Autómatas I.

Parcial I. **Tarea I.**

Aceptadores Finitos Deterministas (DFA).

Juan Francisco Gallo Ramírez

5to Semestre UAA

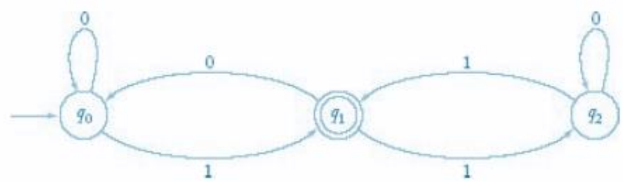
I.C.I.

Ejercicio 1.

Resolver el ejercicio 1 del libro de Peter Linz sección 2.1 (ver imagen adjunta).

1. Which of the strings 0001, 01001, 0000110 are accepted by the dfa in Figure 2.1?

Figure 2.1



Solución

Para resolver el problema planteado, utilizaremos una tabla que muestre los estados en los que se encuentra cada elemento procesado de cada cadena. Esta tabla permitirá representar el cambio de estado en función del elemento que se esté procesando en cada momento y de esta forma, poder determinar si es aceptada o no.

- `0001`. *Aceptada.*

Est. Ant.	Elemento	Est. Sig.	Descripción
q_0	0	q_0	Al ser un "*0*" y encontrarse en q_0 se permanece en este estado q_0.
q_0	0	q_0	Al ser un "*0*" y encontrarse en q_0 se permanece en este estado q_0.
q_0	0	q_0	Al ser un "*0*" y encontrarse en q_0 se permanece en este estado q_0.
q_0	1	q_1	Al ser un "*1*" y encontrarse en q_0 pasa al estado q_1 (estado final) por lo tanto, la cadena es aceptada.

- `01001`. *Aceptada.*

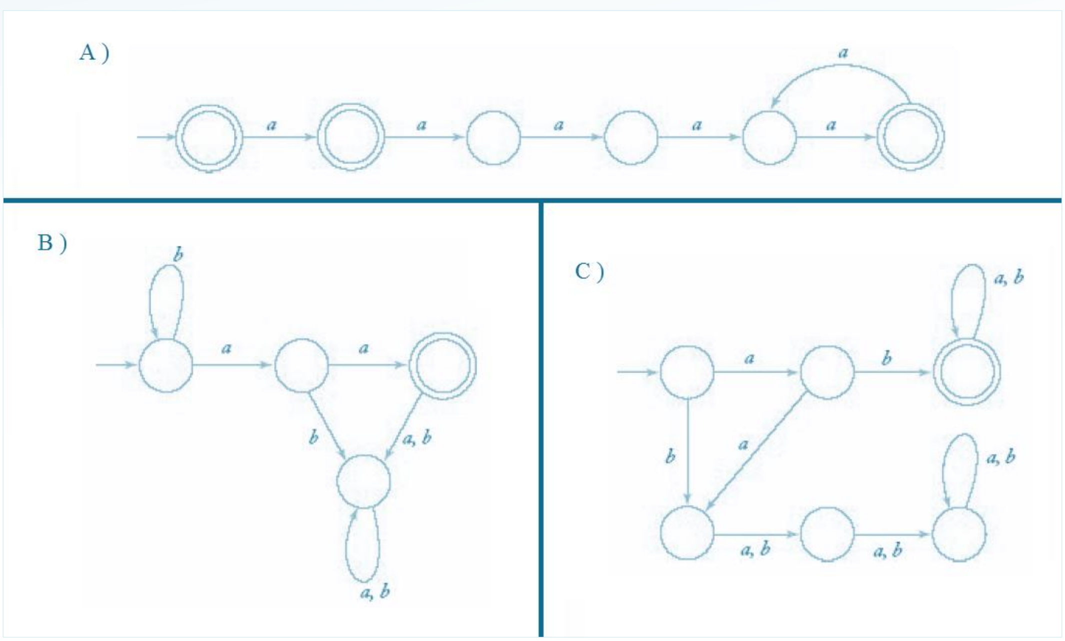
Est. Ant.	Elemento	Est. Sig.	Descripción
q_0	0	q_0	Al ser un “*0*” y encontrarse en q_0 se permanece en este estado q_0.
q_0	1	q_1	Al ser un “*1*” y encontrarse en q_0 pasa al estado q_1.
q_1	0	q_0	Al ser un “*0*” y encontrarse en q_1 pasa al estado q_0.
q_0	0	q_0	Al ser un “*0*” y encontrarse en q_0 se permanece en este estado q_0.
q_0	1	q_1	Al ser un “*1*” y encontrarse en q_0 pasa al estado q_1 (estado final) por lo tanto, la cadena es aceptada.

