Actividades Prácticas

Ejercicios propuestos del Autómata Linealmente Acotado (ALA)

Ejercicio 1

Para cada uno de los siguientes ítems, construir un autómata linealmente acotado que, dada una cadena $\alpha \in$ **{0, 1}**$^+$ escrita en su cinta y encerrada entre los símbolos de inicio y fin de cinta, acepte aquellas que:

a) Tengan un número par de ceros.

b) Tengan un número impar de unos.

c) Tengan un número par de caracteres.

d) Tengan, a la vez, un número par de ceros y un número par de unos.

Ejercicio 2

Identificar las posibles condiciones de error cuando el autómata linealmente acotado del Ejemplo 6.2 verifica que las cadenas de entrada son palíndromas de largo par o impar. Incorporar el estado de error al correspondiente grafo y completar la tabla de la función de transición.

Ejercicio 3

Proponer un ALA que reemplace la presencia de subcadenas **00** por **11** en una cadena que no tiene una longitud predeterminada. Por ejemplo, el ALA transformará la cadena α**=00101100** en la cadena β**=11101111**.

Ejercicio 4

Proponer una solución alternativa al problema presentado en el Ejemplo 6.3, que es el de ordenar en forma creciente los símbolos de las cadenas binarias representadas en su cinta. Comparar ambos autómatas utilizando como indicador la complejidad estructural de Shannon.

Ejercicio 5

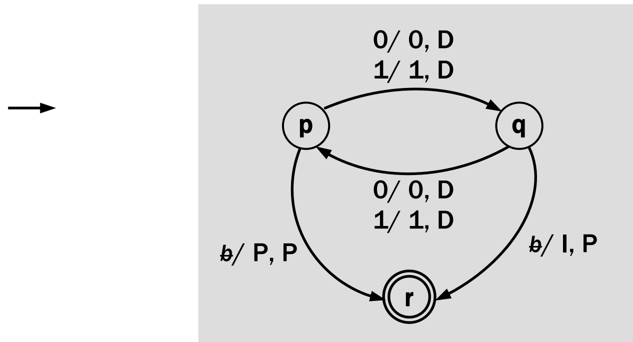
Se desea diseñar un autómata linealmente acotado que comande el siguiente efecto visual en un cartel luminoso: una barra de seis segmentos (símbolo **=**) tiene un largo inicial de seis y color rojo, progresivamente su largo se reduce en uno (alternativamente por cada extremo) hasta desaparecer y luego comienza nuevamente a crecer, esta vez de color verde. La secuencia en que la barra se reduce y aumenta cambiando de color se repite indefinidamente. Presentar el grafo, la función de transición del ALA y un ejemplo que demuestre su correcta operación.

Ejercicios propuestos de la Máquina de Turing (MT)

Ejercicio 6

Comprobar que la función que cumple la siguiente máquina de Turing es verificar si una cadena formada por bits, es de longitud par o impar. Al finalizar graba una **P** si la cadena resultó ser par o una **I** en caso contrario. La cadena no tiene longitud fija.

Solución $\quad MT$ = ({0,1}, {0,1,I,P,ƀ}, {p,q,r}, p, {r}, f, ƀ)



f	0	1	ƀ
→p	q,0,D	q,1,D	r,P,P
q	p,0,D	p,1,D	r,I,P
*r			

Figura 6.24. Tabla 6.12: Grafo y función de transición de la máquina de Turing del Ejercicio 6.

Ejercicio 7

Construir una MT que verifique en una cadena formada por bits, si la cantidad de **0s** (ceros) es par o impar y grabe la letra **P** si es par o **I** si es impar, al finalizar la cadena. La cadena no tiene longitud fija.

Ejemplo: Entrada ~~b~~0100110~~b~~

Salida ~~b~~0100110**P**~~b~~

Ejercicio 8

Diseñar una MT que verifique en una cadena formada por bits, si la cantidad de **1s** (unos) es par o impar. En caso de que sea par, al finalizar la comprobación, negar la cadena (cambiar ceros por unos y viceversa), en caso contrario, la máquina solo se detiene. La cadena no tiene longitud fija.

Ejemplo: Entrada ~~b~~101011~~b~~

Salida: ~~b~~**010100**~~b~~ (cadena negada)

Ejercicio 9

Construir una MT que opere sobre el alfabeto **Σ = {A, B, C, D, E, F, G, H, I}** y sustituya las subtiras en una cadena dada según se indica:

BA por **FG** y **DE** por **HI**

Ejemplo: Entrada ~~b~~CFH**BA**AG**DE**~~b~~

Salida ~~b~~CFH**FG**AG**HI**~~b~~

Ejercicio 10

Diseñar una MT que cree la imagen refleja de la entrada después de un ***** (asterisco), dejando inalterada al finalizar la cadena original.

Ejemplo: Entrada ~~b~~ABB*~~b~~

Salida ~~b~~ABB***BBA**~~b~~

Ejercicio 11

Diseñar una MT que verifique si dos palabras separadas por un ***** (asterisco) son iguales (en este caso, acepta la cadena de entrada).

Ejemplo: ~~b~~**0010*****0010**~~b~~

Ejercicio 12

Diseñar una MT que copie el primer nibble de un byte sobre el segundo, que se encuentra separado por un ***** (asterisco).

Ejemplo: Entrada ~~b~~**1001***0011~~b~~

Salida ~~b~~1001***1001**~~b~~

Ejercicio 13

Construir una MT que intercambie el primer nibble por el segundo, estando ambos separados por un ***** (asterisco).

Ejemplo: Entrada ~~b~~1001*0011~~b~~

Salida ~~b~~**0011*****1001**~~b~~

Ejercicio 14

Construir una MT que reordene en orden ascendente una cadena de bits de cualquier longitud y la grabe sobre la cinta a partir de un ***** (asterisco).

Ejemplo: Entrada ~~b~~0110100*~~b~~

Salida ~~b~~0110100***0000111**~~b~~

Ejercicio 15

Construir una MT que verifique en una cadena formada por bits, que haya la misma cantidad de **0s** (ceros) que de **1s** (unos). La cadena no tiene longitud fija.

Ejemplo: ~~b~~**00100111**~~b~~ (la misma cantidad de **0** que de **1**)

Ejercicio 16

Construir una MT para reconocer cada uno de los lenguajes presentados en el ejercicio 4 del Capítulo 5 (autómatas con pila).

Ejercicio 17

Identificar las posibles condiciones de error cuando la máquina de Turing del Ejemplo 6.1 evalúa sus cadenas de entrada. Incorporar el estado de error en el correspondiente grafo y completar la tabla de la función de transición.

Ejercicio 18

Proponer una MT que resuelva el problema del Ejemplo 6.2 comportándose como un autómata con pila, implementando para ello una memoria LIFO en un sector de la cinta. Presentar el grafo de la MT, su función de transición y comparar esta solución con la del Ejemplo 6.2 utilizando como criterio la complejidad estructural de Shannon.

Ejercicio 19

Proponer una máquina de Turing que, para el caso de palíndromos de largo par, resuelva el problema del Ejemplo 6.2 en dos etapas: primero, identificando la mitad de la cadena y luego, comprobando que se trata verdaderamente de un palíndromo. Se sugiere leer los comentarios referidos a este problema al tratarse la *Combinación de Máquinas de Turing* en el teórico de este capítulo.

Ejercicio 20

Proponer una solución alternativa para el problema de ordenar cadenas de un alfabeto de tres símbolos **{0, 1, 2}**, sin utilizar un movimiento neutro **N** como fue el caso del Ejemplo 6.4. Presentar el grafo de la MT, su función de transición y comparar esta solución con la anterior utilizando como criterio la complejidad estructural de Shannon.

