Modelos de computación

Prácticas de Manuel Enrique López Roldán

Práctica 1.

1. $G = (\{S, A\}, \{a, b\}, P, S)$

$P = \{S \to abAS,\ abA \to baab,\ S \to a,\ A \to b\}$.

Palabras formables:

a, baaba, baabbaaba, abba, abbaabba,
abbbaaba, abbbaabba,

$$L = \{ua,\ u = \{abb, baab\}^*\}$$

Lenguaje de combinaciones de abb, baab, desde cero, siempre terminando en a.

2. $G = (\{N, D\}, \{0 \text{ a } 9\}, P, N\}$

$P = N \to DN,\ N \to D,\ D \to 0 \text{ a } 9$

Lenguaje de combinaciones de números cualesquiera teniendo al menos 2. Es decir se pueden representar todos los números naturales.

$$L = \{u \mid u \in \{0\text{-}9\}^* \mid |u| \geq 2\}$$

Manuel Enrique López Roldán

3. Aplico un cambio de variables:

$a = 👍$, $b = 👎$, $c = ✊$

$L \subseteq \{a, b, c\}^*$. La palabra $w \in L$ no contiene el mismo número de b que de c.

$S \to aS, S_1$

$S_1 \to bS_1c, bcS_1, cbS_1, S_1bc, S_1cb, S_2, aS_1, cS_1b$

$S_2 \to aS_2, S_3, S_4$

$S_3 \to aS_3, cS_3, c$

$S_4 \to aS_4, bS_4, b$

4.

a.) Palabras que comienzan con 000 y terminan en 111

$S \to 000S_1$

$S_1 \to 0S_1, 1S_1, S_2$

$S_2 \to 111$

Es un lenguaje regular.

b.) Palabras que no contengan dos 0 seguidos.

$S \to 1S, 0S_1, S_1$

$S_1 \to 1S, \varepsilon$

Es un lenguaje regular.

C.) Palabras que si contienen 000, le siga al menos un 1.

$S \to 1S, 0S_1, \varepsilon$
$S_1 \to 1S_1, 0S_2, \varepsilon$
$S_2 \to 1S_2, 0S_3, \varepsilon$
$S_3 \to 1S$

5. $S \to gS_1, pS_3, tS, x$
$S_1 \to fddS_2, ddfS_2, dfdS_2$
$S_2 \to fS_2, dS_2, sS$
$S_3 \to dS_3, fS, S$

Práctica 2: Lex

Mi programa básicamente quiere buscar en pccomponentes sus ofertas y compararlas con las de amazon, ya que muchas veces el descuento no se corresponde, que quiere decir esto, pues que cuando tu entras a pccomponentes y ves algo al 50% de descuento y vale 30€ esperas que su precio original fuera de 60€ sin embargo lo buscas en amazon y vale 32€, simplemente intentan que lo compres por un descuento que es irreal. Para resolver este problema está mi programa.

Mi programa en realidad son 4, uno para bajar la información de los artículos, otro para quitar la posible basura que tuviera el primero, y almacenar lo esencial, además de ordenar por descuento y llamar al tercer programa que es el encargado de buscar en amazon, y un cuarto por si no se encuentra el artículo, acorta la búsqueda, qué quiere decir esto? que por ejemplo el nombre de un artículo puede estar compuesto por 5 palabras, “LG55tVRD televisión de 55’ 4k”, y si buscas esto en amazon es posible que no aparezca, sin embargo si buscas “LG55tVRD televisión”, es posible que aparezca algún resultado, en esto se basa la búsqueda de error, sólo busca las dos primeras palabras, evidentemente puede fallar si por ejemplo el nombre del artículo era “Adaptador Wifi W-link 4000”, las dos primeras palabras pocos nos especifican pero la verdad es que este tipo de búsqueda, da muy buenos resultados. Y es la mejor opción que se me ha ocurrido que sea relativamente simple de utilizar.

Si que he encontrado un problema que es que amazon me corta a partir de ciertas peticiones y empiezan a salir errores por pantalla, pero al final hace el archivo correctamente, y acaba bien, la velocidad del programa depende de la velocidad de la conexión a internet, así que si está tardando mucho se puede poner una restricción en el código o simplemente cerrar la terminal, y comprobar el archivo resultadoOrdenado_pccomponentes en la carpeta data y ahí aparecerá la información conseguida.

Si se quiere poner un tope, en el bucle de parseo.cpp hay una variable tope=0; antes del bucle, que va aumentando, en el mismo bucle se añade la comprobación que tope sea menor que 100 y acaba mucho más rápido.

