Modelos de Computación (2015-2016)
Grado en Ingeniería Informática
Universidad de Granada

Práctica 4

José Carlos Martínez Velázquez

15 de diciembre de 2015

1. Dados los alfabetos A={0,1,2,3} y B={0,1} y el homomorfismo f de A^* a B^* dado por: f(0)=00, f(1)=01, f(2)=10, f(3)=11. Resolver las siguientes cuestiones:

a. Sea L_1 el conjunto de palabras de B^* tales que no comienzan con la subcadena 10. Construir un autómata finito determinista que acepte $f^{-1}(L_1)$.

Lo primero que hay que entender es qué es $f^{-1}(L_1)$. Dado que f convierte palabras de A^* en palabras de B^*, entonces f^{-1} convertirá palabras de B^* en palabras de A^*. Como $L_1 \subseteq B^*$, entonces tiene sentido aplicar $f^{-1}(L_1)$, que nos devolverá un subconjunto de A^*. Es decir: $f^{-1}(L_1) \subseteq A^*$.

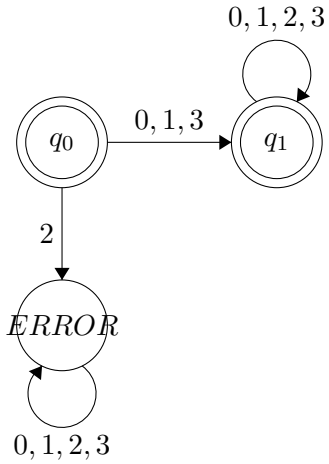
Ahora vamos a tratar de definir el lenguaje L_1 de una manera simbólica. En este caso, con una expresión regular pues suponemos que se puede encontrar (el lenguaje debe ser regular). Entonces:

$$L_1 = \{((00 + 01 + 11)(00 + 01 + 10 + 11)^*)^*\}$$

Esto permite generar cualquier palabra que empiece por 00, por 11, por 10 o la palabra vacía. Hay que restringir que sean pares de números, para poder aplicar el homomorfismo inverso (por ejemplo, las palabras $f^{-1}(0)$ ó $f^{-1}(1)$ no están definidas). Ahora que hemos conseguido formalizar el lenguaje L_1, es trivial definir $f^{-1}(L_1)$:

$$f^{-1}(L_1) = \{((0 + 1 + 3)(0 + 1 + 2 + 3)^*)^*\}$$

Una vez conseguido ésto, también es trivial definir el autómata que nos piden:



b. Construir un autómata finito determinista que acepte el lenguaje $L_2 = \{uu^{-1} \ / \ u \in B^*\}$.

En este ejercicio podríamos intentar encontrar un autómata finito determinista y, tras unas horas (o días) buscando podríamos empezar a sospechar que no es un lenguaje regular. Se podría haber empezado a sospechar antes, dado que no es posible generar una gramática de tipo 3 que defina palíndromos.

Sea como sea, pasaremos a demostrar que el lenguaje no es regular mediante la negación del Lema de Bombeo. Vamos a elegir una palabra del lenguaje, definiendo una longitud arbitraria. Entonces, nuestra palabra sería: $z = 0^n 1^{2n} 0^n$, cuya longitud sería 4n.

Ahora debemos dividir ésta palabra en u, v y w, cumpliendo ciertas condiciones:

$$z = uvw \begin{cases} |uv| & \leq & n \\ |v| & \geq & 1 \end{cases}$$

Donde: $uv = 0^n$ *y* $w = 1^{2n}0^n$. De aquí se puede deducir que:

$$u = 0^k \Rightarrow k < n$$
$$v = 0^l \Rightarrow l \geq 1$$
$$w = 0^{n-k-l}1^{2n}0^n$$

Ahora, si demostramos que existe un numero natural para el que $uv^i w$ no pertenece al lenguaje L_2, estaremos demostrando que L_2 no es regular.