Ejercicio 21

Para aceptar las cadenas del lenguaje propuesto en el ejemplo 6.9 se propuso una MTND. Se pide como alternativa diseñar una MTD y comparar ambas máquinas utilizando la complejidad estructural de Shannon.

Ejercicios propuestos de complejidad de la máquina de Turing

Ejercicio 22

Determinar las expresiones de complejidad temporal **T(*n*)** y complejidad espacial **E(*n*)** en función del largo ***n*** de la cadena procesada en el caso de la máquina de Turing presentada en el Ejemplo 6.1, destinada a validar cadenas de la forma general $\alpha = c^k d^k e^k$, donde $k > 0$.

Ejercicio 23

Determinar las expresiones de complejidad temporal **T(*n*)** y espacial **E(*n*)** en función del largo de la cadena ***n*** en el caso de la MT resuelta en el Ejercicio 10, destinada a generar una imagen refleja de la entrada.

Ejercicio 24

Determinar las expresiones de complejidad temporal **T(*n*)** y espacial **E(*n*)** en el caso de la MT resuelta en el Ejercicio 13, destinada a intercambiar dos nibbles.

Ejercicios resueltos del Autómata Linealmente Acotado (ALA)

Ejercicio 25

Construir un ALA que acepte cadenas de la forma general **$\alpha = a^i b^j$** para cada uno de los siguientes casos, donde **i > 0** y **j > 0**; **a)** i > j , **b)** i < j y **c)** i = j.

Solución 25.a

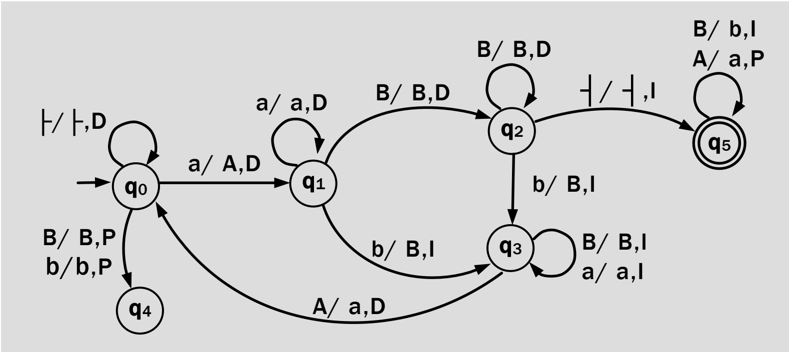


Figura 6.25: Grafo del autómata linealmente acotado del Ejercicio 25.a.

ALA = ({a,b}, {a,b,A,B, ├,┤}, {q_0, q_1, q_2, q_3, q_4,q_5}, q_0, {q_5}, *f*)

Solución 25.b

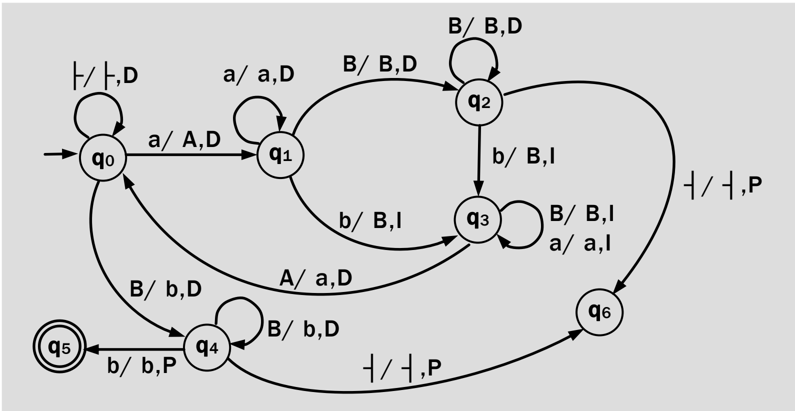


Figura 6.26: Grafo del autómata linealmente acotado del Ejercicio 25.b.

ALA = ({a,b}, {a,b,A,B, ├,┤}, {q_0, q_1, q_2, q_3, q_4,q_5,q_6}, q_0, {q_5}, *f*)

Solución 25.c

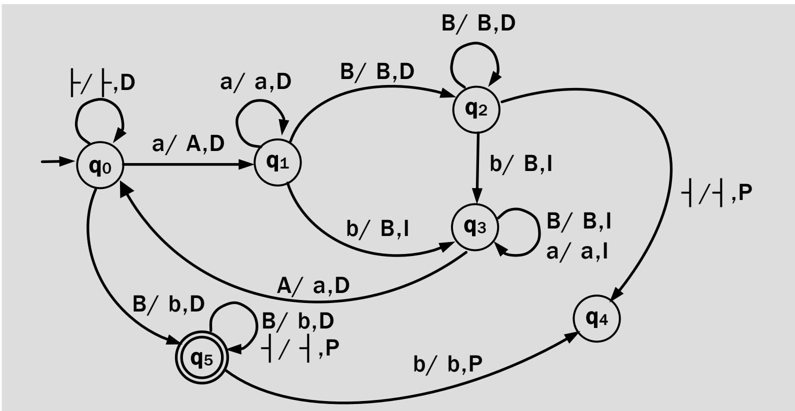


Figura 6.27: Grafo del autómata linealmente acotado del Ejercicio 25.c.

ALA = ({a,b}, {a,b,A,B, ├,┤}, {q_0, q_1, q_2, q_3, q_4,q_5}, q_0, {q_5}, *f*)

Ejercicio 26

Dado cualquier número binario α, construir un ALA que lo sobrescriba como α^{-1}, su cadena recíproca. La solución es:

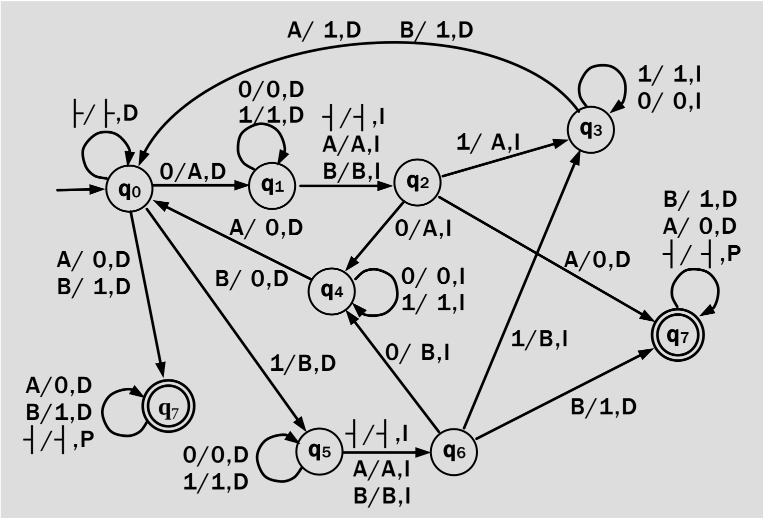


Figura 6.28: Grafo del autómata linealmente acotado del Ejercicio 26 (Notar que el estado final q_7 está repetido para mayor claridad).

$$ALA = (\{0,1\}, \{0,1,A,B,├,┤\}, \{q_0, q_1, q_2, q_3, q_4,q_5,q_6,q_7\}, q_0, \{q_7\}, f)$$

Ejercicio 27

Construir un ALA que acepte bytes de la forma $\alpha\alpha$, siendo α cualquier cadena de cuatro bits.