Aquí aparecen los comentarios de cada programa:

flex.cpp:

```
        * Primer programa y el que ejecuta el resto, busca en el HTML de pccomponentes
        * en concreto en las ofertas, los artículos en oferta, junto con su precio
        * el descuento y el precio anterior, para ello primero descarga el HTML
        * crea un archivo con él, lee ese archivo, y el resultado macheado por las
        * expresiones regulares se guarda en otro archivo para volverse a analizar
        * ya que hay "basura" o elementos que no nos interesan y que también ha
macheado.
```

parseo.cpp:

```
        * Segundo programa y busca en el HTML de pccomponentes ya descargado
        * las ofertas, los artículos en oferta, junto con su precio
        * el descuento y el precio anterior, pero esta vez sí, quitando la parte
        * que no nos resulta útil, y añadiendo cierto formato, además
```

* los porcentajes y los nombres se guardan en un multimap, para que se
* ordene automáticamente, para poder hacer el fichero más interesante
* ordenado por descuento y llama a busqueda_amazon pasandole el nombre
* del artículo, el precio y el descuento.

busqueda_amazon.cpp

* Tercer programa, busca en el HTML de amazon el precio de un artículo
* encontrado en pccomponentes, y si lo encuentra coge su precio y lo añade
* al fichero resultado "resultadoOrdenado_pccomponentes, sino lo encuentra
* llama a busqueda_amazonErr, pasándole el nombre del artículo, el precio y el descuento

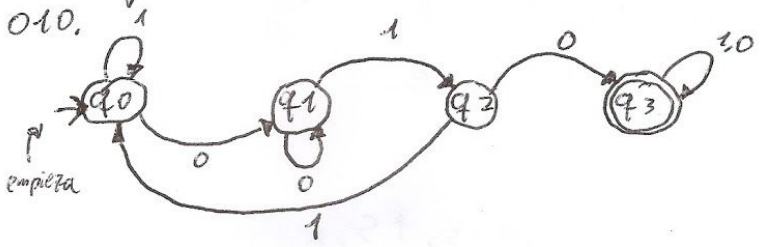
busqueda_amazonErr

* Cuarto y último programa, busca en el HTML de amazon el precio de un artículo
* encontrado en pccomponentes, y si lo encuentra coge su precio y lo añade
* al fichero resultado "resultadoOrdenado_pccomponentes, realmente hace
* lo mismo que el anterior pero con una variante, como el nombre del artículo
* no se encontró reducimos su nombre a las dos primeras palabras para ver si
* así lo encuentra.

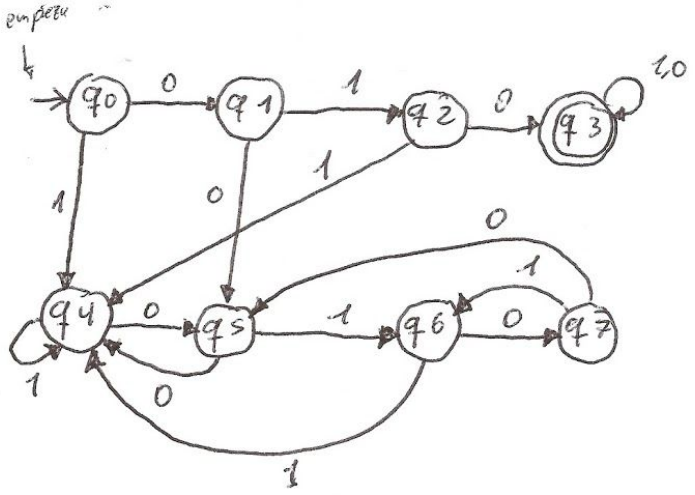
Práctica 3.

1.

a.) El lenguaje de las palabras que contienen la subcadena 010.

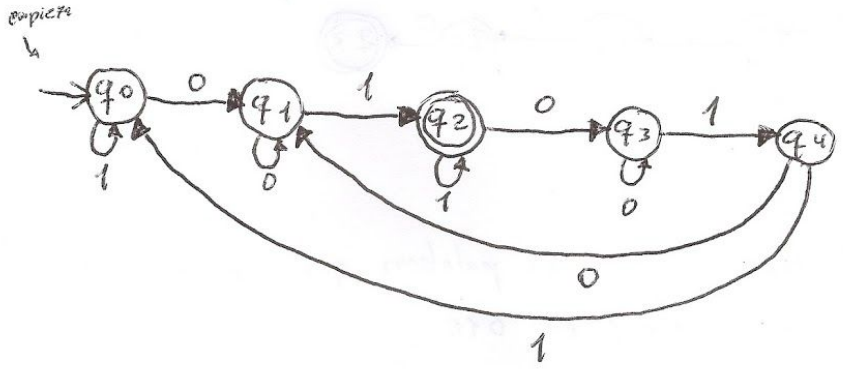


b.) El lenguaje de las palabras que empiecen o terminen (o ambas cosas) en 010.