- `00001100`. *Rechazada.*

Est. Ant.	Elemento	Est. Sig.	Descripción
q_0	0	q_0	Al ser un “*0*” y encontrarse en q_0 se permanece en este estado q_0.
q_0	0	q_0	Al ser un “*0*” y encontrarse en q_0 se permanece en este estado q_0.
q_0	0	q_0	Al ser un “*0*” y encontrarse en q_0 se permanece en este estado q_0.
q_0	0	q_0	Al ser un “*0*” y encontrarse en q_0 se permanece en este estado q_0.
q_0	1	q_1	Al ser un “*1*” y encontrarse en q_0 pasa al estado q_1.
q_1	1	q_2	Al ser un “*1*” y encontrarse en q_1 pasa al estado q_2.
q_2	0	q_2	Al ser un “*0*” y encontrarse en q_2 se permanece en este estado q_2 (estado no final) por lo tanto, la cadena es rechazada.

Ejercicio 2.

Expresar con notación de conjunto los lenguajes aceptados por los DFA de la imagen Aceptadores.png

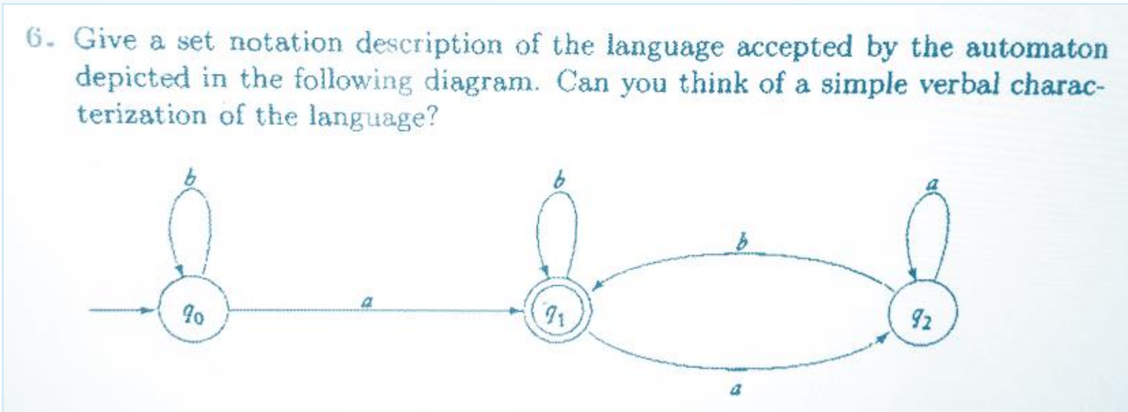


Solución

Inciso		Respuesta
A)	*Descripción del DFA.*	Inicialmente podemos observar que el DFA solo tiene transiciones desencadenadas únicamente por la letra “***a***”, por lo que su alfabeto será *{ a }*. Además, solo tiene 3 estados finales, los cuales los primeros dos solo pueden ser accedidos por: *{ λ , a }* *(“a” de longitud 0 y 1).* El ultimo estado final solo se puede llegar si se presenta una cadena de letras “***a***” con una longitud de al menos 5, y a partir de ahí, si se incrementa una “***a***” (longitud 6) retrocederá un estado no final; a su vez, si se incrementa otra “***a***” (longitud 7) volverá a dicho estado final. Esto nos siguiere que también aceptará las cadenas impares partiendo desde el número 5. Un ejemplo del lenguaje que acepta: *L = { λ, a, aaaaa, aaaaaaa, aaaaaaaaa, … }* *(“a” de longitud 0, 1, 5, 7, 9).*
	Notación del DFA.	$L = \{ a^{\mathrm{n}} : n = \{ 0, 1 \} \vee (n \geq 5 \wedge n \,\%\, 2 \neq 0) \}$
B)	*Descripción del DFA.*	Inicialmente podemos observar que el DFA solo tiene transiciones desencadenadas únicamente por las letras “***a***” y “***b***”, por lo que su alfabeto será *{ a, b }*. Solo tiene un estado final, y es únicamente accedido si se tienen una cadena de dos “***a***” y no existe una tercera ni tampoco una “***b***” posterior; sin embargo, es posible tener una cadena de “***b***” de cualquier longitud seguida por una de exactamente dos “***a***”. Se puede observar que en cualquier estado fuera del inicial, si se ingresa una letra “***b***”, caemos en un estado trampa del que no se puede salir. Esto nos sugiere que solo acepta cadenas de “***b***” de cualquier longitud antecediendo una de exactamente dos “***a***”. En lenguaje que acepta es: *L = { aa, baa, bbaa, bbbaa, bbbbaa, bbbbba, … }*