Solución

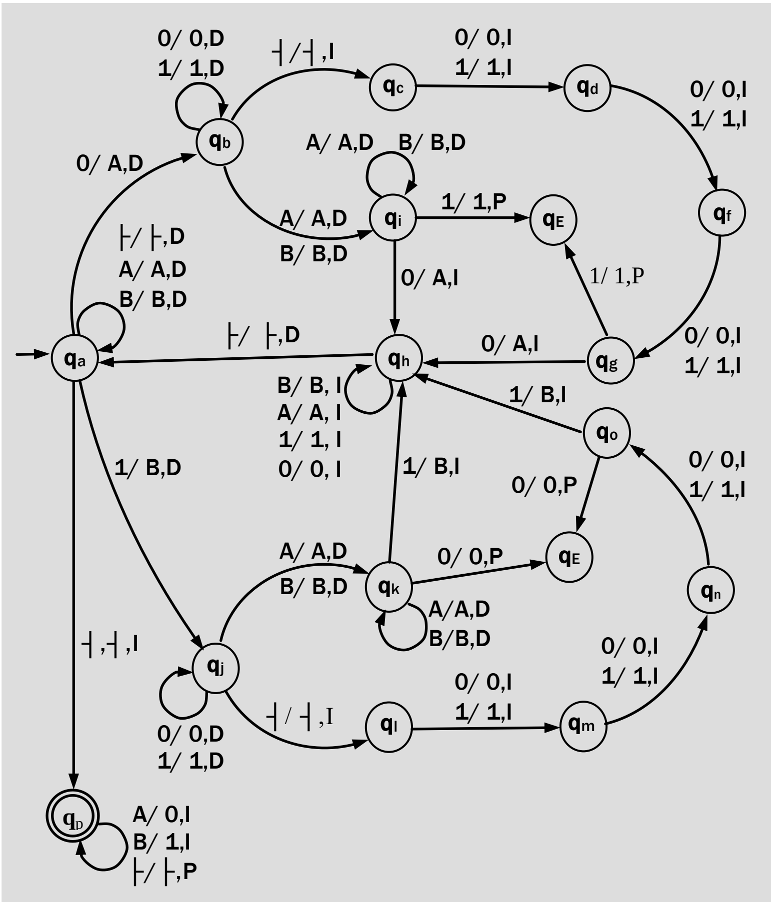


Figura 6.29: Grafo del autómata linealmente acotado del Ejercicio 27.

$ALA = (\ \{0,1\},\ \{0,1,A,B,\ \vdash,\dashv\ \},\ \{q_a,\ q_b,\ q_c,\ q_d,\ q_E,\ q_f,\ q_g,\ q_h,\ q_i,\ q_j,\ q_k,\ q_l,\ q_m,\ q_n,\ q_o,\ q_p\},\ q_a,\ \{q_p\},\ f\ )$

Nótese que en el grafo de la solución propuesta, representado en la Figura 6.29, el estado "$\mathbf{q_E}$" que representa la condición de error se encuentra repetido para mayor claridad.

Ejercicios resueltos de la Máquina de Turing (MT)

Ejercicio 28

Diseñar una MT tal que dada una palabra, encuentra las subcadenas **00** y las cambie por **11** y las subcadenas **11** las cambie por **00**. La palabra finaliza cuando se lee un **~~b~~** (blanco).

Ejemplo: Entrada ~~b~~01**00**10**11011**~~b~~

Salida ~~b~~01**11**10**00000**~~b~~

Solución

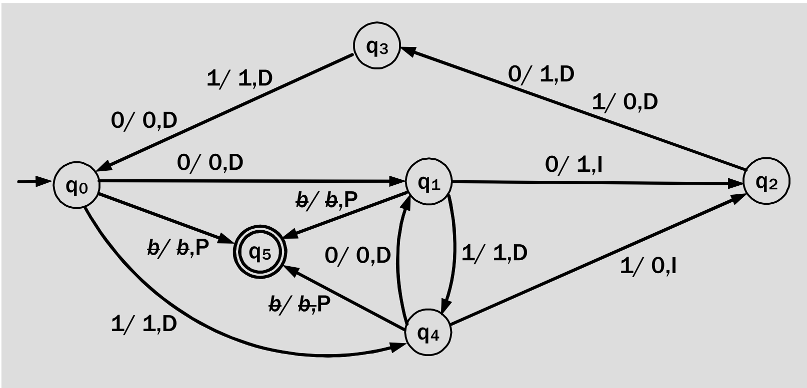


Figura 6.30: Grafo de la máquina de Turing del Ejercicio 28.

$MT = (\ \{0,1\},\ \{0,1,\ \cancel{b}\ \},\ \{q_0,\ q_1,\ q_2,\ q_3,\ q_4,\ q_5\},\ q_0,\ \{q_5\},\ f,\ \cancel{b})$

Ejercicio 29

Construir una MT que copie el nibble que se encuentra a la derecha del ***** (asterisco) sobre el que se encuentra a la izquierda del mismo.

$MT = (\{0,1\},\ \{0,1,\ A,\ B,\ \cancel{b}\ \},\ \{q_0,\ q_1,\ q_2,\ q_3,\ q_4,\ q_5,\ q_6, q_7\},\ q_0,\ \{q_7\},\ f,\ \cancel{b})$

Solución

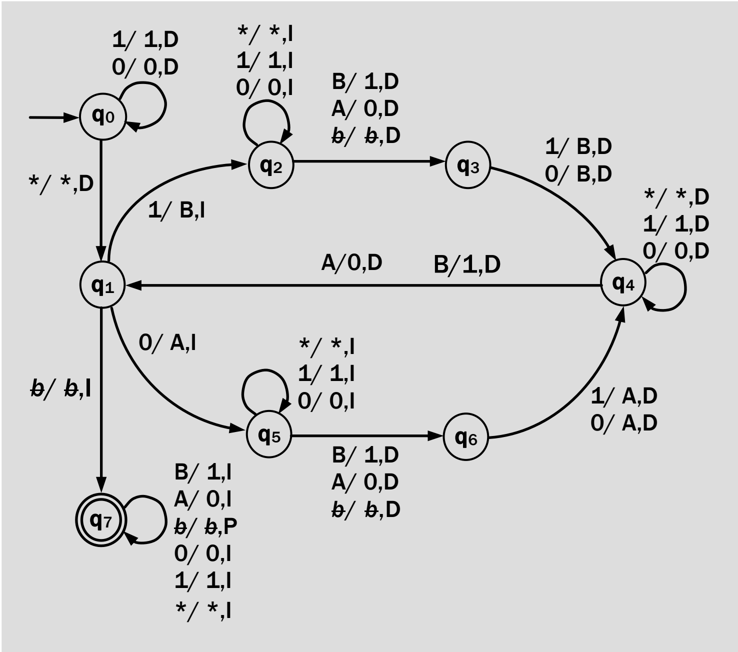


Figura 6.31: Grafo de la máquina de Turing del Ejercicio 29.

Ejercicio 30

Diseñar una MT por cada uno de los siguientes casos, donde la entrada está dada por dos nibbles separados por un ***** (asterisco).

a) Intercambiar los dos primeros bits del primer nibble, por los dos primeros bits del segundo nibble.

Ejemplo Entrada ƀ**01**01*__11__00ƀ

Salida ƀ**11**01*__01__00ƀ

b) Intercambiar los dos primeros bits del primer nibble, por los dos últimos bits del segundo nibble.

Ejemplo: Entrada ƀ**01**01*11**00**ƀ

Salida ƀ**00**01*11**01**ƀ

c) Intercambiar los dos últimos bits del primer nibble, por los dos primeros bits del segundo nibble.

Ejemplo: Entrada ƀ01**01***__00__11ƀ

Salida ƀ01**00***__01__11ƀ

d) Intercambiar los dos últimos bits del primer nibble, por los dos últimos bits del segundo nibble.

