Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI3725 - Traductores e Interpretadores
Septiembre-Diciembre 2007

# Examen Parcial II

(25 puntos)

Carnet:

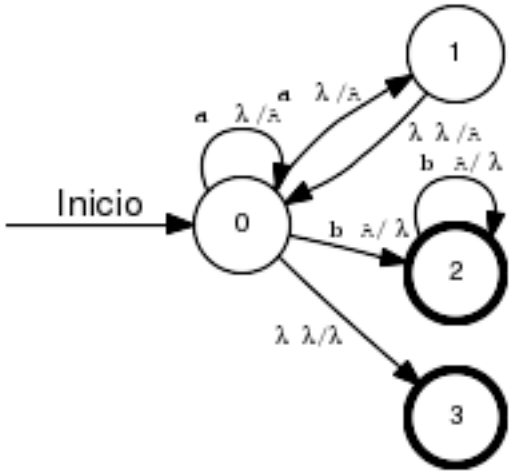
Nombre:

1. Para cada uno de los siguientes lenguajes demuestre si son o no libres de contexto. Para demostrar que un lenguaje es libre de contexto puede construir una gramática libre de contexto que lo genere o construir un PDA que lo acepte. Para demostrar que un lenguaje no es libre de contexto, utilice el Lema de Bombeo para Lenguajes Libres de Contexto.

   *a*) **(3 puntos)** $L = \{a^i b^j | 0 \leq i \leq j \leq 2i\}$
   El lenguaje $L$ puede ser generado con la gramática

$$\begin{aligned} S &\rightarrow \lambda \\ S &\rightarrow aSB \\ B &\rightarrow bb \\ B &\rightarrow b \end{aligned}$$

   notando primero que $\lambda \in L$ y luego que por cada $a$ hay que generar una o dos $b$ correspondientes. El no-terminal $S$ genera las $a$ y deja al no-terminal $B$ la responsabilidad de generar la cantidad de $B$ necesarias.
   El lenguaje $L$ también puede ser generado con el PDA de la figura.




   En el PDA se utiliza el estado $q_3$ para reconocer la palabra vacía. Para palabras no vacías, el estado $q_0$ empila no determinísitcamente una o dos $A$ en la pila. Al encontrar la primera $b$ en la entrada, se pasa al estado $q_2$ en el cual se consumen suficientes $A$ de la pila para corresponder con la cantidad de $b$ leídas.

*b*) **(2 puntos)** $L = \{a^i b^{2i} c^{4i} | i \geq 0\}$

Asumamos que $L$ es libre de contexto y sea $k$ el especificado en el Lema de Bombeo para Lenguajes Libres de Contexto. Sea $z = a^k b^{2k} c^{4k}$, como $z \in L$ y $|z| > k$ entonces de acuerdo con el Lema de Bombeo para Lenguajes Libres de Contexto, $z$ tiene al menos una forma de escribirse $uvwxy$ que satisface $|vwx| \leq k$, $|v| + |x| > 0$ y luego $\forall i \geq 0$ se cumple $uv^i wx^i y \in L$.

Como $|vwx| \leq k$, entonces la subcadena $vwx$ sólo puede tener la forma $a^i$, $b^i$, $c^i$, $a^i b^j$ o $b^i c^j$. Si se bombea $uv^2 wx^2 y$ para las primeras tres formas aumentará la cantidad de solamente un símbolo, mientras que para las dos últimas formas aumentará la cantidad de dos símbolos, sin embargo nunca aumentarán proporcionalmente todos los símbolos de la palabra por lo que la palabra resultante no pertenece a $L$. Esto contradice el Lema de Bombeo para Lenguajes Libres de Contexto y en consecuencia $L$ no puede ser Lenguaje Libre de Contexto.