	Notación del DFA.	$L = \{ \mathrm{b}^n \mathrm{aa} : n \geq 0 \}$
C)	*Descripción del DFA.*	Inicialmente podemos observar que el DFA solo tiene transiciones desencadenadas únicamente por las letras “***a***” y “***b***”, por lo que su alfabeto será *{ a, b }*. Solo tiene un estado final, y solo puede ser accedido si se ingresa una “***a***” y posterior un “***b***”, después de ello no importará que suceda a la cadena ya que permanecerá en dicho estado final; los demás estados no son de nuestro interés analizar, ya que para que sea aceptada solo tiene que cumplir con el prefijo antes mencionado, ya que si se llega a ingresar una “***b***” como primer letra en lugar de “***a***”, o una “***a***” como segunda letra en lugar de “***b***”, se llegará a una serie de estados no finales que no tiene posibilidad de regresar al estado final. Por lo que es posible decir que el lenguaje que acepta es: *L = { ab, aba, abb, abaa, abab, abba, abbb, … }*
	Notación del DFA.	$L = \{ \mathrm{ab}w : w \in \{ \mathrm{a}, \mathrm{b} \}^* \}$

Ejercicio 3.

Resolver el ejercicio de la imagen DFA6.jpg (tomada de la tercera edición del libro de Peter Linz).


6. Give a set notation description of the language accepted by the automaton depicted in the following diagram. Can you think of a simple verbal characterization of the language?

Solución

Para resolver el problema planteado se hará un análisis en cada estado con el propósito que se logre identificar la forma en la que puede ser accesible el estado final por medio de los demás estados no finales.

Estado	Análisis
q_0	Este estado solo es accesible al inicio, se permanece en este estado si se encuentra una letra "*b*", pero una vez salido de este estado no se puede volver a acceder. La forma en la que se llega al estado final q_1 es por medio de una "*a*".
q_1	Este estado es el final, y es accesible por medio de una "*a*" desde q_0 y una "*b*" desde q_2. Además, se permanece en este estado si se encuentra una "*b*".
q_2	Este estado solo es accesible desde q_1 por medio de una letra "*a*", y se permanece en el mismo si se encuentra una "*a*". La forma de llegar al estado final q_1 es por medio de una "*b*".