Ejemplo: Entrada ƀ01**01***11**00**ƀ

Salida ƀ01**00***11**01**ƀ

Solución 30.a

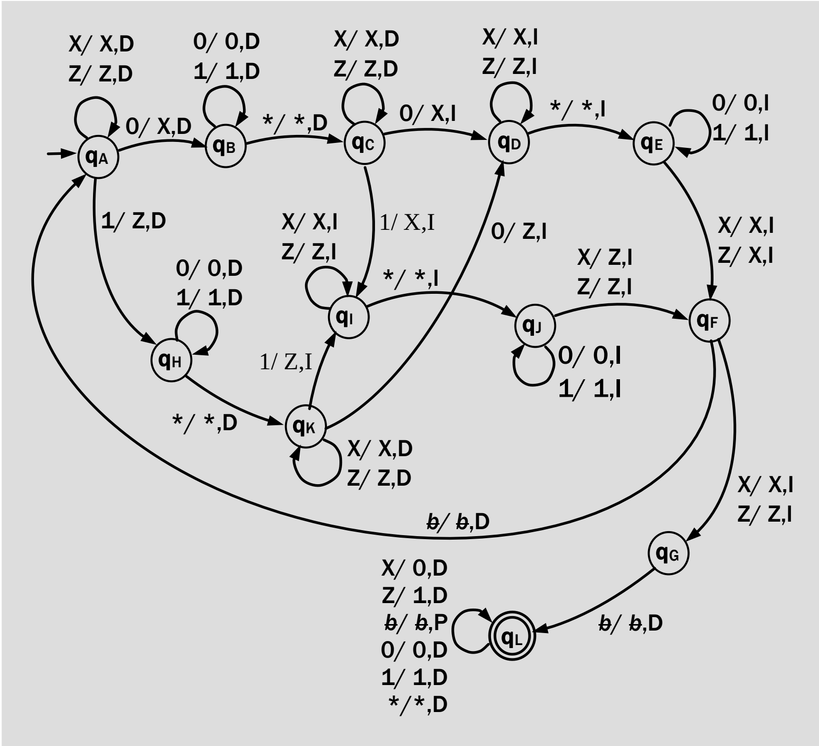


Figura 6.32: Grafo de la máquina de Turing del Ejercicio 30.a.

MT =({0,1}, {0,1, X, Z, ~~b~~}, {q_A, q_B, q_C, q_D, q_E, q_F, q_G, q_H, q_I, q_J, q_K, q_L}, q_A, {q_L}, f, ~~b~~)

Solución 30.b

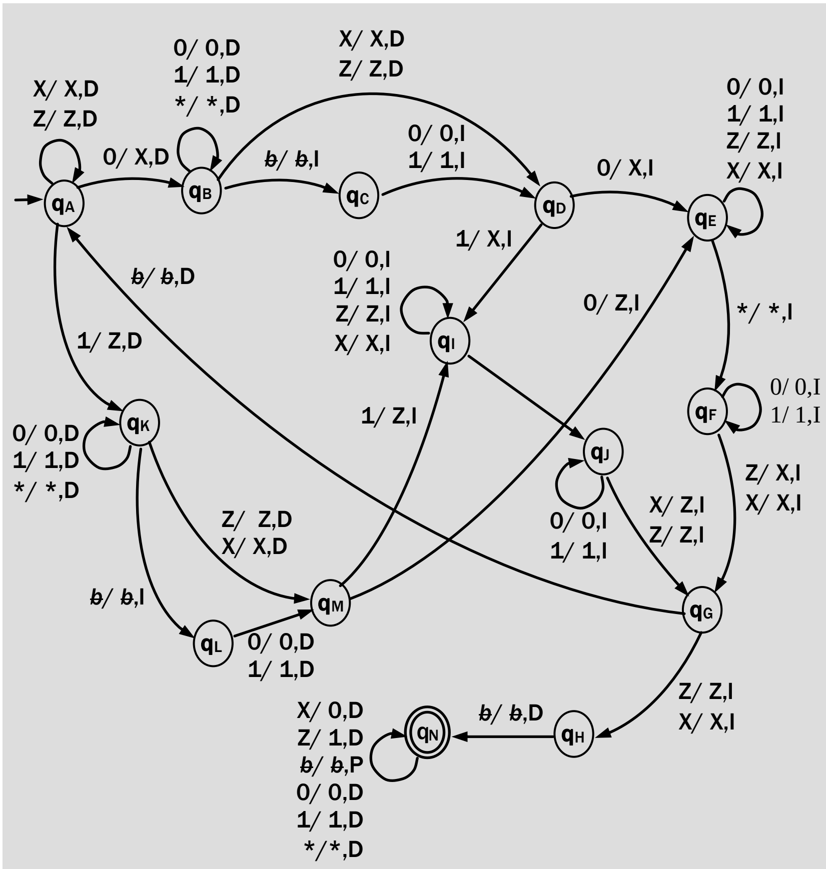


Figura 6.33: Grafo de la máquina de Turing del Ejercicio 30.b.

MT = ({0,1}, {0,1,X,Z,~~b~~}, {q_A, q_B, q_C, q_D, q_E, q_F, q_G, q_H, q_I, q_J, q_K, q_L, q_M, q_N}, q_A, {q_N}, f, ~~b~~)

Solución 30.c

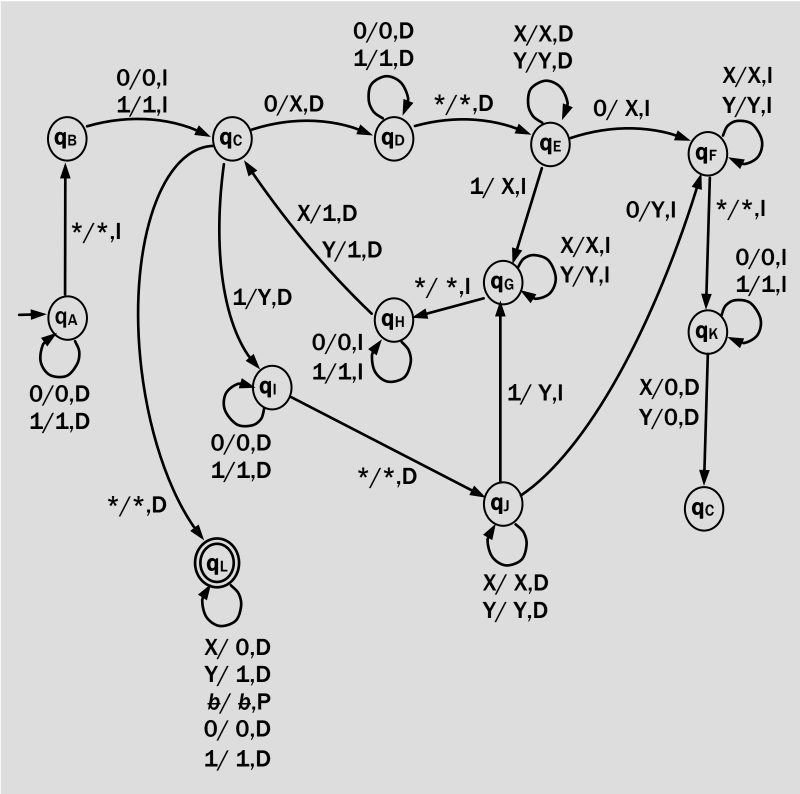


Figura 6.34: Grafo de la máquina de Turing del Ejercicio 30.c.

MT = ({0,1}, {0, 1, X, Z, ƀ}, {q_A, q_B, q_C, q_D, q_E, q_F, q_G, q_H, q_I, q_J, q_K, q_L}, q_A, {q_L}, *f*, ƀ)

Nótese que en el grafo de la solución propuesta, representado en la Figura 6.34, el estado "**q**$_C$" se encuentra repetido para mayor claridad.