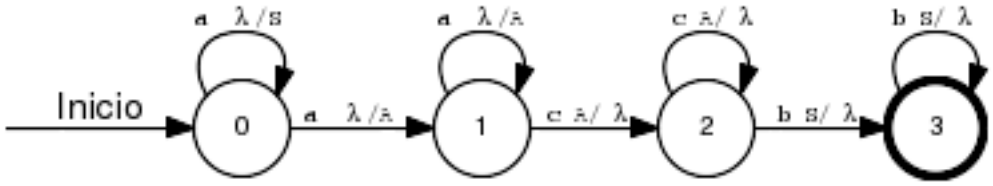
*c*) **(3 puntos)** $L = \{a^{i+j} c^i b^j | j > 0, i \geq 0\}$

El lenguaje $L$ puede ser generado por la gramática

$$\begin{aligned} S &\rightarrow aSb \\ S &\rightarrow aAb \\ A &\rightarrow \lambda \\ A &\rightarrow aAc \end{aligned}$$

notando primero que como $j > 0$ entonces $\lambda \notin L$ y luego que $a^{i+j} c^i b^j = a^{j+i} c^i b^j = a^j a^i c^i b^j$ por lo tanto cuando $j > 2$ utilizamos la primera producción $S$ para generar $a^{j-1} S b^{j-1}$ y luego utilizamos la segunda producción de $S$ para generar $a^j A b^j$. Finalmente, el símbolo no terminal $A$ genera la palabra vacía, o bien $a^i c^i$.

El lenguaje $L$ también puede ser generado con el PDA de la figura.




El estado $q_0$ empila no determinísticamente suficientes $S$ para contar la la cantidad de $a$ en el prefijo de la palabra que correspondan a las $b$ del sufijo, y luego pasa al estado $q_1$ a empilar tantas $A$ como símbolos $a$ queden en el prefijo de la palabra. Al encontrar la primera $c$ en la palabra, se pasa al estado $q_2$ en el cual se consumen suficientes $A$ de la pila para corresponder con la cantidad de $c$ leídas. Finalmente, al encontrar la primera $b$ en la palabra, se pasa al estado $q_3$ en el cual se consumen las $S$ correspondientes.

*d*) **(5 puntos)** $L = \{a^{2i}b^{3i}|i > 0\}^*\{a^ib^jc^k|i = j \vee j = k, \text{con } i, j, k \geq 0\}$

Observamos que $L = L_1^*(L_2 \cup L_3)$ donde $L_1 = \{a^{2i}b^{3i}|i > 0\}$, $L_2 = \{a^ib^jc^k|i = j \text{ con } i, j, k \geq 0\}$ y $L_3 = \{a^ib^jc^k|j = k \text{ con } i, j, k \geq 0\}$, de manera que aprovecharemos las propiedades de clausura de los Lenguajes Libres de Contexto para construir una gramática que reconozca a $L$; esto es, construiremos gramáticas para cada uno de los lenguajes componentes, teniendo cuidado de utilizar símbolos no terminales *diferentes* de manera que luego puedan construirse las gramáticas para la unión, la estrella de Kleene y la concatenación de Lenguajes Libres de Contexto.

Para reconocer $L_1$proponemos $G_1$

$$\begin{aligned} S_1 &\rightarrow aaS_1bbb \\ S_1 &\rightarrow aabbb \end{aligned}$$

Para reconocer $L_2$ proponemos $G_2$

$$\begin{aligned} S_2 &\rightarrow AC \\ A &\rightarrow aAb \\ A &\rightarrow \lambda \\ C &\rightarrow Cc \\ C &\rightarrow \lambda \end{aligned}$$

Para reconocer $L_3$ proponemos $G_3$

$$\begin{aligned} S_3 &\rightarrow BD \\ B &\rightarrow Ba \\ B &\rightarrow \lambda \\ D &\rightarrow bDc \\ D &\rightarrow \lambda \end{aligned}$$

Para reconocer $L_4 = L_2 \cup L_3$ proponemos $G_4$

$$\begin{aligned} S_4 &\rightarrow S_2 \\ S_4 &\rightarrow S_3 \end{aligned}$$

Para reconocer $L_5 = L_1^*$ proponemos $G_5$

$$\begin{aligned} S_5 &\rightarrow S_1S_5 \\ S_5 &\rightarrow \lambda \end{aligned}$$

Finalmente, para reconocer $L = L_5L_4$ proponemos

$$S \rightarrow S_5S_4$$

2. Sea la gramática $G = (\{A\}, \{\mathbf{a}, \mathbf{c}, \mathbf{i}, \mathbf{m}, \mathbf{n}, \mathbf{p}\}, P, A)$ cuyo conjunto de producciones $P$ se define como

$$\begin{aligned} A &\rightarrow AA\mathbf{m} \\ A &\rightarrow AA\mathbf{p} \\ A &\rightarrow \mathbf{n} \\ A &\rightarrow \mathbf{i} \\ A &\rightarrow \mathbf{ia}A\mathbf{c} \end{aligned}$$

*a*) **(8 puntos)** Modifique la gramática hasta que sea fuertemente $LL(1)$ y demuestre formalmente tal propiedad en la gramática resultante.