- En análisis anterior nos indica que es necesario tener al menos una letra "*a*" para salir del estado inicial, nos es indiferente si hay n cantidad de letras "*b*".
- En el estado final, no importa si hay *n* cantidad de letras "*b*", permaneceremos en el estado final.
- Si en el estado final encontramos una letra "*a*", pasaremos a un estado donde solo puede regresarse al estado final por medio de una letra "*b*".

Entonces, el aceptador esta basado en un alfabeto *{ a, b }*, donde no nos interesa en que estado estemos, siempre podremos ir al estado final, por lo que podemos establecer que se trata de un sufijo. Además, es necesario al menos una letra "*a*", seguido de una cantidad cualquiera de "*b*" (de longitud desde 0 hasta ∞) para que la cadena sea aceptada, pero si se trata de un sufijo que tiene una longitud de letras "*a*" mayor a 1, es necesario al menos una letra "*b*" para llegar al estado final. Dicho esto, las notaciones que se han encontrado son:

$$L = \{\, w\mathrm{a}^n\mathrm{b}^m : w \in \{\,\mathrm{a},\mathrm{b}\,\}^*, (n = 1 \wedge m \geq 0) \vee (n \geq 1 \wedge m > 0)\}$$

$$L = \{\, w\mathrm{a}^n\mathrm{b}^m : w \in \{\,\mathrm{a},\mathrm{b}\,\}^*, n \geq 1 \wedge m > 0 \Leftrightarrow n > 1\}$$

Caracterización verbal del lenguaje:
*El autómata acepta cadenas cuyo sufijo es de al menos una "**a**" seguida de cualquier cantidad de "**b**" si solo hay una "**a**", o al menos una "**b**" si hay más de una "**a**".*

Ejercicio 4.

Resolver los ejercicios 2, 12, 13 y 11 del libro de Peter Linz sección 2.1 (ver imagen adjunta).

2. For $\Sigma = \{a,b\}$, onstruct dfa's that accept the sets consisting of
   (a) all strings with exactly one a,
   (b) all strings with at least one a,
   (c) all strings with no more than three a's,
   (d) all strings with at least one a and exactly two b's,
   (e) all the strings with exactly two a's and more than two b's.

11. Show that the language $L = \{vwv : v, w \in \{a,b\}^*, |v| = 2\}$ is regular.

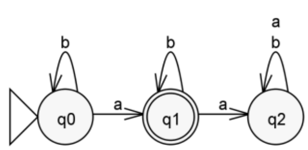
12. Show that $L = \{a^n : n \geq 4\}$ is regular.

13. Show that the language $L = \{a^n : n \geq 0, n \neq 4\}$ is regular.

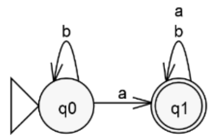
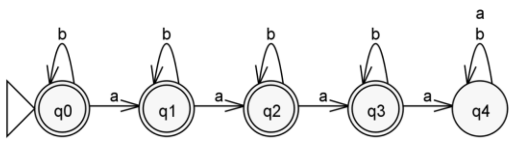
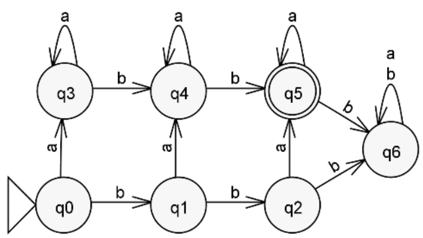
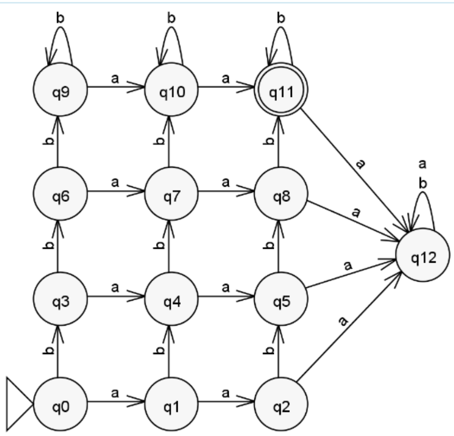
Solución

2. Construcción de DFA's

(a)



El DFA tiene un estado inicial, q_0, donde se permanece con letras "*b*" y transita al estado final q_1 al recibir una "*a*". En q_1, permanece aceptando cualquier número de "*b*", cumpliendo la condición de tener una sola "*a*". Si recibe otra "*a*", transita al estado trampa q_2, que no es final, indicando que se ha incumplido la condición de tener solo una "*a*".