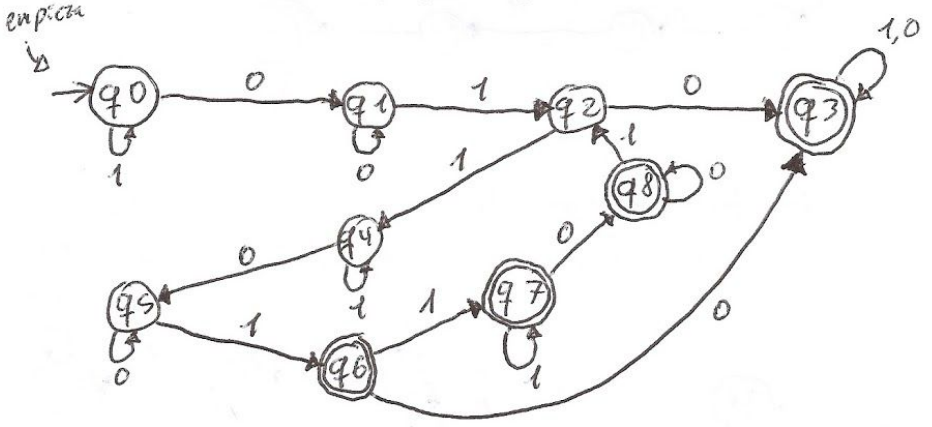


Manuel Enrique López Roldán

c.) El lenguaje $L \subseteq \{0,1\}^*$ de todas las palabras con un número impar de ocurrencias de la subcadena 01.

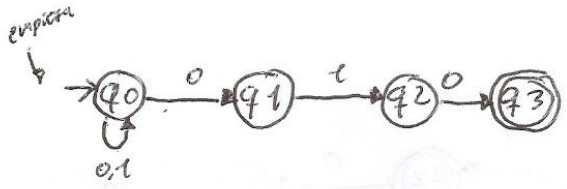


Al principio me equivoqué e hice un AFD, que cumpliese a), b) y c), lo pongo, porque como ya lo hice, no me cuesta nada:

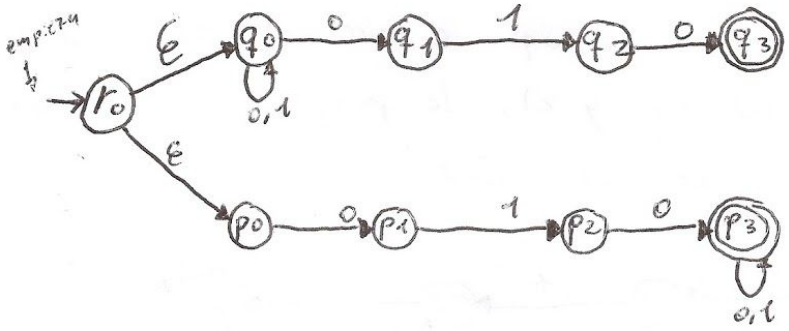


2.

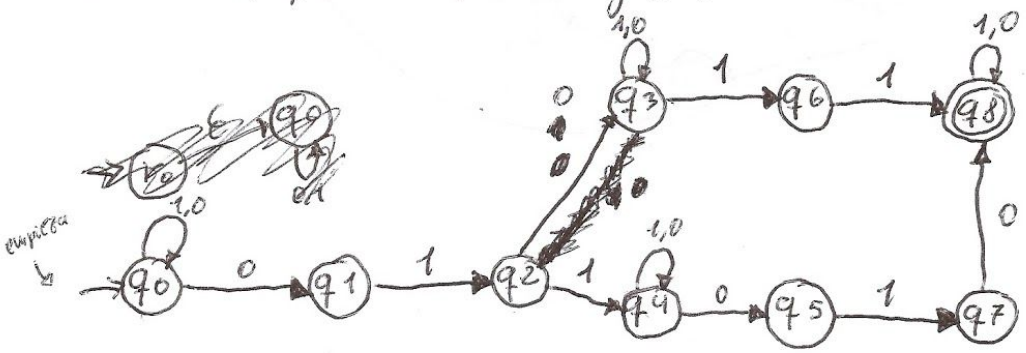
a.) El lenguaje de las palabras que terminan en 010.