Solución 30.d

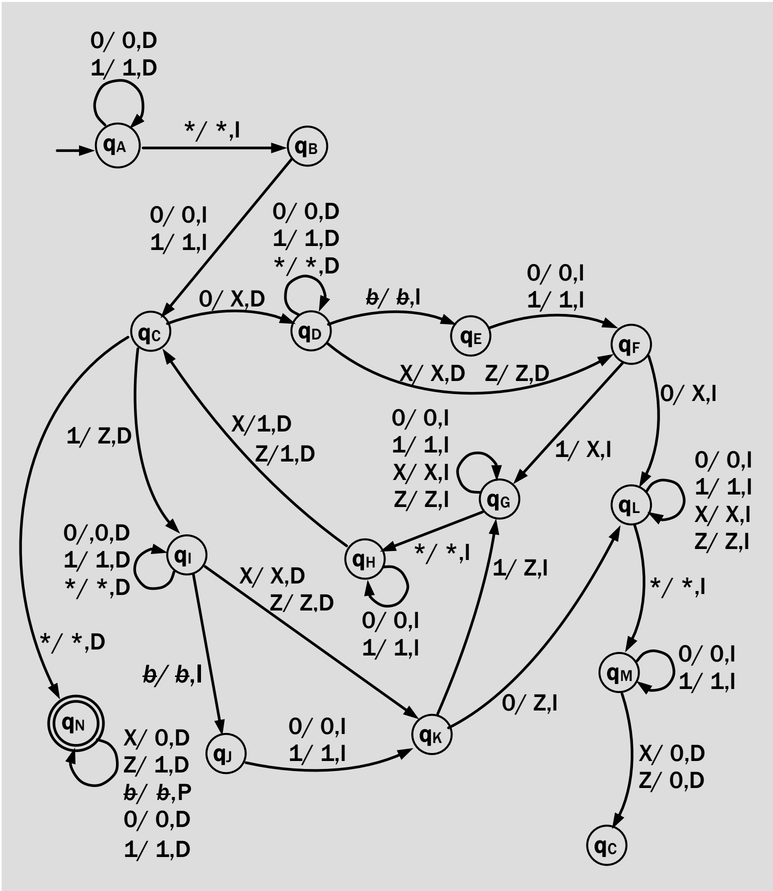


Figura 6.35: Grafo de la máquina de Turing del Ejercicio 30.d.

$MT = (\{0,1\}, \{0,1,X,Z,ƀ\}, \{q_A,q_B,q_C,q_D,q_E,q_F,q_G,q_H,q_I,q_J,q_K,q_L,q_M,q_N\}, q_A, \{q_N\}, f, ƀ)$

Al igual que en ejercicios anteriores, el estado "**q**$_C$" se encuentra repetido para mayor claridad, tal como lo muestra el grafo representado en la Figura 6.35.

Ejercicios resueltos a través de soluciones alternativas

Ya se ha expuesto que no hay una única solución para resolver un cierto problema. Por el contrario, siempre hay variadas alternativas, y para compararlas debe adoptarse un criterio, como puede ser su simplicidad lógica, el espacio requerido (complejidad espacial), el tiempo de proceso demandado (complejidad temporal), etc.

Para esclarecer esta idea se propone resolver las cuatro variantes del ejercicio 30 a través de un tipo de solución que reconoce dos etapas: *i)* identificación de los bits a ser intercambiados, para lo cual se utilizan dos símbolos auxiliares Ω_1={a,b} y *ii)* concreción del intercambio de los bits ya identificados, utilizándose los símbolos auxiliares Ω_2={A,B}.

Al adoptarse este enfoque, la primera etapa demanda un procedimiento específico de identificación para cada caso, mientras que en todos los casos la etapa de intercambio de bits es exactamente la misma. Con este fin, la *MT* que cumple esta segunda etapa es definida con suficiente generalidad como para contemplar todos los casos que puedan presentarse.

El enfoque propuesto queda encuadrado en el concepto de combinación de máquinas de Turing, que ya fue presentado al desarrollarse los conceptos teóricos, y la ventaja que ofrece no requiere de mayores comentarios. La *MT* de la etapa *ii* es definida, implementada y probada una sola vez, con el ahorro de tiempo que esto significa, adoptando la forma de un "componente".

Sin embargo, al igual de lo que ocurre en programación, estas soluciones generales no necesariamente son las más eficientes, y con seguridad cada caso tiene soluciones específicas de menor complejidad espacial y temporal. Al utilizar el enfoque propuesto, se obtiene mayor claridad conceptual y el reaprovechamiento de un mismo componente, pagando el costo de menor eficiencia operativa. Por lo tanto, la conveniencia de implementar este tipo de soluciones debe ser analizada en cada situación.

Con posterioridad a la presentación de estas soluciones alternativas, se hará una comparación de las complejidades de las soluciones del Ejercicio 30. Se recomienda al lector definir soluciones específicas o bien utilizar las soluciones propuestas en las figuras 6.32 a 6.35 para comparar las complejidades con la de la siguiente solución general propuesta.

Las soluciones alternativas son identificadas como 30.a’ a 30.d’ y sus grafos se presentan en las figuras 6.36 a 6.39.

Solución 30.a’

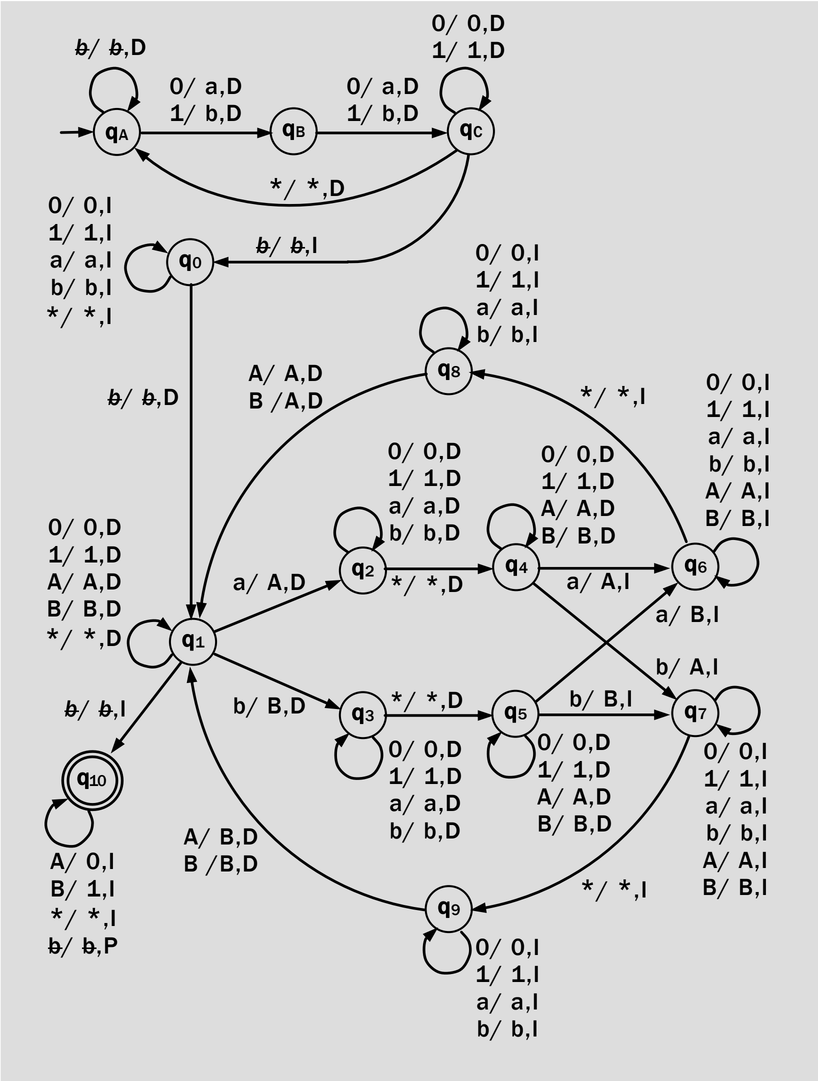


Figura 6.36: Grafo de la máquina de Turing del Ejercicio 30.a’.

