Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI3725 - Traductores e Interpretadores
Septiembre-Diciembre 2014

Carnet:______________________________

Nombre:______________________________

Examen I

(20 puntos)

Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4	Pregunta 5	**Total**
2 puntos	6 puntos	6 puntos	3 puntos	3 puntos	**20 puntos**

Pregunta 1 - 2 puntos

Construya expresiones regulares para los lenguajes que se indican a continuación. Solamente puede utilizar concatenación, unión, clausura reflexo-transitiva (Estrella de Kleene), o notación de conjuntos. En ambos casos, los lenguajes se construyen sobre el alfabeto $\Sigma = \{a, b\}$.

1. **(1 punto)** Palabras en las cuales nunca hay tres a seguidas, excepto al principio.

$$(a + aa + aaa + \lambda)(b + ba + baa)*$$

2. **(1 punto)** Palabras de longitud impar que contienen ab y ba.

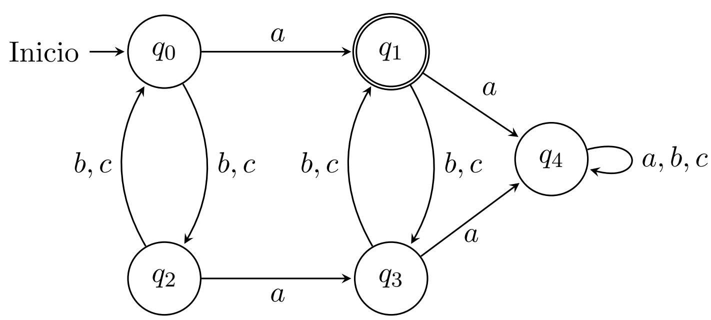
   Sean $E = ((a + b)(a + b))*$ [1] y $O = E(a + b)$ [2] que nos permiten escribir

$$EabEbaO{+}EabObaE{+}OabEbaE{+}EbaEabO{+}EbaOabE{+}ObaEabE{+}E(aba{+}bab)E{+}O(aba{+}bab)O$$

Pregunta 2 - 6 puntos

Sea el alfabeto $\Sigma = \{a, b, c\}$.

1. **(2 puntos)** Construya un AFD que reconozca el lenguaje de las palabras sobre Σ^* de longitud impar que contengan *exactamente* una a.



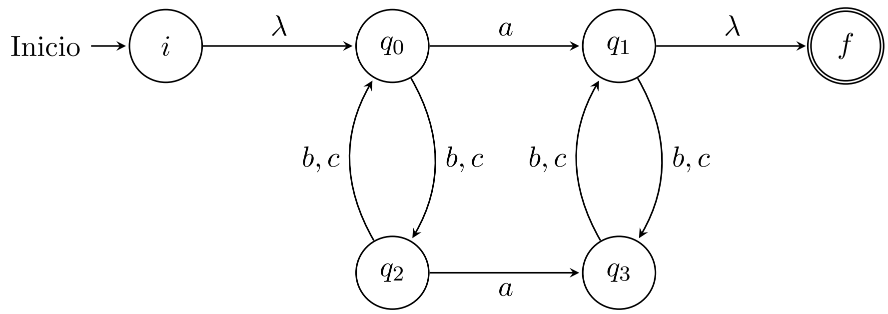
El AFD que se presenta es *completo*. Como el enunciado no lo exige, el estado q_4 podría omitirse y quedar incompleto.

[1] E por *even*, denota palabras de longitud par.

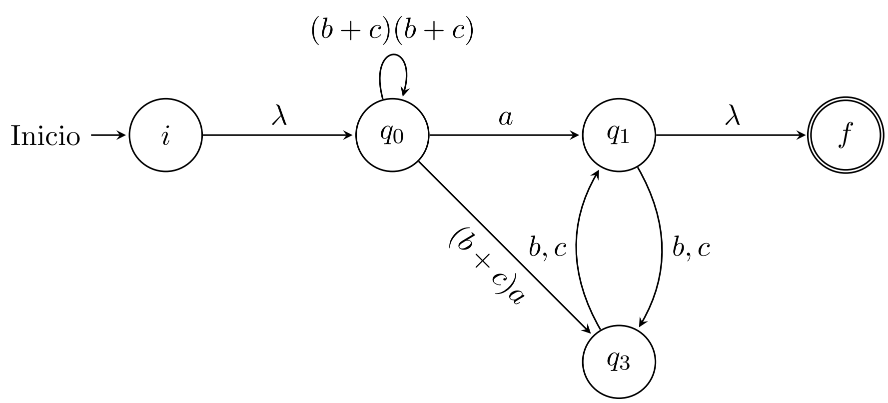
[2] O por *odd*, denota palabras de longitud impar.