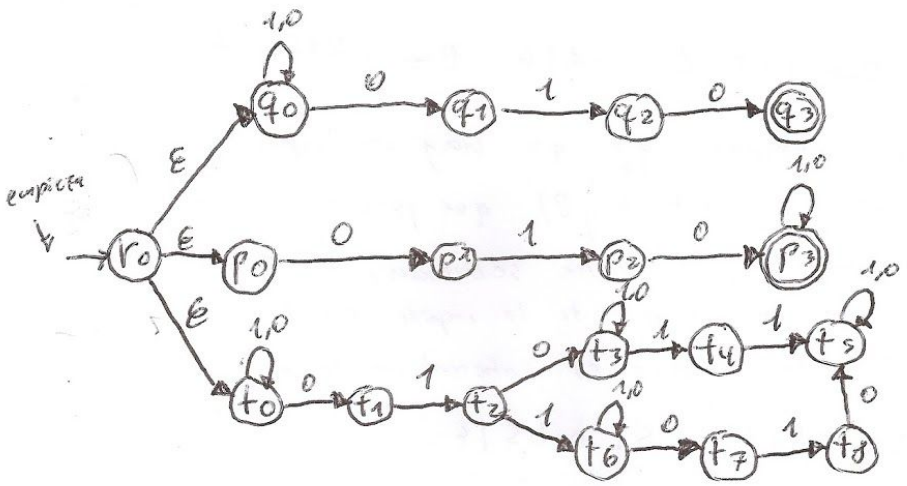
b.) El lenguaje de las palabras que empiezan o terminan (o ambas cosas) en 010.



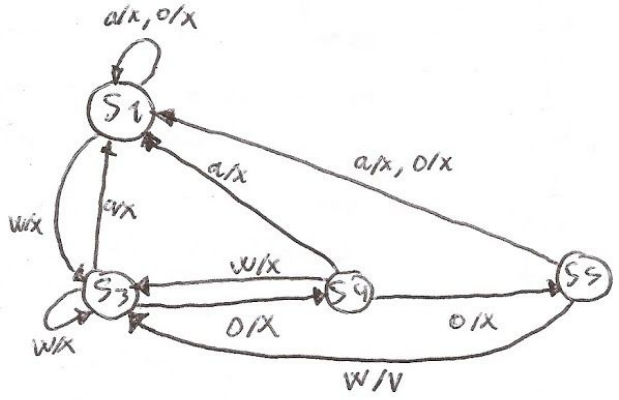
c.) El lenguaje que contenga, las subcadenas 010 y or 1, a la vez, pueden compartir 0 y 1. (realmente solo el 0)



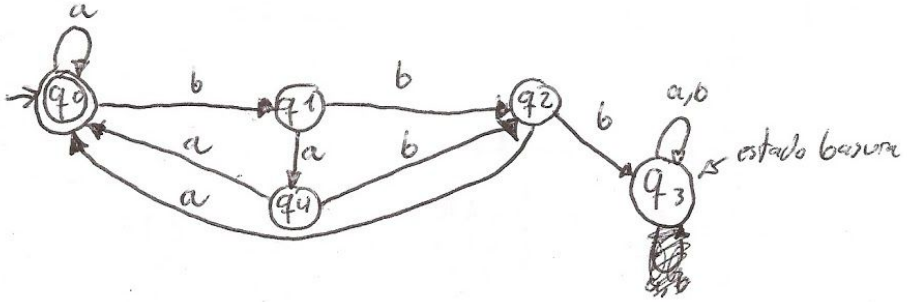
También hice los tres a la vez:



3.



4.



Práctica 4.

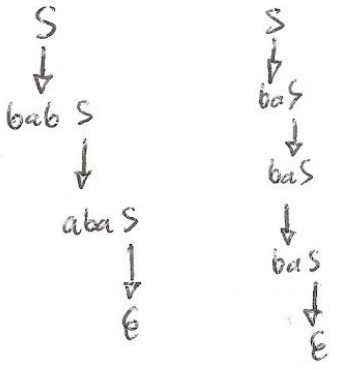
1.

a.) $S \to AbB \quad A \to aA \mid \epsilon \quad B \to aB \mid bB \mid \epsilon$

No es ambigua, ya que hay una especie de separador (la b, entre A y B), que provoca que no puedas repetir la misma secuencia, lo que podrías hacer con la a, pero te lo impide esa b.
Por lo tanto tumbaco es inherentemente ambigua

