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

Examen II

(20 puntos)
Solución

0. (12 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 dos (2) puntos**, 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) Sea $L = \{a^i b^j c^k d^h \mid ?\}$. Este Lenguaje no es Libre de Contexto cuando se pide que:

      i. $i = j \wedge k = h$

         No, este Lenguaje es la concatenación de $\{a^n b^n \mid n \geq 0\}$ *con* $\{c^n d^n \mid n \geq 0\}$.

      ii. $i = h \wedge j = k$

         No, este Lenguaje es equivalente a $\{a^n x b^n \mid n \geq 0 \wedge x \in \{c^n d^n \mid n \geq 0\}$.

      iii. [[$i = k \wedge j = h$]]

         Si, este Lenguaje no es Libre de Contexto, ya que los índices están intercalados.

      iv. Ninguna de las anteriores.

         No, la opción iii. es la correcta.

      v. *No Sabe / No Contesta.*

   (b) ¿Cuál de las siguientes afirmaciónes es *falsa*?

      i. Para todo Autómata de Pila, que termina con pila vacía, existe un Autómata de Pila, que termina con estado final, que reconoce el mismo lenguaje que el primero.

         No, las tres formas de aceptación para Autómatas de Pila son equivalentes.

      ii. [[**Para todo Autómata de Pila no–determinista, existe un Autómata de Pila determinista que reconoce el mismo lenguaje que el primero.**]]

         Si, existen Lenguajes Libres de Contexto, para los cuales no es posible un Autómata de Pila determinista.

      iii. Para toda Gramática Libre de Contexto, existe un Autómata de Pila que reconoce el mismo lenguaje que genera dicha Gramática.

         No, el algoritmo de transformación dado en clase es general (para cualquier Gramática).

      iv. Para toda Gramática Libre de Contexto, existe una Gramática Libre de Contexto, en Forma Normal de Chomsky, que genera el mismo lenguaje que la primera.

         No, el algoritmo de transformación dado en clase es general (para cualquier Gramática).

      v. *No Sabe / No Contesta.*

(c) ¿Cuál de las siguientes operaciones sobre Lenguajes, es cerrada sobre los Lenguajes Libres de Contexto?

i. [[**La concatenación.**]]

Si, sean $G_A = (\Sigma_A, V_A, P_A, S_A)$ *y* $G_B = (\Sigma_B, V_B, P_B, S_B)$*, tal que* $V_A \cap V_B = \emptyset$*, se define* $G_{AB} = (\Sigma_A \cup \Sigma_B, V_A \cup V_B \cup \{S\}, P_A \cup P_B \cup \{S \to S_A S_B\}, S)$*, con* $S \notin (V_A \cup V_B)$

ii. La diferencia.

No, los Lenguajes Libre de Contexto no son cerrados bajo la diferencia.

iii. El complemento.

No, los Lenguajes Libre de Contexto no son cerrados bajo el complemento.

iv. Ninguna de las anteriores.

No, la opción i. es la correcta.

v. *No Sabe / No Contesta.*

(d) ¿La existencia de una transición que no consuma entrada, en un Autómata de Pila, garantiza que este último sea no determinista? ¿Por qué?

i. Si. Por que la existencia de una λ-transición en un Autómata hace que sea no–determinista.

No, esto se cumple para los Autómatas Finitos pero no para los Autómatas de Pila.

ii. [[**No. Por que se pueden tomar decisiones usando solamente el tope de la pila.**]]

Si, aunque no se consuma entrada, se puede usar el tope de la pila para tomar una decisión.

iii. Si. Por que no consumir entrada siempre garantiza la existencia de más de una transición posible a tomar.

No, la posibilidad de tomar más de una transición en este caso se da si dos transiciones comparten el mismo tope de pila y una de las dos transiciones no consume entrada. Si solamente existe una transición que depende de un símbolo particular de pila, se preserva el determinismo.

iv. No. Por que todo Autómata de Pila es determinista.