MT = ({0,1,*}, {0,1,*,a,b,A,B,ƀ}, {$q_A, q_B, q_C, q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}$}, q_A, {q_{10}}, f, ƀ)

Solución 30.b’

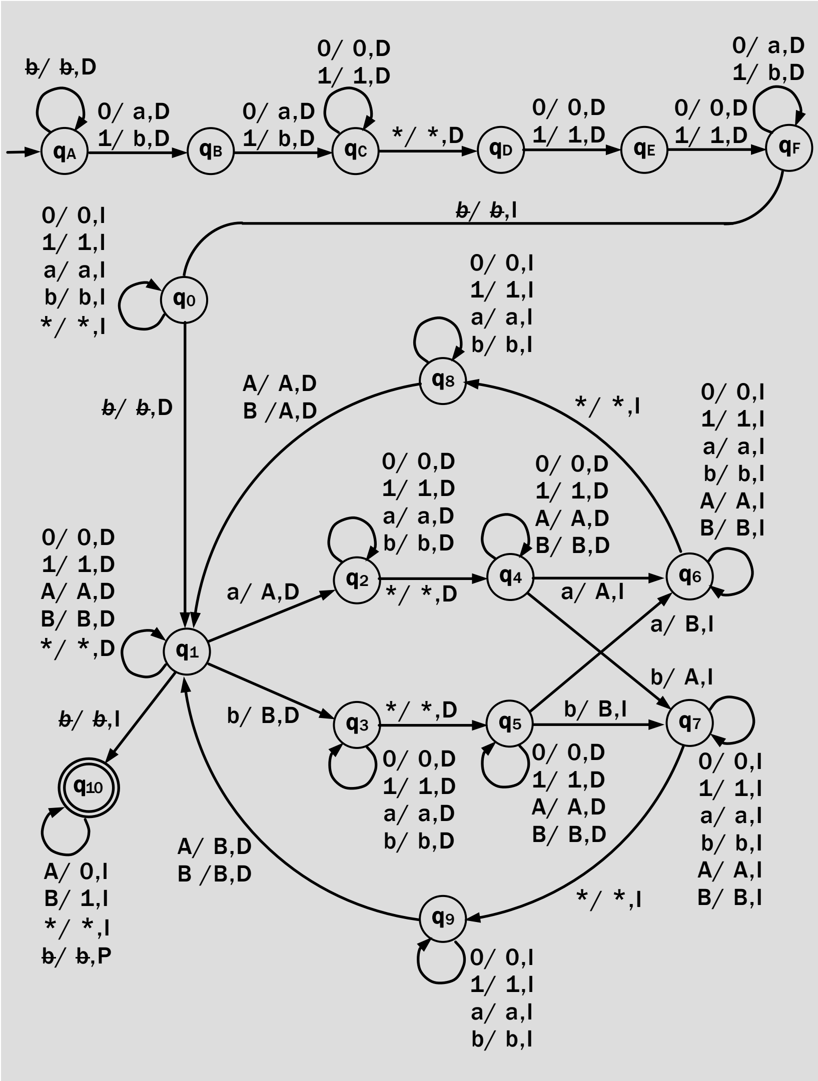


Figura 6.37: Grafo de la máquina de Turing del Ejercicio 30.b’.

MT=({0,1,*},{0,1,*,a,b,A,B,ƀ},{q_A,q_B,q_C,q_D,q_E,q_F,q_0,q_1,q_2,q_3,q_4,q_5,q_6,q_7,q_8,q_9,q_{10}}, q_A,{q_{10}}, f, ƀ)

Solución 30.c’

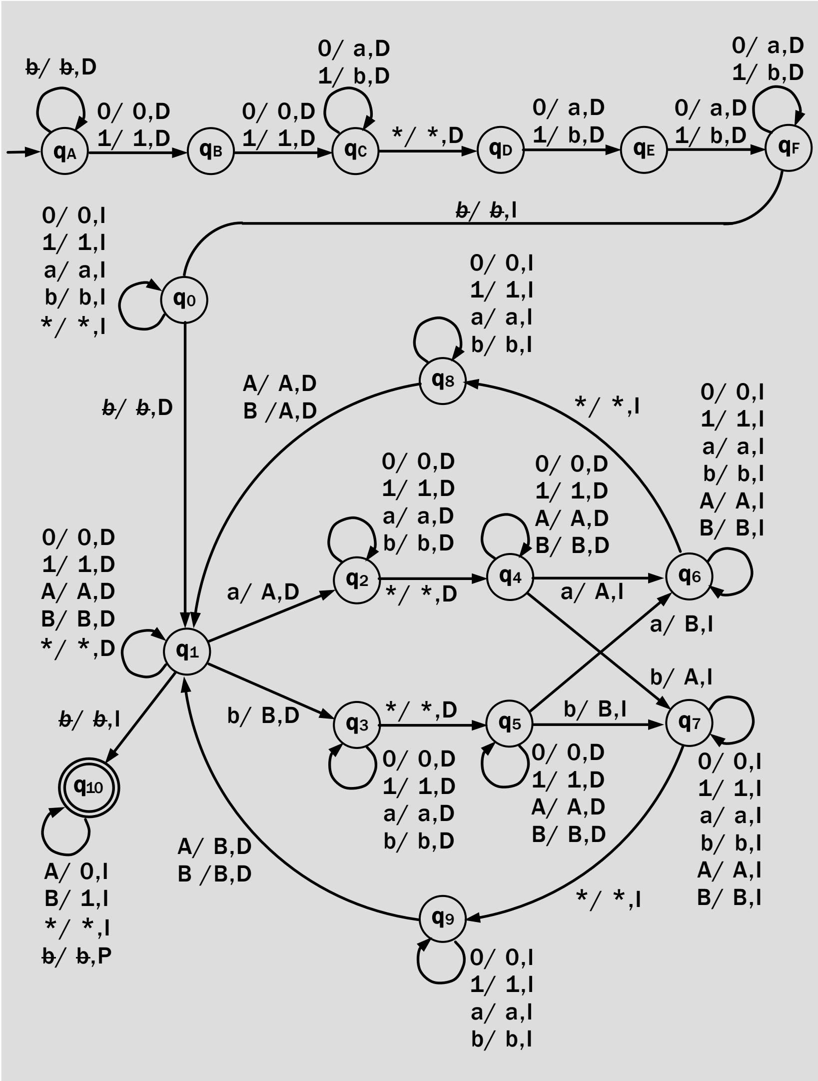


Figura 6.38: Grafo de la máquina de Turing del Ejercicio 30.c’.

MT = ({0,1,*},{0,1,*,a,b,A,B,ƀ},{q_A,q_B,q_C,q_D,q_E,q_F,q_0,q_1,q_2,q_3,q_4,q_5,q_6,q_7,q_8,q_9,q_{10}},q_A,{q_{10}},f, ƀ)

Solución 30.d’

