Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información.
CI–3725 – Traductores e Interpretadores
Abril–Julio 2012

Examen I

(25 puntos)
Solución

0. (15 puntos) Considere cuidadosamente cada uno de los siguientes enunciados y seleccione exactamente una respuesta de entre las opciones disponibles, que haga cierto el enunciado en cuestión. Cada respuesta correcta **suma un punto y medio (1.5)**, sin embargo **tres (3) respuestas incorrectas cancelan una correcta**. Tiene la opción de contestar la opción *No sabe / No contesta*, lo cual anula la puntuación de dicha respuesta, mas tampoco es penalizada. Cualquier enunciado que no sea respondido, o que sea respondido con dos o más respuestas, se considerará incorrecto.

   (a) Siendo L un Lenguaje cualquiera, se puede asegurar siempre que:

      i. $L \cap S$ es un Lenguaje Regular, si S es Regular.

         No siempre es el caso, por ejemplo sea $L = \{a^n b^n \mid n \geq 0\}$ y $S = \{a, b\}^*$. S es regular, pero $L \cap S$ no lo es.

      ii. Si L cumple con el Lema de Bombeo, entonces L es Regular.

         No, el recíproco es el que es cierto. Cumplir el Lema de bombeo es condición necesaria para que un Lenguaje sea Regular, pero no suficiente.

      iii. Si $L = S \cup T$ y tanto S como T no son Regulares, entonces L tampoco es Regular.

         No siempre es el caso, por ejemplo $S = \{a^n b^n \mid n \geq 0\}$ y $T = \{a^i b^j \mid i, j \geq 0 \wedge i \neq j\}$. Notemos que ni S ni T son regulares, sin embargo $S \cup T = \{a, b\}^*$ si lo es.

      iv. [[**Si L es Regular, entonces existe un Autómata Finito M, con un solo estado final, tal que $\mathcal{L}(M) = L$**]]

         Si L es Regular, siempre habrá un Autómata Finito que lo reconozca. Dicho autómata puede transformarse siempre en uno de un solo estado final, colocando un solo nuevo estado q_f como dicho final y creando λ–transiciones desde cada uno de los finales originales, hasta q_f.

      v. *No Sabe / No Contesta.*

   (b) ¿Cuándo *no* se puede afirmar que un Lenguaje L es Regular?

      i. Cuando es reconocido por un Autómata Finito Determinista.

         No, todo Lenguaje que tiene un Autómata Finito Determinista que lo reconozca, es Regular.

      ii. Cuando es representado por una Expresión Regular.

         No, todo Lenguaje que tiene una Expresión Regular que lo represente, es Regular.

      iii. [[**Cuando es generado por una Gramática.**]]

         Si, pues nunca se especifica si la Gramática es Regular o no. Por lo tanto no se puede afirmar nada.

      iv. Cuando es reconocido por un Autómata Finito No Determinista.

         No, todo Lenguaje que tiene un Autómata Finito No Determinista que lo reconozca, es Regular.

      v. *No Sabe / No Contesta.*

(c) ¿Cuál de los siguientes lenguajes, sobre el alfabeto $\Sigma = \{0, 1\}$, no es regular?

i. $\{0x1y0 \mid x, y \in \Sigma^* \wedge |x| \leq 3 \wedge |y| > 3\}$

No, pues x y y son independientes, lo cual hace regular este Lenguaje.

ii. [[$\{w \mid w \in \Sigma^* \wedge$ **"w no es palindrome"**$\}$]]

Si, saber si una palabra es palíndrome o no depende de un contexto, por lo que el Lenguaje no puede ser Regular.

iii. $\{w \mid w \in \Sigma^* \wedge 5 \leq |w| \leq 7 \wedge$ "w contenga a lo sumo dos 1's"$\}$

No, pues la intersección de dos Lenguajes Regulares es Regular.

iv. $\{w \mid w \in \Sigma^* \wedge$ "w tiene exactamente cuatro 0's o exactamente cuatro 1's"$\}$

No, pues la unión de dos Lenguajes Regulares es Regular.

v. *No Sabe / No Contesta.*

(d) Considere la siguiente expresión regular sobre el alfabeto $\Sigma = \{a, b\}$, $e = a(a + b)* ab$. ¿Qué puede asegurarse sobre la semántica de e?