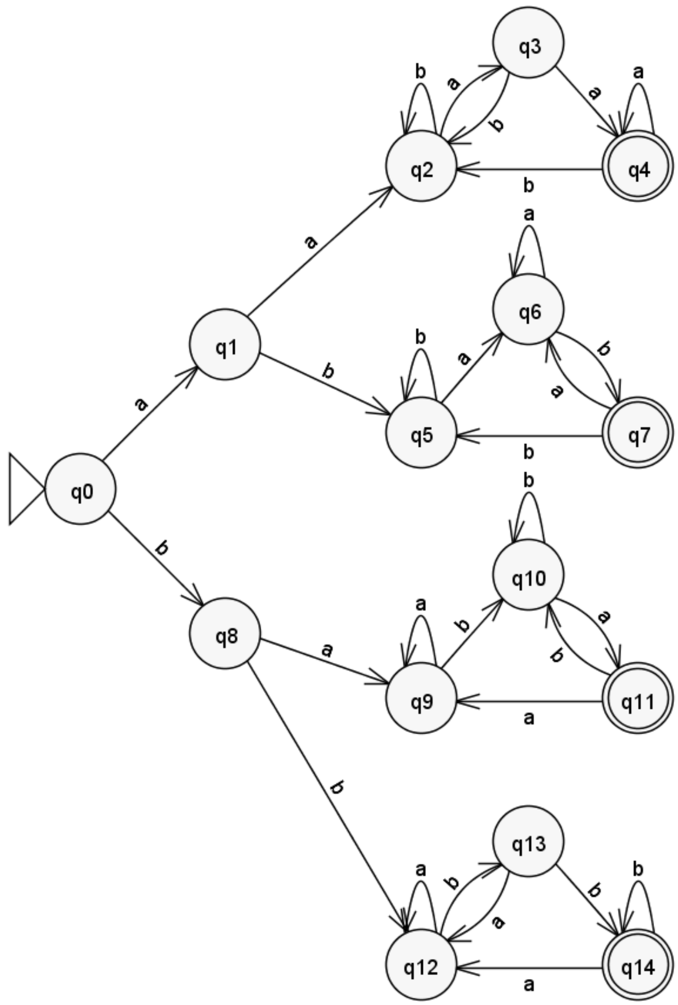
<table>
<tr>
<td>(b)</td>
<td>
</td>
<td>El DFA cuenta con solo dos estados, donde en el inicial, q₀, se permanece si se encuentran letras “b” y pasa al estado final q₁ si encuentra una letra “a”, en donde no se puede salir ya que se ha cumplido con la única condición de haber encontrado una letra “a”.</td>
</tr>
<tr>
<td>(c)</td>
<td>
</td>
<td>El DFA que se muestra tiene 4 estados finales consecutivos partiendo de q_0 (inicial) a q_3, para cada escenario donde no haya “a” o haya 1, 2 o 3, y que cuando se excede de las mismas, pasará al estado trampa q_4 indicando que se ha incumplido con la condición de no haber más de 3 letras “a”. En cada estado es indiferente si se encuentran letras “b”, ya que no son de nuestro interés y no estarían modificando los casos.</td>
</tr>
<tr>
<td>(d)</td>
<td>
</td>
<td>El DFA fue diseñado con estados específicos para cada condición que se pueda presentar, entre más a la derecha esté el estado, más letras “b” se han encontrado; si se ha llegado al estado trampa q_6, es porque se ha excedido de las 2 letras “b” necesarias. Entre más arriba se encuentre el estado, más letras “a” se han encontrado; en los estados de más arriba, es indiferente si se ha ingresado más letras “a”, ya que esta condición solo pide contener al menos una “a”, por lo que si se encuentran más no se incumple la condición. Finalmente, q_5, nuestro estado final, es accesible encontrando 2 letras “b” y al menos una “a”, es por ello que se encuentra hasta el extremo superior derecha.</td>
</tr>
<tr>
<td>(e)</td>
<td>
</td>
<td>El DFA es similar al anterior, cuenta con cada estado específico de la cadena ingresada, donde entre más a la derecha se encuentre el estado, representa la cantidad de letras “a” ingresadas, donde se pasa al estado trampa q_{12} si se excede de 2 letras “a”. A su vez, entre más arriba se encuentren los estados, más letras “b” han sido encontradas, donde en el tope nos es indiferente si se ingresan más letras “b”, ya que permaneceremos en el estado que nos encontremos debido a que no se incumple con la condición. Finalmente, el estado final q_{11}, en el extremo superior derecho, es accesible con exactamente dos letras “a” y más de 2 letras “b”, evidentemente sin importar el orden.</td>
</tr>
</table>