2. **(4 puntos)** Calcule la expresión regular que denote el lenguaje reconocido por el AFD recién construido usando el algoritmo de reducción de expresiones, indicando cada paso. *Nota:* si bien no es obligatorio, se sugiere simplificar las expresiones en cada paso de transformación para ahorrar tiempo.

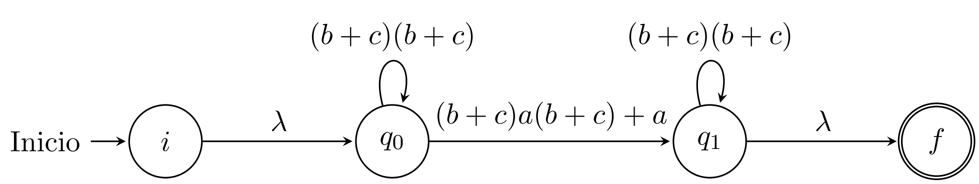
   Eliminamos el estado sumidero q_4. Agregamos un nuevo estado inicial i, con una $\lambda-$transición hacia el estado inicial original. Agregamos un nuevo estado final f, con $\lambda-$transición desde el estado final original.



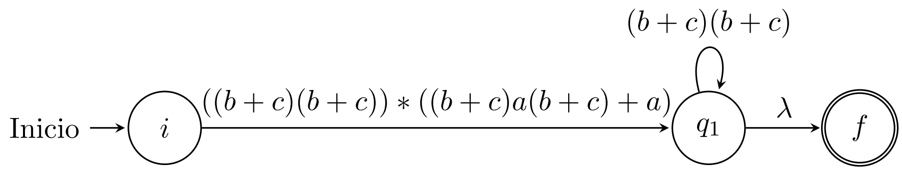
   Eliminamos el estado q_2. Es necesario preservar un camino entre q_0 y q_0 con la expresión $(b+c)(b+c)$, y un camino entre q_0 y q_3 con la expresión $(b+c)a$.



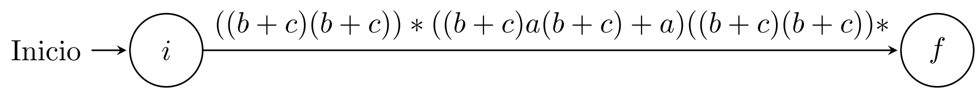
   Eliminamos el estado q_3. Es necesario preservar un camino entre q_0 y q_1 con la expresión $(b+c)a(b+c)$ combinado con el camino ya existente con expresión a, y un camino entre q_1 y q_1 con la expresión $(b+c)(b+c)$.



Eliminamos el estado q_0. Es necesario preservar un camino entre i y q_1 con la expresión $((b+c)(b+c))*((b+c)a(b+c)+a)$



Eliminamos el estado q_1. Es necesario preservar un camino entre i y f



que corresponde a la expresión definitiva que estamos buscando

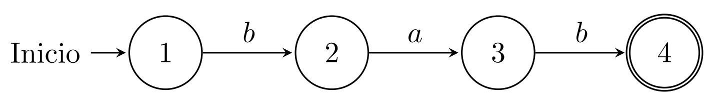
$$((b+c)(b+c))*((b+c)a(b+c)+a)((b+c)(b+c))*$$

Pregunta 3 - 6 puntos

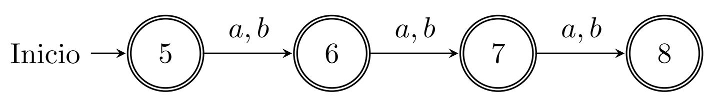
Sea el alfabeto $\Sigma = \{a, b\}$.