i. [[$sem(e) \cdot sem(e) \subseteq sem(e)$]]

Si, pues $sem(e) \cdot sem(e) = sem(ee) = sem(a(a + b)* aba(a + b)* ab)$, es decir todas las palabras que empiezan con "a", terminan con "ab" y contienen la cadena "aba" al menos una vez. Esto es un subconjunto de $sem(e)$, que relaja la última condición.

ii. $sem(e) \subseteq D_b(sem(e)^R)$

No, el reverso de $sem(e)$ sería todas las palabras que empiezan con "ba" y terminan con "a". Al derivarlo con respecto a "b", quedarían todas las palabras que empiezan con "a" y terminan con "a", lo cual no tiene relación de contenencia con $sem(e)$.

iii. $D_a(sem(e)) \subseteq sem(e)$

No, derivar $sem(e)$ con respecto a "a" resultaría en todas las palabras que terminan con "ab", sin restricción sobre como empiecen. Nótese que en realidad lo que se cumple es la relación de contención recíproca: $sem(e) \subseteq D_a(sem(e))$.

iv. $sem(e) \cup sem(e)^R = \Sigma^*$

No, pues unir $sem(e)$ con su reverso, resultaría en las palabras que si empiezan con "a" deben terminar con "ab" y si empiezan con "b", el segundo símbolo debe ser "a" y debe terminar con "a".

v. *No Sabe / No Contesta.*

(e) Considere una Gramática Regular $G = (\Sigma, V, P, S)$ donde todas las producciones en P son de la forma $A \to xB$, con $A \in V,\ x \in \Sigma,\ B \in (V \cup \{\lambda\})$. ¿En cuantos pasos de derivación genera G, una palabra de longitud n que pertenezca a $\mathcal{L}(G)$?

i. [[n]]
ii. $n + 1$
iii. Depende de la palabra.
iv. Depende de la presencia o falta de símbolos recursivos (directa o indirectamente).
v. *No Sabe / No Contesta.*

Cada símbolo terminal que se genera viene de exactamente un paso de derivación, por lo que palabras de tamaño n deben ser generadas en n pasos. Esto, siempre y cuando pertenezcan a $\mathcal{L}(G)$.

(f) Considere el siguiente Lenguaje Regular, sobre el alfabeto $\Sigma = \{\clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$, donde cada frase debe contener *al menos* una ocurrencia de la subfrase "$\clubsuit\diamondsuit\heartsuit\spadesuit$" (sin las comillas). ¿Cuantos estados tiene como mínimo un Autómata Finito Determinista que reconozca este lenguaje?

i. 4
ii. [[5]]
iii. 8
iv. Ninguna de las anteriores.
v. *No Sabe / No Contesta.*

El Autómata Finito Determinista Mínimo para este lenguaje es 5. Nótese que esta ya era una cota inferior pues para reconocer una frase como "$\clubsuit\diamondsuit\heartsuit\spadesuit$", que es de longitud 4, hacen falta 4 transiciones (y por ende 5 estados).

(g) En el proceso de traducción/interpretación de un Lenguaje de Programación:

i. El *Análizador Léxico* entiende el lenguaje y el *Análizador Sintáctico* lo procesa.

No, ambos entienden el lenguaje. Solo que a distintos niveles cada uno.

ii. El *Análizador Sintáctico* entiende el lenguaje y el *Análizador Léxico* lo procesa.

No, ambos entienden el lenguaje. Solo que a distintos niveles cada uno.

iii. El *Back End* entiende el lenguaje y el *Front End* lo procesa.

No, el Front End *es quien lee y tranforma el Lenguaje en una representación intermedia, para que luego el* Back End *lo utilice para finalizar el proceso de traducción/interpretación.*

iv. [[**El *Front End* entiende el lenguaje y el *Back End* lo procesa.**]]

Si, el Front End *es quien lee y tranforma el Lenguaje en una representación intermedia, para que luego el* Back End *lo utilice para finalizar el proceso de traducción/interpretación.*

v. *No Sabe / No Contesta.*

(h) ¿Cuál de las siguientes afirmaciones es *falsa*?

i. Si un Autómata Finito Determinista M, con k estados, reconoce una palabra de longitud mayor o igual que k, entonces $\mathcal{L}(M)$ es infinito.