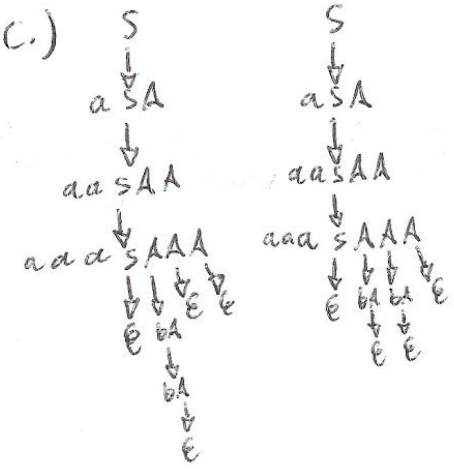
b.) $S \to abaS \mid baS \mid babS \mid \epsilon$

bababa



La gramática es ambigua como se puede ver, aunque no es inherentemente ambigua ya que ~~para~~ lo puedes generar con:
$S \to abS_1 \mid baS_2 \mid \epsilon$
$S_1 \to aS, \quad S_2 \to bS$
que no es ambigua.

c.)



aaabb
La gramática es ambigua como se puede ver, aunque no es inherentemente ambigua ya que lo puedes generar con:
$S \to aS \mid \epsilon \mid S_1$
$S_1 \to bS_1 \mid \epsilon$

2. $L = \{ x^i y^j z^k \mid i+k=j \; ; \; i,j,k \in \mathbb{N} \}$

$\delta(q_0, x, z_1) = \{(q_0, x z_1)\}$

$\delta(q_0, x, x) = \{(q_0, xx)\}$

$\delta(q_0, y, x) = \{(q_1, \epsilon)\}$

$\delta(q_0, y, z_1) = \{(q_1, [illegible])\}$

$\delta(q_1, y, x) = \{(q_1, \epsilon)\}$

$\delta(q_1, y, z_1) = \{(q_1, y z_1)\}$

$\delta(q_1, y, y) = \{(q_1, yy)\}$

$\delta(q_1, z, y) = \{(q_2, \epsilon)\}$

~~$\delta(q_1, z, z_1) = \{(q_2, z_1)\}$~~

~~$\delta(q_1$~~

$\delta(q_2, z, y) = \{(q_2, \epsilon)\}$

$\delta(q_2, \epsilon, z_1) = \{(q_2, \epsilon)\}$

$\delta(q_0, \epsilon, z_1) = \{(q_0, \epsilon)\}$

3. $S \to a S b \mid b S a \mid \epsilon$

$L = \{ a^i b^i \mid i \geq 0 \} \cup \{ b^i a^i \mid i \neq 0 \}$

4.

a.) $L_1 = \{(ab)^i (bc)^j \mid i,j \geq 0\}$

$S \to AB$

$A \to aAb \mid \epsilon$

$B \to bBc \mid \epsilon$

b.) $L_2 = \{a^i b^j c^{i+j} \mid i,j \geq 0\}$

$S \to aSc \mid S_1 \mid \epsilon$

$S_1 \to bS_1c \mid \epsilon$

c.) $L_3 = \{a^i b^j c^j d^i \mid i,j \geq 0\}$

$S \to aSd \mid S_1 \mid \epsilon$

$S_1 \to bS_1c \mid \epsilon$

d.) $S \to aabS_1$

$S_1 \to aS_2 \mid bS_4 \mid cS_1 \mid F$

$S_2 \to aS_3 \mid bS_1 \mid cS_1 \mid F$

$S_3 \to aS_3 \mid cS_1$

$S_4 \to bS_5 \mid aS_2 \mid F$

$S_5 \to bS_5 \mid aS_2 \mid F$

$F \to bbc$

Ejercicios adicionales

Hacer de tipo 3:

a) subcadena que no contiene 666

b) tiene 2 o 3 a, no más, ni menos.

c) número de a no es 2

a) $S \to aS \mid bS_1 \mid bbS_1 \mid \varepsilon$
$S_1 \to aS \mid \varepsilon$

b) $S \to aS_1 \mid bS$
$S_1 \to aS_2 \mid bS_1 \mid S_2$
$S_2 \to bS_2 \mid aS_3$
$S_3 \to bS_3 \mid \varepsilon$

c) $S \to bS \mid aS_1 \mid \varepsilon$
$S_1 \to bS_1 \mid aS_2 \mid \varepsilon$
$S_2 \to bS_2 \mid aS_3$
$S_3 \to bS_3 \mid aS_3 \mid \varepsilon$

$L = \{ avb \mid u \in A^* \} \cup \{ (aa)^i w \mid w \in A^*, i \in \mathbb{N} \}$

$S \to aS_1$
$S_1 \to aaS_1 \mid S_2$
$S_2 \to b$

Manuel Enrique López Roldán 1/12

Minimizar