Elegimos i=2, entonces:

$$uv^2w = [0^k(0^l)^2 0^{n-k-l}]1^{2n}0^n = 0^{n+l}1^{2n}0^n$$

Dado que $l \geq 1$, uv^2w no pertenece al lenguaje y, por ende, hemos demostrado que L_2 no es regular, por lo tanto, no se puede encontrar ni un AFD, ni una expresión regular para el lenguaje y tampoco una gramática de tipo 3 que lo genere.

c. Sea L_3 el conjunto de palabras de A^* definido como $L_3 = \{0^k 3^k \ / \ 1 \leq k \leq 20\}$. Construir una expresión regular que represente a $f(L_3)$.

En este apartado partimos de la base de que todo lenguaje finito es regular, por lo tanto se debe poder encontrar una expresión regular que defina el lenguaje. Sabemos que $f(L_3) = \{(00)^k(10)^k / 1 \leq k \leq 20\}$, por lo tanto, la expresión regular que buscamos es la siguiente:

$$f(L_3) = ($$
$$0010+$$
$$00001010+$$

000000101010+

0000000010101010+

00000000001010101010+

000000000000101010101010+

0000000000000010101010101010+

00000000000000001010101010101010+

000000000000000000101010101010101010+

0000000000000000000010101010101010101010 + |

00000000000000000000001010101010101010101010+

000000000000000000000000101010101010101010101010+

0000000000000000000000000010101010101010101010101010+

00000000000000000000000000001010101010101010101010101010+

000000000000000000000000000000101010101010101010101010101010+

0000000000000000000000000000000010101010101010101010101010101010+

00000000000000000000000000000000000101010101010101010101010101010101010+

000000000000000000000000000000000000101010101010101010101010101010101010+

000000000000000000000000000000000000000101010101010101010101010101010101010)

Como el lenguaje es finito, siempre se puede expresar el lenguaje como la unión de todas las palabras que lo componen y eso es una expresión regular.

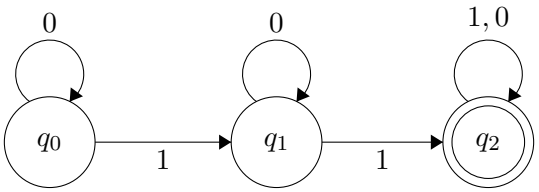
2. Sea L_4 el conjunto de palabras de B^* que contienen la subcadena 11. Sea L_5 el conjunto de las palabras de B^* de longitud múltiplo de tres. Construir el AFD minimal que acepte el lenguaje $L_4 \cap L_5$.

Primero definiremos los lenguajes mediante expresiones regulares, de los que luego, construiremos sus autómatas. Los lenguajes son los siguientes:

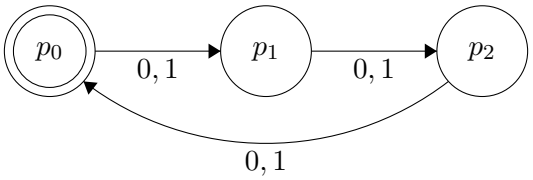
$$L_4 = (0|1)^*11(0|1)^*$$

$$L_5 = (000|001|010|011|100|101|110|111)^*$$

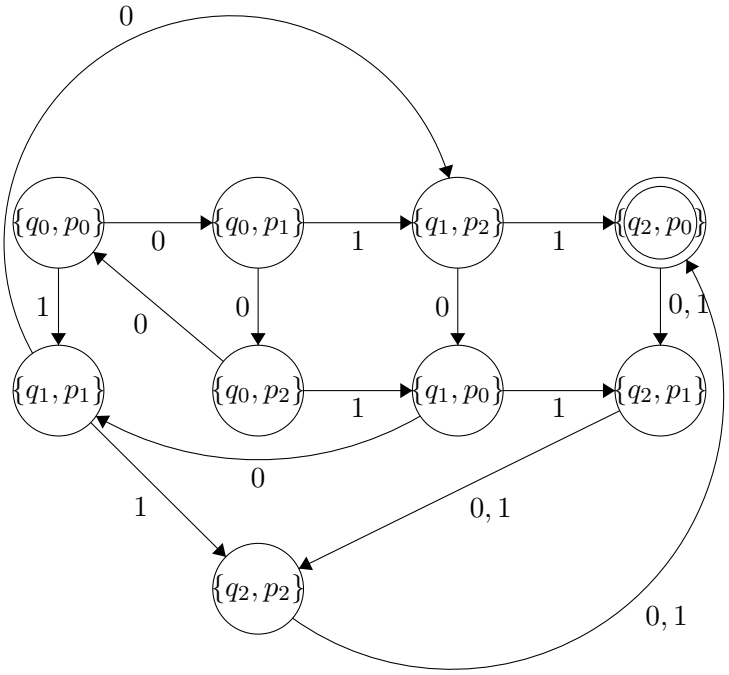
Una vez tenemos los lenguajes es sencillo ver que los autómatas son:
L_4:



L_5:



El autómata intersección será, por tanto:



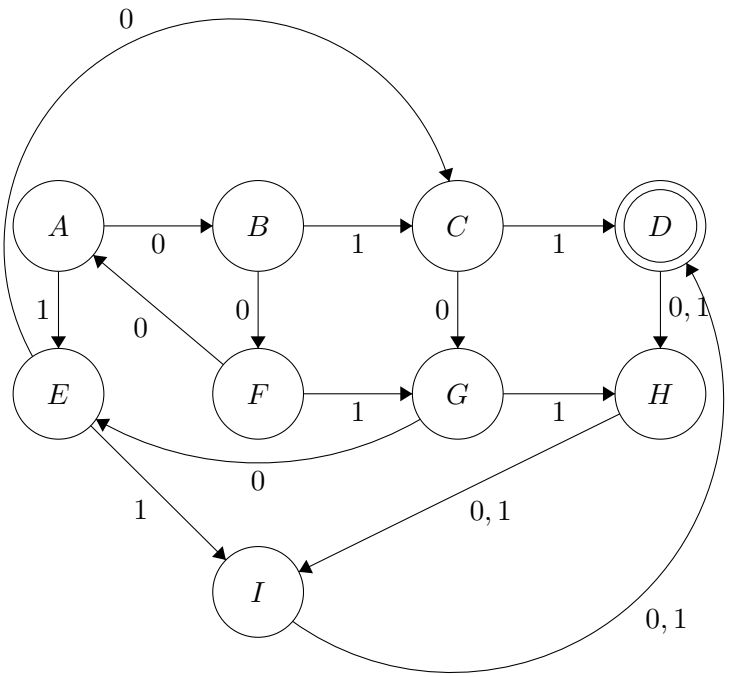
Antes de continuar, para facilitar el algoritmo de minimización establecemos las siguientes equivalencias

$$Estado\ A\ =\ \{q_0, p_0\} \quad Estado\ B\ =\ \{q_0, p_1\} \quad Estado\ C\ =\ \{q_1, p_2\}$$

$$Estado\ D\ =\ \{q_2, p_0\} \quad Estado\ E\ =\ \{q_1, p_1\} \quad Estado\ F\ =\ \{q_0, p_2\}$$

$$Estado\ G\ =\ \{q_1, p_0\} \quad Estado\ H\ =\ \{q_2, p_1\} \quad Estado\ I\ =\ \{q_2, p_2\}$$

Entonces, nuestro autómata es el siguiente:



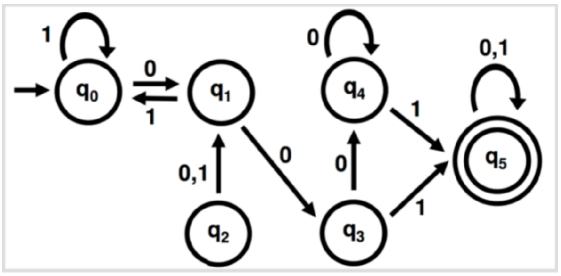
Aplicamos el algoritmo de minimización, y vemos cómo todos los estados son independientes:

	{A	B	C	E	F	G	H	I}	{D}
0	B	F	G	C	A	E	I	D	H
	{A	B	C	E	F	G	H}	{I}	{D}
0	B	F	G	C	A	E	I	D	H
	{A	B	C	E	F	G}	{H}	{I}	{D}
0	B	F	G	C	A	E	I	D	H
1	E	C	D	I	G	H	I	D	H
	{A	B	C	F}	{E}	{G}	{H}	{I}	{D}
0	B	F	G	A	C	E	I	D	H
	{A	B	F}	{C}	{E}	{G}	{H}	{I}	{D}
0	B	F	A	G	C	E	I	D	H
1	E	C	G	D	I	H	I	D	H
	{A}	{B}	{F}	{C}	{E}	{G}	{H}	{I}	{D}

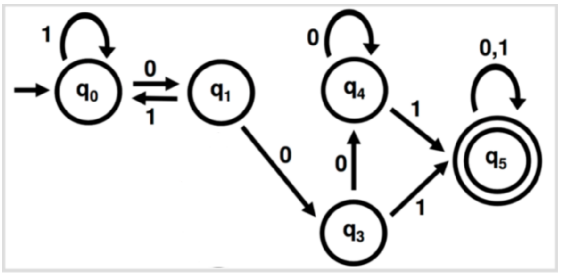
Por tanto, el autómata obtenido es minimal.

3. Calcular el AFD Minimal que acepte el mismo lenguaje que el siguiente AFD. Utilizar el algoritmo de minimización visto en clase.

a)



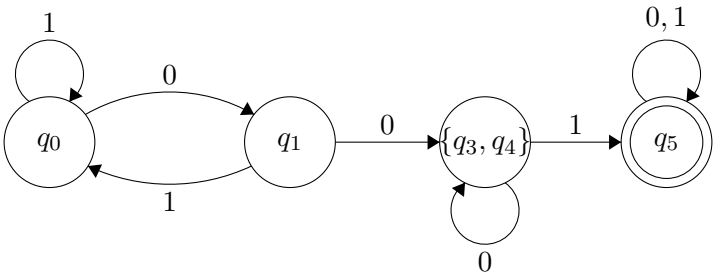
Lo primero para aplicar correctamente el algoritmo es eliminar los estados inaccesibles. Comprobamos pues, cómo no se puede llegar a q_2 desde ningún estado (no le llega niguna flecha). Eliminamos el estado y nos queda el siguiente autómata:



Aplicamos el algoritmo y obtenemos la siguiente tabla:

	$\{q_0$	q_1	q_3	$q_4\}$	$\{q_5\}$
0	q_1	q_3	q_4	q_4	q_5
1	q_0	q_0	q_5	q_5	q_5
	$\{q_0$	$q_1\}$	$\{q_3$	$q_4\}$	$\{q_5\}$
0	q_1	q_3	q_4	q_4	q_5
	$\{q_0\}$	$\{q_1\}$	$\{q_3$	$q_4\}$	$\{q_5\}$
0	q_1	q_3	q_4	q_4	q_5
1	q_0	q_5	q_5	q_5	q_5
	$\{q_0\}$	$\{q_1\}$	$\{q_3$	$q_4\}$	$\{q_5\}$

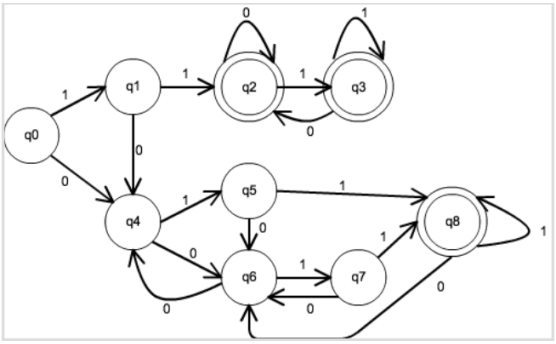
Por los resultados, observamos que todos los estados son independientes excepto q_3 y q_4, que son equivalentes. El autómata minimal obtenido es el siguiente:



El cual parece aceptar el lenguaje formado por todas las cadenas que contienen la subcadena 001:

$$L = \{uvw \in \{0,1\}^* \ / \ u, w \ \in \{0,1\}^* \wedge v = 001\}$$

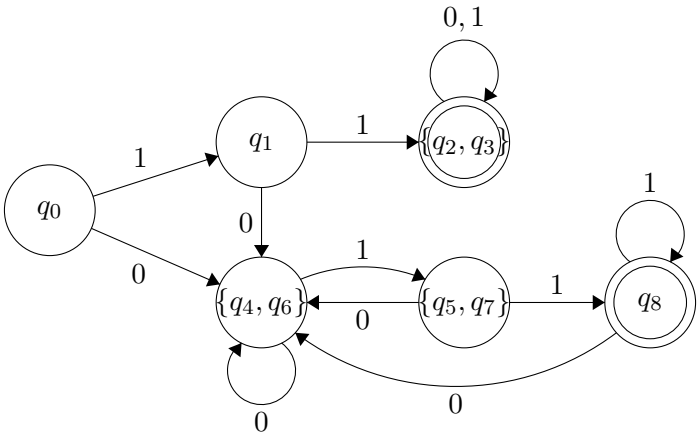
b)



En este ejercicio vemos cómo todos los estados son accesibles (a cada estado le llega al menos una flecha), entonces no se puede quitar ninguno. Aplicamos el algoritmo con todos éstos estados y obtenemos la siguiente tabla:

	$\{q_0$	q_1	q_4	q_5	q_6	$q_7\}$	$\{q_2$	q_3	$q_8\}$
0	q_4	q_4	q_6	q_4	q_4	q_6	q_2	q_2	q_6
	$\{q_0$	q_1	q_4	q_5	q_6	$q_7\}$	$\{q_2$	$q_3\}$	$\{q_8\}$
0	q_4	q_4	q_6	q_6	q_4	q_6	q_2	q_2	q_6
1	q_1	q_2	q_5	q_8	q_7	q_8	q_3	q_3	q_8
	$\{q_0$	q_4	$q_6\}$	$\{q_1\}$	$\{q_5$	$q_7\}$	$\{q_2$	$q_3\}$	$\{q_8\}$
0	q_4	q_6	q_4	q_4	q_6	q_6	q_2	q_2	q_6
1	q_1	q_5	q_7	q_2	q_8	q_8	q_3	q_3	q_8
	$\{q_0\}$	$\{q_4$	$q_6\}$	$\{q_1\}$	$\{q_5$	$q_7\}$	$\{q_2$	$q_3\}$	$\{q_8\}$
0	q_4	q_6	q_4	q_4	q_6	q_6	q_2	q_2	q_6
1	q_1	q_5	q_7	q_2	q_8	q_8	q_3	q_3	q_8
	$\{q_0\}$	$\{q_4$	$q_6\}$	$\{q_1\}$	$\{q_5$	$q_7\}$	$\{q_2$	$q_3\}$	$\{q_8\}$

Donde vemos que los estados q_4 y q_6 son equivalentes entre sí, también los estados q_5 y q_7 y los estados q_2 y q_3. Por lo que nos hemos ahorrado 3 estados. El autómata minimal resultante sería el siguiente:



El cual parece aceptar el conjunto de palabras que contienen empiezan o terminan por 11:

$$L = \{uvw \in \{0,1\}^* \ / \ v \ \in \{0,1\}^* \wedge (u = 11 \ \vee \ w = 11)\}$$