1. **(0.75 puntos)** Construya sendos autómatas finitos *no-determinísticos,* usando $\lambda-$transiciones si le resulta conveniente, que reconozcan las expresiones regulares correspondientes a los conjuntos:

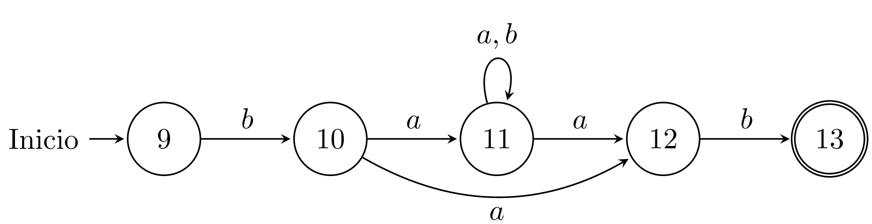
   - L_1 de la palabra *bab*.



   - L_2 de las palabras en Σ^* cuya longitud es menor o igual a tres.

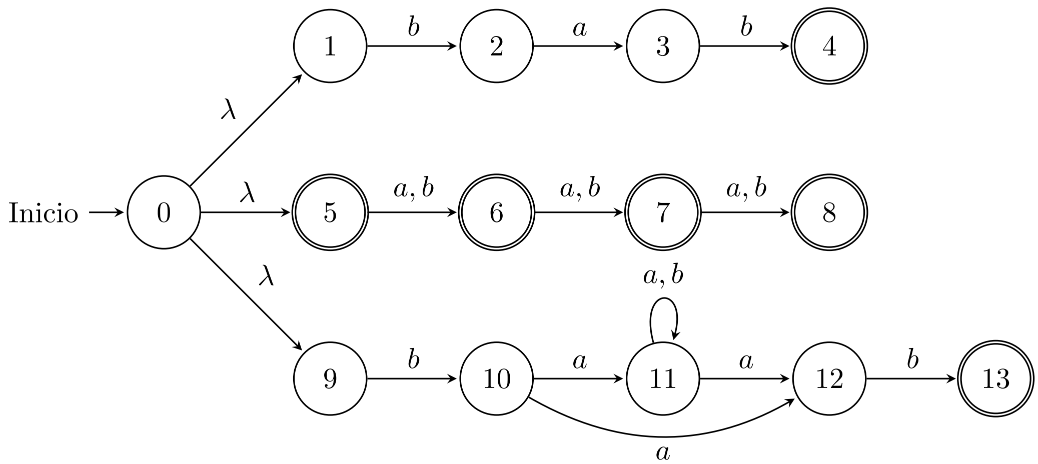


   - L_3 de las palabras en Σ^* que comienzan con *ba* y terminan con *ab*.