11. Mostrar la regularidad de L.

L es un lenguaje regular ya que existe un DFA que lo logre representar, el cual es el que se presenta en seguida.

El DFA se diseño con la cantidad de estados específicos según la cadena que puede tomar ***v***, es decir, si sabemos que el alfabeto solo consta de la letra “*a*” y “*b*”, y que la longitud de ***v*** solo puede ser 2, entonces tendremos que ***v*** puede ser alguna cadena del conjunto *{aa, ab, ba, bb}*.

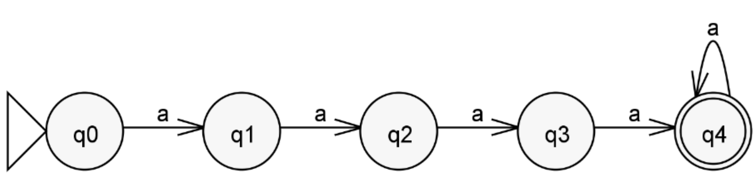
Entonces tendremos que los caminos se van a bifurcar en 4 posibles, donde una vez “leídos” los dos primeros elementos de la cadena, se validará que sean exactamente los mismos al acabarla, y que, en caso contrario, regresarán hasta el punto donde recién se ingresaron los 2 primeros, evidentemente para volver a "validar" desde ese punto y de esta forma sean iguales los dos primeros elementos con los dos últimos.



12. Mostrar la regularidad de L.

L es un lenguaje regular ya que existe un DFA que lo logre representar, el cual es el que se presenta en seguida.

Este DFA es bastante sencillo, ya que solo es necesario tener 5 estados, donde los primeros 4 (desde q_0 hasta q_3) van a ser de estados a la espera de que se ingrese una letra "*a*" y de esta forma hacer la transición al siguiente estado, para que cuando se llegue al último estado final q_4, no importe la cantidad de "*a*" que sean ingresadas después, la cadena ya habrá cumplido con la condición de tener longitud mayor o igual a 4.



13. Mostrar la regularidad de L.

L es un lenguaje regular ya que existe un DFA que lo logre representar, el cual es el que se presenta en seguida.

Este DFA cuenta con 5 estados finales, en donde a partir de q_0 hasta q_3 se tendrían cadenas de longitud 0 hasta 3, lo cual cumple con la condición de ser de longitud distinta a 4; q_4 es el estado para cuando no se cumple la condición, por ello no es estado final, sin embargo, después de este estado se encuentra q_5, donde además de ser final, no importará la cantidad de "*a*" que sean ingresadas a posterior, ya que habrá cumplido con la condición de ser distinto a 4.