No, es cierta. Esta afirmación es justamente la base para la formulación del Lema de Bombeo.

ii. Un Lenguaje Regular $L = S \cup T$ puede ser Regular, siendo S y T no Regulares.

No, es cierta. Por ejemplo con $S = \{a^n b^n \mid n \geq 0\}$, $T = \{a^i b^j \mid i, j \geq 0 \wedge i \neq j\}$ y $L = S \cup T = \{a, b\}^*$.

iii. Si una Gramática Regular Derecha G_D no contiene regla alguna que termine en un símbolo terminal (o λ), entonces $\mathcal{L}(G_D) = \emptyset$.

No, es cierta. Pues entonces no habría forma en que la Gramatica se detuviera y lograra generar una sola palabra.

iv. Si una expresión regular e, contiene la expresión $\emptyset$ del lado derecho de una concatenación, entonces $sem(e) = \emptyset$.

Si, es falsa. Pues nunca se especifica donde debe aparecer tal concatenación a la derecha. En particular, considérese el siguiente ejemplo: $sem(a + b\emptyset) = \{a\}$.

v. *No Sabe / No Contesta.*

(i) Dada la Gramática Regular, $G = (\Sigma, V, P, S)$, siendo P de la siguiente forma:

$$\begin{array}{rcl} S & \rightarrow & aS \\ & | & baA \\ & | & b \\ A & \rightarrow & aA \\ & | & abS \end{array}$$

¿Cuál de las siguientes expresiones regulares representan a $\mathcal{L}(G)$?

i. $(a* baab)* (a*b + a* baa*)$
ii. $(a* baab)* ((ab)*b + a* baa*)$
iii. $(a* baa* ab)* (ab)*b$
iv. [[$(a* baa* ab)* a*b$]]
v. *No Sabe / No Contesta.*

Aplicando el algoritmo de transformación de Gramaticas Regulares a Expresiones Regulares visto en clase, la expresión resultante es $(a* baa* ab)* a*b$. *Además puede verse intuitivamente, de la siguiente forma: Se dan tantan vueltas como se quiera con la regla* $S \rightarrow aS$, *luego se puede ir a la* A, *con la regla* $S \rightarrow baA$. *Se puede quedar dando vueltas con la regla* $A \rightarrow aA$ *y regresarse luego a la* S *con la regla* $A \rightarrow abS$. *Como hicimos un ciclo, eso hay que envolverlo en una clausura de Kleene, pues podemos repetirlo tantas veces como queramos. Finalmente, podemos terminar con la regla* $S \rightarrow b$, *mas no sin antes haber dado tantas vueltas como se quiera con la regla* $S \rightarrow aS$.

(j) Considere una modificación sobre los Autómatas Finitos, donde la función de transición puede trabajar sobre frases y no únicamente símbolos o lambda. Es decir, la firma de la función de transición ahora modificada sería $\delta_m : Q \times \Sigma^* \rightarrow 2^Q$. ¿Qué condiciones son necesarias para que un Autómata Finito de este estilo sea determinista?

i. $(\forall q, x, y : q \in Q \wedge x, y \in \Sigma^* : \delta_m(q, x) = \delta_m(q, y) \Rightarrow x = y)$

No, esto mas bien diría que desde un estado dado, solo se puede mover a otro estado con una única frase.

ii. [[$(\forall q, s, t, x : q, s, t \in Q \wedge x \in \Sigma^* : (q, x) \left|\frac{*}{M}\right. (s, \lambda) \wedge (q, x) \left|\frac{*}{M}\right. (t, \lambda) \Rightarrow s = t)$]]

Si, cualquier cadena lleva al Autómata de un estado a otro, el camino siempre debe ser el mismo. (Nótese que se puede hablar del camino completo, al haber cuantificado sobre todas las frases y todos los estados.)

iii. $(\forall q, x : q \in Q \wedge x \in \Sigma^* : |\delta_m(q, x)| = 1)$

No. Esta es condición suficiente para Autómatas donde la transición es con símbolos y no frases. Por ejemplo, esta regla no prohibe que de un estado q_0 *vaya a un estado* q_1 *con "a" y luego desde* q_1 *hasta otro estado* q_2 *con "b". Mientras que desde* q_0 *también pudiese llegar directamente a* q_3 *con "ab", lo cual rompería su determinismo (*q_0 *podría llegar a* q_2 *y* q_3 *con la misma frase:"ab").*

iv. Es imposible que sea determinista.