No, todo Autómata de Pila es no-determinista (siendo los deterministas un caso especial y no equivalente).

v. *No Sabe / No Contesta.*

(e) Sea G una Gramática Libre de Contexto en Forma Normal de Chomsky. ¿En cuantos pasos de derivación genera G una palabra w, de longitud n, tal que $w \in \mathcal{L}(G)$?

i. n
ii. $2 * n + 1$
iii. $n^2 - 1$
iv. [[$2 * n - 1$]]
v. *No Sabe / No Contesta.*

Como fue visto en clase, la cantidad de derivaciones es n *para transformar cada terminal en un no–terminal y* $n-1$ *para luego reducir la frase hasta un solo símbolo no terminal. Esto da un total de* $2*n-1$ *derivaciones.*

(f) Considere la siguiente Gramática de Atributos, incompleta, para el conjunto de producciones de una Gramática Libre de Contexto sobre algún Σ (con un atributo *delta* que tiene la función de transicion asociada a una Autómata de Pila que reconoce el mismo lenguaje que genera la Gramática. Dicha función es vista como un conjunto.):

S	$\rightarrow$	P	$\left\{ \begin{array}{l} S_0.delta = P_1.delta \cup \{((qf, a, a), (qf, \lambda)) \mid a \in \Sigma\} \\ P_1.inicial = true \end{array} \right\}$
P	$\rightarrow$	$n{\rightarrow}L{,}P$	$\left\{ \begin{array}{l} P_0.delta = ?_0 \cup P_1.delta \\ P_1.inicial = false \\ L_1.simb = n.valor \end{array} \right\}$
	$\mid$	$n{\rightarrow}L$	$\left\{ \begin{array}{l} P_0.delta = ?_0 \\ L_1.simb = n.valor \end{array} \right\}$
L	$\rightarrow$	$C\mid L$	$\left\{ \begin{array}{l} L_0.delta = ?_1 \cup L_1.delta \\ L_1.simb = L_0.simb \end{array} \right\}$
	$\mid$	C	$\{ \; L_0.delta = ?_1 \; \}$
C	$\rightarrow$	tC	$\{ \; C_0.cadena = concat(C_1.cadena, t.valor) \; \}$
		nC	$\{ \; C_0.cadena = concat(C_1.cadena, n.valor) \; \}$
	$\mid$	t	$\{ \; C_0.cadena = concat(\text{“”}, t.valor) \; \}$
	$\mid$	n	$\{ \; C_0.cadena = concat(\text{“”}, n.valor) \; \}$
	$\mid$	λ	$\{ \; C_0.cadena = \text{“”} \; \}$

¿Que valor deberían tomar $?_0$ y $?_1$, tal que la Gramática de Atributos correctamente calcule el atributo *delta*, suponiendo que el símbolo inicial de la Gramática es aquel que aparezca primero en el conjunto de producciones?

i. $?_0 = L_1.delta \cup \left\{ \begin{array}{ll} \{((q_0, \lambda, \lambda), (q_f, n.valor))\} & \text{si } P_0.inicial \\ \emptyset & \text{si } \neg P_0.inicial \end{array} \right.$
$?_1 = C_1.delta$

No, $C_1.delta$ ni siquiera está definido.

ii. $?_0 = L_1.delta$
$?_1 = \{((q_f, \lambda, L_0.simb), (q_f, C_1.cadena))\}$

No, falta la regla que empila el símbolo inicial de la gramática en primera instancia.

iii. $\left[\left[\begin{array}{l} ?_0 = L_1.delta \cup \left\{ \begin{array}{ll} \{((q_0, \lambda, \lambda), (q_f, n.valor))\} & \text{si } P_0.inicial \\ \emptyset & \text{si } \neg P_0.inicial \end{array} \right. \\ ?_1 = \{((q_f, \lambda, L_0.simb), (q_f, C_1.cadena))\} \end{array} \right] \right]$

Si, de esta forma la definición está completa.

iv. Ningún valor para $?_0$ y $?_1$ resultaría en una Gramática de Atributos correcta.