1) La gramática no es $LL(1)$ porque tiene recursión izquierda. Después de eliminar la recursión izquierda, la gramática queda

$$\begin{aligned} A &\rightarrow \mathbf{n}B \\ A &\rightarrow \mathbf{i}B \\ A &\rightarrow \mathbf{ia}A\mathbf{c}B \\ B &\rightarrow A\mathbf{m}B \\ B &\rightarrow A\mathbf{p}B \\ B &\rightarrow \lambda \end{aligned}$$

2) La gramática aún no es $LL(1)$ porque tiene producciones con prefijos izquierdos comunes. Después de aplicar factorización por izquierda y extender la gramática con un nuevo símbolo inicial $S$, la gramática queda

$$\begin{aligned} S &\rightarrow A\$ \\ A &\rightarrow \mathbf{n}B \\ A &\rightarrow \mathbf{i}C \\ C &\rightarrow B \\ C &\rightarrow \mathbf{a}A\mathbf{c}B \\ B &\rightarrow \lambda \\ B &\rightarrow AD \\ D &\rightarrow \mathbf{m}B \\ D &\rightarrow \mathbf{p}B \end{aligned}$$

3) Calculamos el $FIRST$ y el $FOLLOW$ para cada símbolo no terminal

$$\begin{aligned} FIRST(S) &= \{\mathbf{i}, \mathbf{n}\} \\ FIRST(A) &= \{\mathbf{i}, \mathbf{n}\} \\ FIRST(B) &= \{\lambda, \mathbf{i}, \mathbf{n}\} \\ FIRST(C) &= \{\lambda, \mathbf{a}, \mathbf{i}, \mathbf{n}\} \\ FIRST(D) &= \{\mathbf{m}, \mathbf{p}\} \\ FOLLOW(S) &= \{\$\} \\ FOLLOW(A) &= \{\$, \mathbf{c}, \mathbf{m}, \mathbf{p}\} \\ FOLLOW(B) &= \{\$, \mathbf{c}, \mathbf{m}, \mathbf{p}\} \\ FOLLOW(C) &= \{\$, \mathbf{c}, \mathbf{m}, \mathbf{p}\} \\ FOLLOW(D) &= \{\$, \mathbf{c}, \mathbf{m}, \mathbf{p}\} \end{aligned}$$

4) Calculamos los Conjuntos de Lookahead $LA(A \to \alpha) = FIRST(FIRST(\alpha) \cdot FOLLOW(A))$ para las producciones:

$$\begin{aligned}
LA(S \to A\$) = FIRST(\{\mathbf{i}, \mathbf{n}\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{i}, \mathbf{n}\} \\
LA(A \to \mathbf{n}B) = FIRST(\{\mathbf{n}\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{n}\} \\
LA(A \to \mathbf{i}C) = FIRST(\{\mathbf{i}\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{i}\} \\
LA(C \to B) = FIRST(\{\lambda, \mathbf{i}, \mathbf{n}\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{c}, \mathbf{i}, \mathbf{m}, \mathbf{n}, \mathbf{p}, \$\} \\
FIRST(C \to \mathbf{a}A\mathbf{c}B) = FIRST(\{\mathbf{a}\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{a}\} \\
LA(B \to \lambda) = FIRST(\{\lambda\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\} \\
LA(B \to AD) = FIRST(\{\mathbf{i}, \mathbf{n}\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{i}, \mathbf{n}\} \\
LA(D \to \mathbf{m}B) = FIRST(\{\mathbf{m}\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{m}\} \\
LA(D \to \mathbf{p}B) = FIRST(\{\mathbf{p}\} \cdot \{\mathbf{c}, \mathbf{m}, \mathbf{p}, \$\}) &= \{\mathbf{p}\}
\end{aligned}$$

Puede observarse que $\forall A \to \alpha | \beta \in P, \alpha \neq \beta$ se cumple $LA(A \to \alpha) \cap LA(A \to \beta) = \emptyset$, por lo tanto la gramática es fuertemente $LL(1)$.