No, si es posible que lo sea. Una condición necesaria es la de la opción ii.

v. *No Sabe / No Contesta.*

1. (4 puntos) Dado un número entero $n \geq 2$, dé un Autómata Finito Determinista que reconozca todas las palabras w, sobre el alfabeto $\Sigma = \{0, 1\}$, tal que interpretar w como un número binario resulte en que el mismo sea divisible por n. (Recuerde que un número no puede comenzar por 0, a menos que sea únicamente la frase "0").

   Pista: *Defina las transiciones a través de reglas generales para las mismas. No intente construir el diagrama de transiciones (representación gráfica.)*

   Solución:

   Lo que se quiere es que la cadena leída, al ser interpretada como un número en binario, sea múltipo de n. Esto es equivalente a pedir que al calcular su módulo con n, el resultado sea cero (0). Y en esto se basará el autómata. Cada estado representará cuanto es el valor de la cadena leída hasta el momento, módulo n.

   Nótese que, al tener parcialmente leída la cadena, si el siguiente símbolo es un 0 en la entrada, se multiplica el número visto por dos y si es un 1 se multiplica por dos y seguido se le suma uno, todo esto siempre módulo n. Usando esto construiremos un Autómata Finito Determinista, de forma general, para resolver el problema.

   $$M = (\{0, 1\}, Q, \delta, q_{ini}, F), \text{ donde}$$

   - $Q = \{q_k \mid 0 \leq k < n\} \cup \{q_{ini}, q_{cero}\}$
   - Se define δ como:
     - $(\forall k : 0 \leq k < n : \delta(q_k, 0) = q_{(2*k) \ mod \ n})$
     - $(\forall k : 0 \leq k < n : \delta(q_k, 1) = q_{(2*k+1) \ mod \ n})$
     - $\delta(q_{ini}, 0) = q_{cero}$
     - $\delta(q_{ini}, 1) = q_1$
   - $F = \{q_0, q_{cero}\}$

   Este autómata comienza por verificar si la frase empieza con 0 o con 1. Si es el primer caso, solo puede ser la frase 0. En caso contrario, se mueve al estado que representa números cuyo valor modulo n es 1. Los estados finales son haber visto solo el 0 (pues el cero es multiplo de todo n) y el estado q_0 que es donde el valor módulo n es cero y por ende es múltiplo de n.

2. (3 puntos) Considere la Expresión Regular, sobre el alfabeto $\Sigma = \{a, b, .\}$ (letras a, b y punto):

$$e = (a + b + .)* .(a + b)(a + b)*$$

La Expresión Regular e representa a todos los posibles nombres de archivo, formados por los símbolos en Σ, donde cada archivo tiene una extensión no vacía.

Deduzca, útilizando el método de construcción, usando Derivadas sobre Expresiones Regulares, un Autómata Finito Determinista M, tal que $\mathcal{L}(M) = sem(e)$.

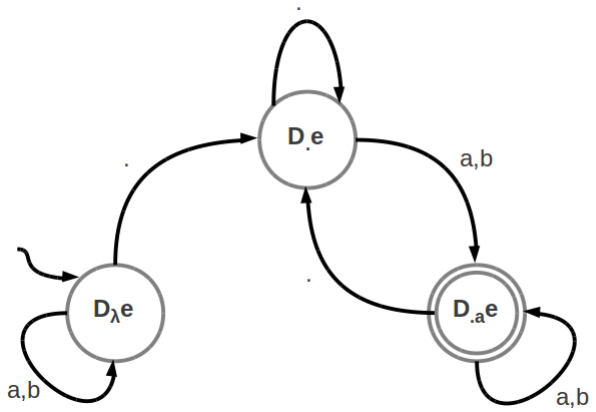
Nota: *<u>Debe</u> mostrar cada una de las Derivadas diferentes encontradas durante el proceso de forma explícita. No tiene que demostrar que el Autómata construido es correcto.*

Solución:

Calculemos las derivadas pertinentes:

$$\begin{aligned}
D_\lambda e &= e \\
D_a e &= (a + b + .)* .(a + b)(a + b)* \;=\; D_\lambda e \\
D_b e &= (a + b + .)* .(a + b)(a + b)* \;=\; D_\lambda e \\
D_. e &= (a + b + .)* .(a + b)(a + b)* + (a + b)(a + b)* \\
D_{.a} e &= (a + b + .)* .(a + b)(a + b)* + (a + b)* \\
D_{.b} e &= (a + b + .)* .(a + b)(a + b)* + (a + b)* \;=\; D_{.a} e \\
D_{..} e &= (a + b + .)* .(a + b)(a + b)* + (a + b)(a + b)* \;=\; D_. e \\
D_{.aa} e &= (a + b + .)* .(a + b)(a + b)* + (a + b)* \;=\; D_{.a} e \\
D_{.ab} e &= (a + b + .)* .(a + b)(a + b)* + (a + b)* \;=\; D_{.a} e \\
D_{.a.} e &= (a + b + .)* .(a + b)(a + b)* + (a + b)(a + b)* \;=\; D_. e
\end{aligned}$$

Con las derivadas calculadas podemos construir el siguiente Autómata (mostrado de forma gráfica a través de su diagrama de transiciones):



3. (3 puntos) Dados dos Lenguajes Regulares, A y B, se define el operador de entrecruce de la siguiente manera:

$$A \bowtie B = \{a_0 b_0 a_1 b_1 \dots a_k b_k \mid k \geq 0 \wedge a_0 a_1 \dots a_k \in A \wedge b_0 b_1 \dots b_k \in B\}$$

Proponga una Expresión Regular, Gramática Regular o Autómata Finito que represente/genere/reconozca al Lenguaje $A \bowtie B$.

Nota: *Solamente debe proponer la Expresión Regular, Gramática Regular o Autómata Finito que, no debe demostrar formalmente la igualdad del lenguaje que representa/genera/reconoce éste con el lenguaje propuesto.*

Solución:

Se construirá un Autómata Finito Determinista para reconocer este lenguaje. Dados $M_A = (\Sigma, Q_A, \delta_A, q_{0_A}, F_A)$ y $M_B = (\Sigma, Q_B, \delta_B, q_{0_B}, F_B)$, Autómatas Finito Deterministas que reconocen los lenguajes A y B, respectivamente.

El nuevo Autómata debe funcionar de la siguiente manera:

a) Debe moverse por una sola transición a través del primer Autómata.

b) Seguido de eso, debe moverse por una sola transición del segundo Autómata.

c) Repetir, hasta que se acabe la frase.

d) Si se está en un estado que es final para ambos, entonces acepta, si no rechaza. (Nótese que esta verificación siempre se hace *después* de moverse por una transición de M_B, lo que es igual a decir que la siguiente tendría que haber sido a través de M_A).

Considerando esto, se construirá un Autómata cuyos estados están conformados por una tupla de estados, representando lo que se ha recorrido en cada uno de los Autómatas M_A y M_B, además de un guardia que permita saber si el siguiente movimiento debe ser por el primer Autómata o por el segundo.

Se define entonces formalmente el autómata propuesto:

$$M_{A \bowtie B} = (\Sigma, Q_{A \bowtie B}, \delta_{B \bowtie B}, q_{0_{A \bowtie B}}, F_{A \bowtie B})$$

- $Q_{A \bowtie B} = Q_A \times Q_B \times \{moversePorA, moversePorB\}$
- Se define $\delta_{A \bowtie B}$ como:
  - $(\forall a : a \in \Sigma : \delta_{A \bowtie B}((q_a, q_b, 0), moversePorA) = (\delta_A(q_a, a), q_b, moversePorB))$
  - $(\forall a : a \in \Sigma : \delta_{A \bowtie B}((q_a, q_b, 1), moversePorB) = (q_a, \delta_B(q_b, a), moversePorA))$
- $q_{0_{A \bowtie B}} = (q_{0_A}, q_{0_B}, moversePorA)$
- $F_{A \bowtie B} = F_A \times F_B \times \{moversePorA\}$

Como el autómata es determinista, recorrer cada transición consume exactamente un símbolo. El lenguaje reconocido por $M_{A \bowtie B}$ es en efecto $A \bowtie B$.