No, la opción iii. es la correcta.

v. *No Sabe / No Contesta.*

1. (3 puntos) Sean $S = \{a^n b^n \mid n \geq 1\}$ y $T = \{b^n a^n \mid n \geq 1\}$, considere el siguiente Lenguaje:

$$\{xy \mid x \in S^k \wedge y \in T^k \wedge k \geq 0\}.$$

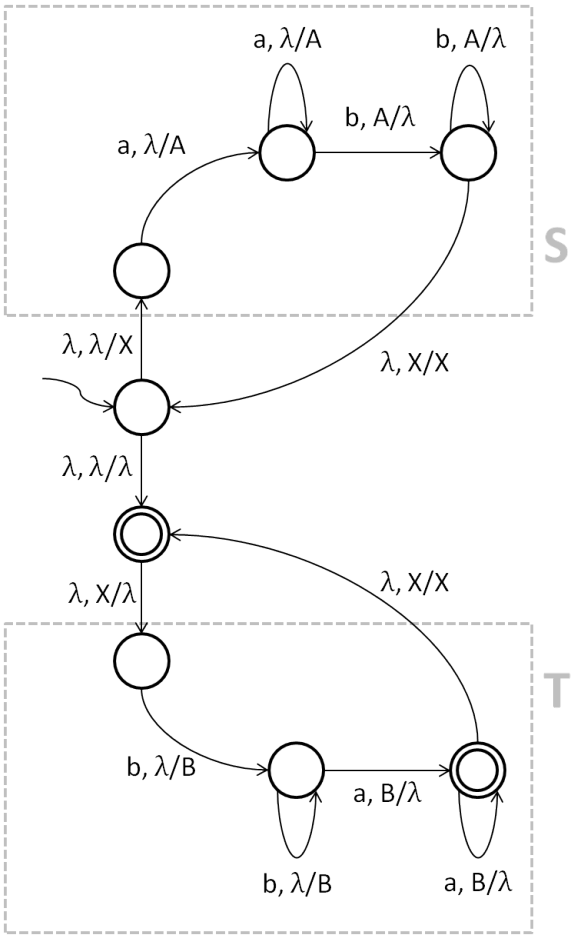
Proponga el diagrama de transiciones para un Autómata de Pila que reconozca dicho Lenguaje. Recuerde especificar la forma de aceptación de su Autómata.

Nota: Recuerde que la exponenciación de un Lenguaje L^k, es la concatenación repetida k veces de L consigo mismo.

Pista: Proponga primero los Autómatas de Pila para los lenguajes S y T, luego proponga el Autómata pedido.

Solución:

A continuación el diagrama de transiciones. El autómata propuesto acepta por estado final y pila vacía. Los nombres de los estados son omitidos, pues los mismos no ofrecen información de importancia en este caso.



2. (5 puntos) Considérese las Expresiones Regulares sobre $\Sigma = \{a, b\}$, conformadas por los literales $\emptyset$ y λ, símbolos a y b, y operadores de unión, concatenación y clausura de kleene, con las precedencias y asociatividades tradicionales, así como la posibilidad de parentización.

   a) (2 pts) Plantee una Gramática Libre de Contexto no–ambigua que reconozca dichas Expresiones Regulares.

   Solución:

$$\begin{array}{rcl} E & \rightarrow & E + T \\ & | & T \\ T & \rightarrow & TF \\ & | & F \\ F & \rightarrow & F* \\ & | & (E) \\ & | & \emptyset \\ & | & \lambda \\ & | & a \\ & | & b \end{array}$$

   b) (3 pts) Aumente la Gramática propuesta en la pregunta anterior con los atributos que crea conveniente, tal que el símbolo inicial de la Gramática tenga dos atributos D_a y D_b, que almacenen la derivada de la Expresión Regular reconocida sobre a y b respectivamente.

   Cuenta únicamente con los operadores regulares de unión, concatenación y clausura de Kleene para construir las Expresiones Regulares que serán sus derivadas, así como los literales $\emptyset$, λ y símbolos a y b.