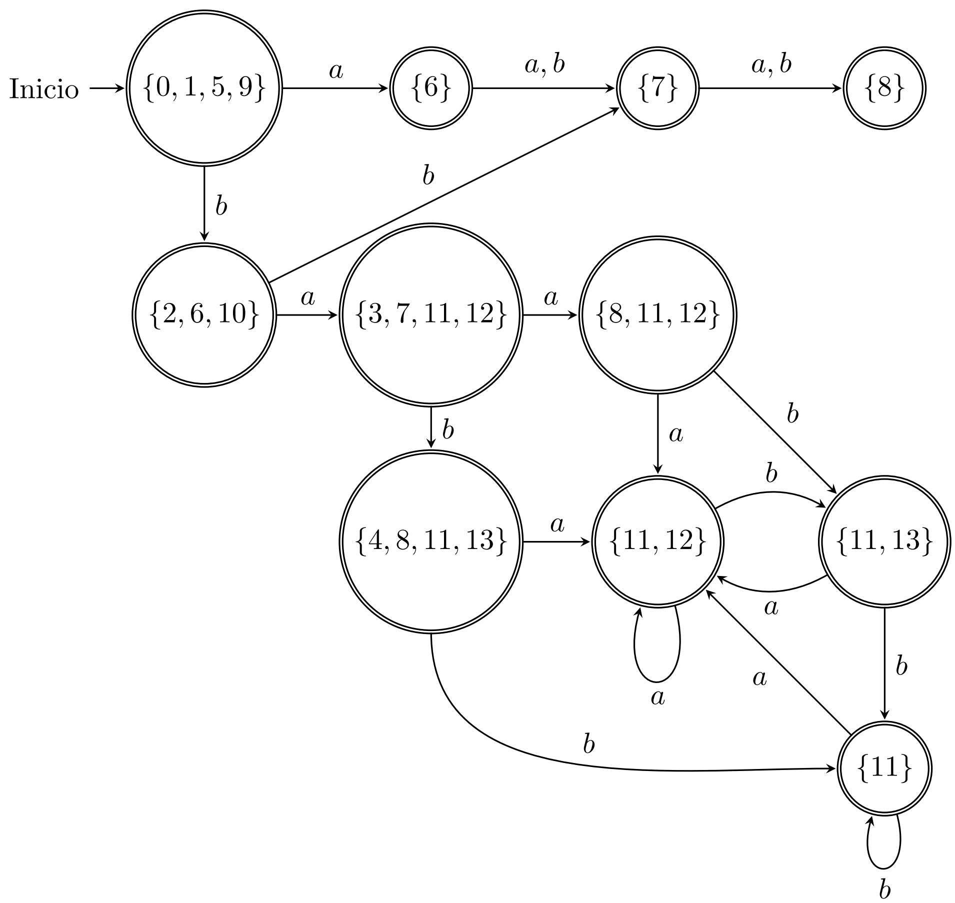


   Basta la representación gráfica de cada uno de los autómatas.

2. **(0.25 puntos)** Combine los tres autómatas para crear un AFN-λ que reconozca la unión de los tres lenguajes anteriores, de manera que al procesar alguna cadena de entrada y reconocerla, se pueda saber a cuál de los tres lenguajes pertenece.



3. **(4 puntos)** Convierta el AFN-λ construido en un AFD.



4. **(1 punto)** Indique a cuál de los lenguajes originales corresponde cada estado final del AFD. En caso de ambigüedad, se prefieren las palabras de L_1 antes que las palabras de L_2, y se prefieren las palabras de L_2 antes que las palabras de L_3.

   - Si el AFD acepta en el estado $\{4, 8, 11, 13\}$, entonces está aceptando una palabra que corresponde a L_1, i.e. la palabra *bab*, pues en el $\lambda-$AFN original el estado 4 es de aceptación para L_1, el estado 8 es de aceptación para L_2, y el estado 13 es de aceptación para L_3, pero las precedencias indican que debemos preferir L_1.
   - Si el AFD acepta en los estados $\{0, 1, 5, 9\}$, $\{6\}$, $\{7\}$, $\{8\}$, $\{2, 6, 10\}$ o $\{8, 11, 12\}$, entonces está aceptado una palabra que corresponde a L_2, i.e. palabras de longitud menor o igual a tres, pues en el $\lambda-$AFN original los estados 5, 6, 7 y 8 son de aceptación para dicho lenguaje.
   - Si el AFD acepta en los estados $\{11, 13\}$ entonces está aceptando una palabra que corresponde a L_3, i.e. las palabras que comienzan con *ba* y terminan en *ab*, pues en el $\lambda-$AFN original el estado 13 es de aceptación para L_3.

Pregunta 4 - 3 puntos

Construya el AFD mínimo equivalente, mostrando y justificando la construcción de los $\equiv_i$ necesarios, para el AFD definido por la 5-tupla

$$M = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}, \{a, b\}, \delta, q_0, \{q_5, q_6\})$$

δ	a	b
q_0	q_2	q_1
q_1	q_3	q_0
q_2	q_4	q_5
q_3	q_4	q_5
q_4	q_6	q_5
q_5	q_5	q_5
q_6	q_6	q_5

Por definición, la clase de equivalencia $\equiv_0$ tiene dos conjuntos, el de estados finales y el de estados no finales, por tanto

$$\equiv_0 = \{\{q_0, q_1, q_2, q_3, q_4\}, \{q_5, q_6\}\}$$

Para calcular $\equiv_1$ consideramos:

1. Los estados q_0 y q_1 **si** son equivalentes puesto que $\delta(q_0, a) \equiv_0 \delta(q_1, a) \wedge \delta(q_0, b) \equiv_0 \delta(q_1, b)$.
2. Los estados q_0 y q_2 **no** son equivalentes puesto que $\delta(q_0, b) \not\equiv_0 \delta(q_2, b)$.
3. Los estados q_0 y q_3 **no** son equivalentes puesto que $\delta(q_0, b) \not\equiv_0 \delta(q_3, b)$.
4. Los estados q_0 y q_4 **no** son equivalentes puesto que $\delta(q_0, a) \not\equiv_0 \delta(q_4, a)$.
5. Los estados q_2 y q_4 **no** son equivalentes puesto que $\delta(q_2, a) \not\equiv_0 \delta(q_4, a)$.
6. Los estados q_2 y q_3 **si** son equivalentes puesto que $\delta(q_2, a) \equiv_0 \delta(q_3, a) \wedge \delta(q_2, b) \equiv_0 \delta(q_3, b)$.
7. Los estados q_5 y q_6 **si** son equivalentes puesto que $\delta(q_5, a) \equiv_0 \delta(q_6, a) \wedge \delta(q_5, b) \equiv_0 \delta(q_6, b)$.

