{a, b\}^* \mid w\ no\ contenga\ 3\ b\ (bbb)consecutivas\}$, que es el ejemplo propuesto en el documento.

```
import java.util.*;
import java.lang.*;
class Automata{
	private int estado;

	Automata(){
		estado=0;
	}
	public int dame_estado(){
		return estado;
	}
	public void cambia_estado(char e){
		switch (estado){
			case 0:
				switch (e){
					case 'a':
						estado=0;
						break;
					case 'b':
```

```
                                        estado=1;
                                        break;
                                default:
                                        estado=3;
                        }
                        break;
                case 1:
                        switch (e){
                                case 'a':
                                        estado=0;
                                        break;
                                case 'b':
                                        estado=2;
                                        break;
                                default:
                                        estado=3;
                        }
                        break;
                case 2:
                        switch (e){
                                case 'a':
                                        estado=0;
                                        break;
                                case 'b':
                                        estado=3;
                                        break;
                                default:
                                        estado=3;
                        }
                        break;
                case 3:
                        switch (e){
                                case 'a':
                                        estado=3;
                                        break;
                                case 'b':
                                        estado=3;
                                        break;
                                default:
                                        estado=3;
                        }
                        break;
                }
        }
}
public class Automata_01 {
```

```
   public Automata_01() {
   }

   public static void main(String[] args) {
      Automata M1=new Automata();
      Scanner sc = new Scanner(System.in);
      int estado,n,i;
      double radio;
      String cadena;
      System.out.println("Programa para determinar si una cadena pertenece a un automata");
      System.out.println("El automata acepta palabras que no contengan bbb");
      System.out.print("Introduzca la cadena a evaluar: ");
      cadena = sc.nextLine();
      n=cadena.length(); //devuelve el numero de caracteres que contiene cadena
      for(i=0;i<n;i++){
         M1.cambia_estado(cadena.charAt(i));
      }
      estado=M1.dame_estado();
      switch (estado){
                        case 0:
                        case 1:
                        case 2:
                              System.out.println("La cadena "+cadena+" es aceptada");
                              break;
                        default:
                           System.out.println("La cadena "+cadena+" NO es aceptada");
                  }

   }
}
```

5. REFERENCIA BIBLIOGRÁFICA.

Hopcroft, John E.; Ullman, Jeffrey d.; Introducción a la Teoría de Autómatas, Lenguajes y Computación. Editorial Continental., 1993.

Kolman, Bernard; Busby, Robert C.; Estructuras de matemáticas discretas para la computación. Editorial Prentice Hall, 1984.

Lewis, Harry R.; Papadimitriou, Christos H. *Elements of the theory of computation.* Prentice-Hall, Inc., 1998.