   Como recordatorio, se define la derivada de una Expresión regular, sobre un símbolo $a \in \Sigma$, como:

$$\begin{aligned} D_a\emptyset &= \emptyset \\ D_a\lambda &= \lambda \\ D_a b &= \begin{cases} \lambda & \text{si } a = b \\ \emptyset & \text{si } a \neq b \end{cases} \\ D_a(e + f) &= D_a e + D_a f \\ D_a(ef) &= (D_a e)f + \Lambda(e)D_a f \\ D_a(e*) &= D_a(e)e* \end{aligned}$$

   Recuerde que $\Lambda(e)$ es una función que evalúa a λ si $\lambda \in sem(e)$ y a $\emptyset$ de lo contrario.

   Pista: No puede utilizar Λ en la construcción de las derivadas. Posiblemente deba incorporar un nuevo atributo para tratar con este caso.

Solución:

Además de D_a y D_b, se incluyen atributos: L, para almacenar el valor de Λ de cada expresión y *expr*, para almacenar la Expresión Regular vista, sin modificaciones.

$$
\begin{array}{lcll}
E & \rightarrow & E+T & \left\{ \begin{array}{r} E_0.expr = E_1.expr + T.expr \\ E_0.D_a = E_1.D_a + T.D_a \\ E_0.D_b = E_1.D_b + T.D_b \\ E_0.L = E_1.L + T.L \end{array} \right\} \\
 & | & T & \left\{ \begin{array}{r} E_0.expr = T_1.expr \\ E_0.D_a = T_1.D_a \\ E_0.D_b = T_1.D_b \\ E_0.L = T_!.L \end{array} \right\} \\
T & \rightarrow & TF & \left\{ \begin{array}{r} T_0.expr = T_1.expr\ F.expr \\ T_0.D_a = (T_1.D_a)\ (F.expr) + (T_1.L)\ (F.D_a) \\ T_0.D_b = (T_1.D_b)\ (F.expr) + (T_1.L)\ (F.D_b) \\ T_0.L = T_1.L\ F.L \end{array} \right\} \\
 & | & F & \left\{ \begin{array}{r} T_0.expr = F_1.expr \\ T_0.D_a = F_1.D_a \\ T_0.D_b = F_1.D_b \\ T_0.L = F_1.L \end{array} \right\} \\
F & \rightarrow & F* & \left\{ \begin{array}{r} F_0.expr = F_1.expr* \\ F_0.D_a = (F_1.D_a)\ (F_1.expr*) \\ F_0.D_b = (F_1.D_b)\ (F_1.expr*) \\ F_0.L = \lambda \end{array} \right\} \\
 & | & (E) & \left\{ \begin{array}{r} F_0.expr = E_1.expr \\ F_0.D_a = E_1.D_a \\ F_0.D_b = E_1.D_b \\ F_0.L = E_1.L \end{array} \right\} \\
 & | & \emptyset & \left\{ \begin{array}{r} E.expr = \emptyset \\ E.D_a = \emptyset \\ E.D_b = \emptyset \\ E.L = \emptyset \end{array} \right\} \\
 & | & \lambda & \left\{ \begin{array}{r} E.expr = \lambda \\ E.D_a = \emptyset \\ E.D_b = \emptyset \\ E.L = \lambda \end{array} \right\} \\
 & | & a & \left\{ \begin{array}{r} E.expr = a \\ E.D_a = \lambda \\ E.D_b = \emptyset \\ E.L = \emptyset \end{array} \right\} \\
 & | & b & \left\{ \begin{array}{r} E.expr = b \\ E.D_a = \emptyset \\ E.D_b = \lambda \\ E.L = \emptyset \end{array} \right\}
\end{array}
$$

Nota: Los paréntesis colocados en el cálculo de atributos son innecesarios, dada la precedencia de las Expresiones Regulares. Su objetivo es puramente de legibilidad en este caso.