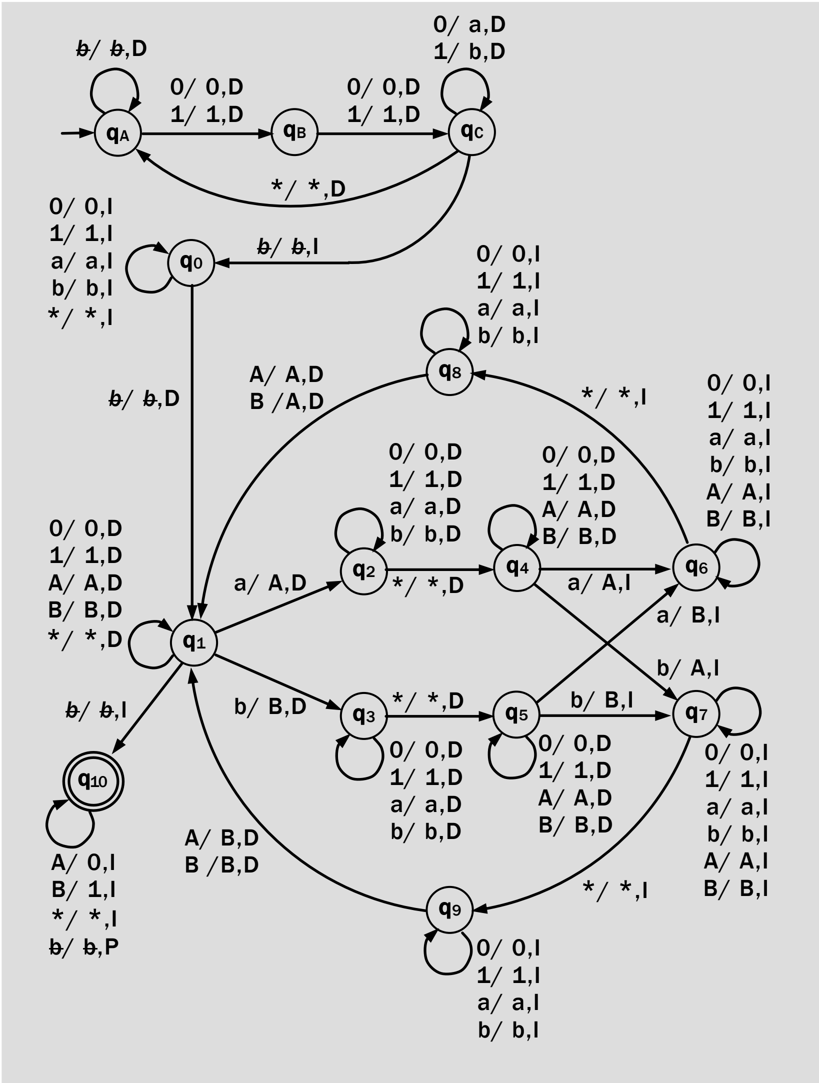


Figura 6.39: Grafo de la máquina de Turing del Ejercicio 30.d’.

MT= ({0,1,*},{0,1,*,a,b,A,B,ƀ},{q_A,q_B,q_C,q_0,q_1,q_2,q_3,q_4,q_5,q_6,q_7,q_8,q_9,q_{10}}, q_A, {q_{10}}, *f*, ƀ)

Si se analizan las complejidades temporales de las dos soluciones propuestas para cualquiera de los casos del Ejercicio 30, se comprueba que no dependen de la longitud de la cadena, ya que la longitud de la cadena no cambia. Es una complejidad polinómica de orden cero.

Pero, si además se comparan en cada caso dichas constantes, resultarán distintas. Por ejemplo, para la solución 30.a propuesta en la Figura 6.32, la cantidad de intervalos de tiempo necesarios para completar el algoritmo resulta ser de 36, mientras que para la solución 30.a’ propuesta en la Figura 6.36, la cantidad de intervalos requeridos son 60. Luego, podemos concluir en que la primera solución presenta mayor eficiencia operativa.

Ejercicio 31

Proponer una MT para reconocer cada uno de los lenguajes que se definen a continuación. En cada caso, presentar el grafo y la función de transición.

a) L = { $1^n 2^n 3^n$ | n > 0 }

b) L = { $1^m 2^n 3^m$ | m > 0, n > 0 }

c) L = { $1^m 2^n 3^m 4^n$ | m > 0, n > 0 }

Solución 31.a

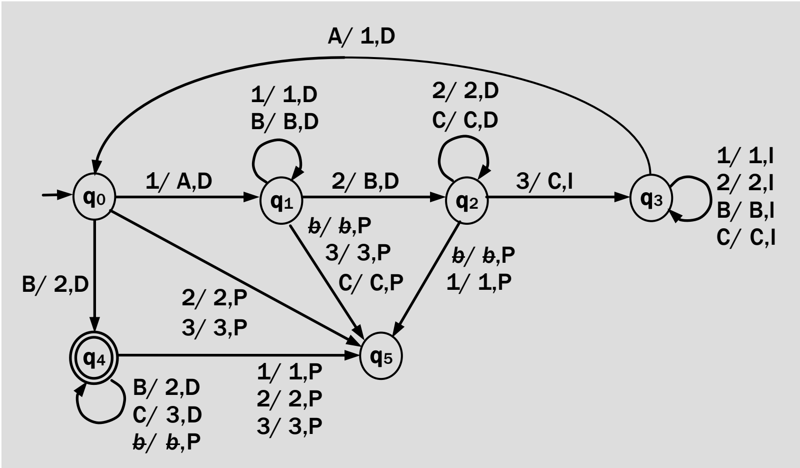


Figura 6.40: Grafo de la máquina de Turing del Ejercicio 31.a.

MT = ({1,2,3},{1,2,3,A,B,C,ƀ},{ q_0, q_1, q_2, q_3, q_4, q_5 }, q_0, { q_4}, *f*, ƀ)

f	1	2	3	A	B	C	ƀ
→q_0	q_1,A,D	q_5,2,P	q_5,3,P		q_4,2,D		
q_1	q_1,1,D	q_2,B,D	q_5,3,P		q_1,B,D	q_5,C,P	q_5,ƀ,P
q_2	q_5,1,P	q_2,2,D	q_3,C,I			q_2,C,D	q_5,ƀ,P
q_3	q_3,1,I	q_3,2,I		q_0,1,D	q_3,B,I	q_3,C,I	
*q_4	q_5,1,P	q_5,2,P	q_5,3,P		q_4,2,D	q_4,3,D	q_4,ƀ,P
q_5							

Tabla 6.13: Función de transición de la MT del Ejercicio 31.a.

Observación

Esta máquina es similar a la ya resuelta en el Ejemplo 6.1. La diferencia está en el alfabeto de entrada y que, en este ejercicio, se ha incorporado un estado que corresponde a errores de entrada, por lo que se reconocen las condiciones que provocan movimientos a ese estado.

Solución 31.b

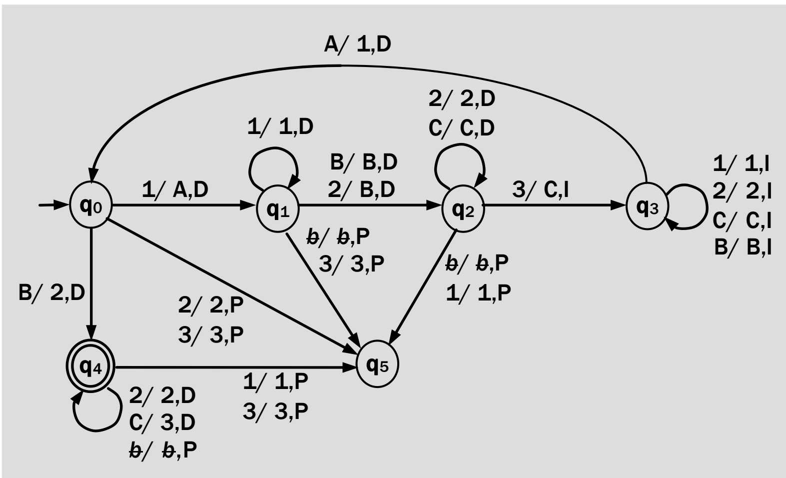


Figura 6.41: Grafo de la máquina de Turing del Ejercicio 31.b.