en consecuencia

$$\equiv_1 = \{\{q_0, q_1\}, \{q_2, q_3\}, \{q_4\}, \{q_5, q_6\}\}$$

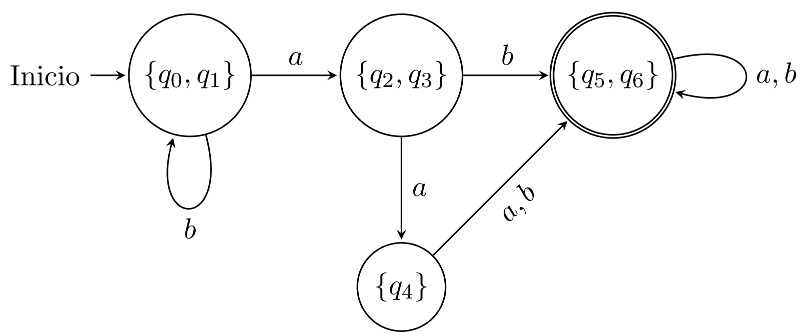
Para calcular $\equiv_2$ consideramos:

1. Los estados q_0 y q_1 **si** son equivalentes puesto que $\delta(q_0, a) \equiv_1 \delta(q_1, a) \wedge \delta(q_0, b) \equiv_1 \delta(q_1, b)$.
2. Los estados q_2 y q_3 **si** son equivalentes puesto que $\delta(q_2, a) \equiv_1 \delta(q_3, a) \wedge \delta(q_2, b) \equiv_1 \delta(q_3, b)$.
3. Los estados q_5 y q_6 **si** son equivalentes puesto que $\delta(q_5, a) \equiv_1 \delta(q_6, a) \wedge \delta(q_5, b) \equiv_1 \delta(q_6, b)$.

en consecuencia

$$\equiv_2 = \{\{q_0, q_1\}, \{q_2, q_3\}, \{q_4\}, \{q_5, q_6\}\}$$

Como $\equiv_2 = \equiv_1$ hemos llegado al punto fijo de las clases de equivalencia. Tendremos un estado por cada clase de equivalencia, de manera que el AFD mínimo resultante será



Pregunta 5 - 3 puntos

Sea el alfabeto $\Sigma = \{a, b\}$ y $L = \{a^n b^m \mid n/m \in \mathbb{N}\}$. Use el Lema de Bombeo de Lenguajes Regulares para demostrar que L no es regular.

Supongo que L es Lenguaje Regular, por lo tanto existe un autómata finito $M = (Q, \Sigma, \delta, q_0, F)$ con $|Q| = k$ que acepta precisamente las palabras de L. El Lema de Bombeo para Lenguajes Regulares garantiza que $\forall z \in L$ con $|z| \geq k$ siempre se puede descomponer $z = uvw$ tal que $|uv| \leq k$, $|v| > 0$, y luego $\forall i \geq 0$ se cumple $uv^i w \in L$.

Consideremos la palabra $z = a^k b^k \in L$, entonces cualquier partición de z que cumpla con las condiciones del Lema de Bombeo para Lenguajes Regulares debe tener necesariamente $v = a^p$ con $0 < p \leq k$ siendo de la forma

$$a^q a^p a^{k-p-q} b^k$$

Ahora bien, dado cualquier p, consideremos lo que ocurre al bombear cuando $i = 0$. En este caso, la palabra tendrá la forma

$$a^{k-p} b^k$$

y como $0 < p \leq k$ sigue que $k - p < k$ y por tanto $(k - p)/k \notin \mathbb{N}$.

Mostramos que para cualquier partición hay precisamente un i para el cual la palabra resultante no pertenece a L, contradiciendo el Lema de Bombeo para Lenguajes Regulares. Esa contradicción es consecuencia de haber supuesto que L era en efecto un Lenguaje Regular, así que no puede serlo.