MT = ({1,2,3},{1,2,3,A,B,C,ƀ },{ q_0, q_1, q_2, q_3, q_4, q_5 }, q_0, { q_4}, *f*, ƀ)

f	1	2	3	A	B	C	~~b~~
$\rightarrow q_0$	q_1,A,D	q_5,2,P	q_5,3,P		q_4,2,D		
q_1	q_1,1,D	q_2,B,D	q_5,3,P		q_2,B,D		q_5,~~b~~,P
q_2	q_5,1,P	q_2,2,D	q_3,C,I			q_2,C,D	q_5,~~b~~,P
q_3	q_3,1,I	q_3,2,I		q_0,1,D	q_3,B,I	q_3,C,I	
$*q_4$	q_5,1,P	q_4,2,D	q_5,3,P			q_4,3,D	q_4,~~b~~,P
q_5							

Tabla 6.14: Función de transición de la MT del Ejercicio 31.b.

Observación

Para reconocer el lenguaje previsto se presenta una variante de la máquina de Turing del ejercicio anterior. La diferencia está en que ahora basta con que la cadena tenga por lo menos un **2**, como separador entre un prefijo de **1s** y un sufijo de **3s**, ambos de la misma longitud. Se invita al lector que analice las diferencias entre ambas máquinas, que identifique las condiciones de error previstas en cada una, las compare, y determine si alguna de estas máquinas es vulnerable a una cadena errónea. En ese caso, justifique su conclusión.

Solución 31.c

Observación

Para resolver este problema se lo desdobla en dos, 1) se verifica que sea correcto el prefijo $\boldsymbol{\alpha=1^m2^n3^m}$ y luego 2) se verifica que sea correcto el sufijo $\mathbf{2^n3^m4^n}$. Para esto, se recurre a dos MT iguales a la que fue propuesta para en el Ejercicio 31.b, que son combinadas para que una opere a continuación de la otra, dando lugar a la máquina cuyo grafo fue representada en la Figura 6.36. Se sugiere representar la función de transición y revisar las condiciones de error de esta máquina. Se trata de un excelente ejemplo de combinación de máquinas de Turing, tema tratado con los conceptos teóricos de este capítulo.

$$MT = (\{1,2,3,4\},\{1, 2, 3, 4, A, B, C, E, \not{b}\},\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9\}, q_0, \{q_4\}, f, \not{b})$$

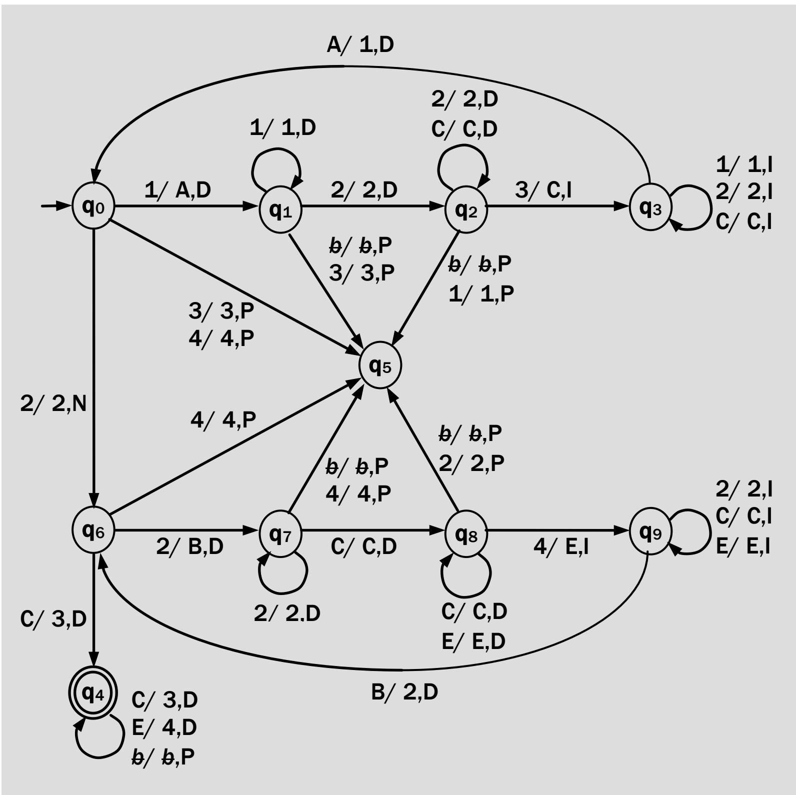


Figura 6.42: Grafo de la máquina de Turing del Ejercicio 31.c.

Ejercicios resueltos de complejidad de Máquina de Turing

Ejercicio 6.32

Se desea conocer la expresión de la complejidad temporal de una máquina de Turing que reconoce un lenguaje de palíndromos **L = {α#α^{-1} / $\alpha \in$ {0, 1}$^+$},** es decir que el sufijo es la cadena inversa del prefijo. Este problema ya fue resuelto en el Ejemplo 6.12, pero en aquella oportunidad se optó por determinar la expresión buscada a partir de resultados obtenidos con simuladores de la MT, al resolver cadenas de cierto largo y por medio del álgebra matricial.

Como se recordará, para validar las cadenas la MT mueve el cabezal hasta el separador **#**, y a partir de allí avanza desde el medio de la cadena hacia los extremos, verificando los símbolos del sufijo y prefijo con sucesivos movimientos de avance y retroceso. Para facilitar la interpretación de la operación de la máquina, con una cadena de cierto largo, que en este caso es $n = 2|\alpha|+1 = |\beta| = 9$, se repite a continuación la Figura 6.21.

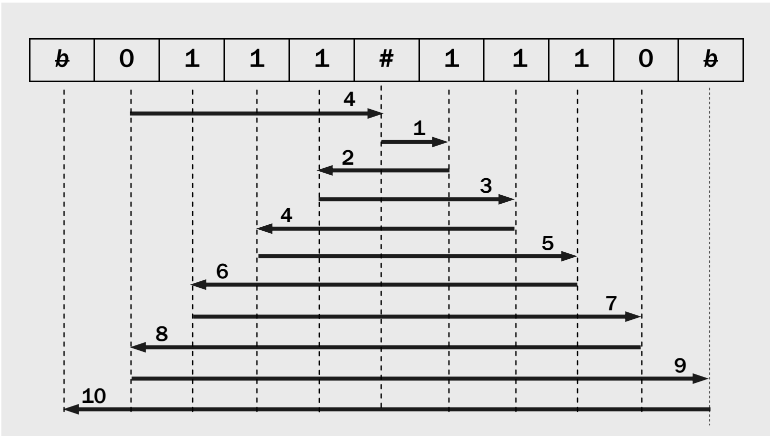


Figura 6.21 (repetida): Esquema del proceso de aceptación de $\alpha\#\alpha^{-1}$.

Se analizan los movimientos para expresarlos en función de ***n***, deduciendo:

- Un primer movimiento, desde el primer símbolo del prefijo hasta el separador de la cadena, requiere de 0,5.(n-1) unidades de tiempo.
- El tiempo que demandan los movimientos de avance desde los sucesivos símbolos del prefijo a los del sufijo se determinan considerando la media de los valores extremos (primero y último) por la cantidad de avances: [1/2.(1 + n)].[1/2.(1+n)].
- El tiempo que demandan los movimientos de retroceso desde los sucesivos símbolos del sufijo a los del prefijo se determinan de la misma manera: [1/2.(2 + n + 1)]. [1/2.(1+n)].

En resumen, combinando las expresiones anteriores se obtiene:

$$T(n) = \frac{n-1}{2} + \left[\left(\frac{1+n}{2}\right) + \left(\frac{2+n+1}{2}\right)\right]\left(\frac{1+n}{2}\right) = \frac{n-1}{2} + \frac{(2n+4)(n+1)}{4} =$$

$$= \frac{n-1}{2} + \frac{2n^2+6n+4}{4} = \frac{1}{2}n^2 + 2n + \frac{1}{2} = O(n^2)$$

Como puede comprobarse, es la misma expresión obtenida en el Ejemplo 6.12 a través del álgebra matricial, o de un gráfico, a partir de los intervalos de tiempo demandados para procesar ciertas cadenas de diferente largo.

//