*b*) **(2 puntos)** Construya la tabla de análisis para un reconocedor predictivo no recursivo utilizando el algoritmo presentado en clase.

| | **a** | **c** | **i** | **m** | **n** | **p** | $ |
|---|---|---|---|---|---|---|---|
| $S$ | | | $S \to A\$$ | | $S \to A\$$ | | |
| $A$ | | | $A \to \mathbf{i}C$ | | $A \to \mathbf{n}B$ | | |
| $B$ | | $B \to \lambda$ | $B \to AD$ | $B \to \lambda$ | $B \to AD$ | $B \to \lambda$ | $B \to \lambda$ |
| $C$ | $C \to \mathbf{a}A\mathbf{c}B$ | $C \to B$ | $C \to B$ | $C \to B$ | $C \to B$ | $C \to B$ | $C \to B$ |
| $D$ | | | | $D \to \mathbf{m}B$ | | $D \to \mathbf{p}B$ | |

*c*) **(2 puntos)** Utilice el reconocedor para encontrar la derivación más izquierda de la palabra **niainmcipm**.

| Pila | Entrada | Acción |
|---|---|---|
| $S\$$ | **niainmcipm**$ | $S \to A\$$ |
| $A\$$ | **niainmcipm**$ | $A \to \mathbf{n}B$ |
| $\mathbf{n}B\$$ | **niainmcipm**$ | Consumir **n** |
| $B\$$ | **iainmcipm**$ | $B \to AD$ |
| $AD\$$ | **iainmcipm**$ | $A \to \mathbf{i}C$ |
| $\mathbf{i}CD\$$ | **iainmcipm**$ | Consumir **i** |
| $CD\$$ | **ainmcipm**$ | $C \to \mathbf{a}A\mathbf{c}B$ |
| $\mathbf{a}A\mathbf{c}BD\$$ | **ainmcipm**$ | Consumir **a** |
| $A\mathbf{c}BD\$$ | **inmcipm**$ | $A \to \mathbf{i}C$ |
| $\mathbf{i}C\mathbf{c}BD\$$ | **inmcipm**$ | Consumir **i** |
| $C\mathbf{c}BD\$$ | **nmcipm**$ | $C \to B$ |
| $B\mathbf{c}BD\$$ | **nmcipm**$ | $B \to AD$ |
| $AD\mathbf{c}BD\$$ | **nmcipm**$ | $A \to \mathbf{n}B$ |
| $\mathbf{n}BD\mathbf{c}BD\$$ | **nmcipm**$ | Consumir **n** |
| $BD\mathbf{c}BD\$$ | **mcipm**$ | $B \to \lambda$ |
| $D\mathbf{c}BD\$$ | **mcipm**$ | $D \to \mathbf{m}B$ |
| $\mathbf{m}B\mathbf{c}BD\$$ | **mcipm**$ | Consumir **m** |
| $B\mathbf{c}BD\$$ | **cipm**$ | $B \to \lambda$ |
| $\mathbf{c}BD\$$ | **cipm**$ | Consumir **c** |
| $BD\$$ | **ipm**$ | $B \to AD$ |
| $ADD\$$ | **ipm**$ | $A \to \mathbf{i}C$ |
| $\mathbf{i}CDD\$$ | **ipm**$ | Consumir **i** |
| $CDD\$$ | **pm**$ | $C \to B$ |
| $BDD\$$ | **pm**$ | $B \to \lambda$ |
| $DD\$$ | **pm**$ | $D \to \mathbf{p}B$ |
| $\mathbf{p}BD\$$ | **pm**$ | Consumir **p** |
| $BD\$$ | **m**$ | $B \to \lambda$ |
| $D\$$ | **m**$ | $D \to \mathbf{m}B$ |
| $\mathbf{m}B\$$ | **m**$ | Consumir **m** |
| $B\$$ | $ | $B \to \lambda$ |
| $ | $ | **Acepta** |

Por lo tanto, la derivación más izquierda para la palabra será

$$\begin{aligned}
S &\Rightarrow \underline{A} \\
&\Rightarrow \mathbf{n}\underline{B} \\
&\Rightarrow \mathbf{n}\underline{A}D \\
&\Rightarrow \mathbf{ni}\underline{C}D \\
&\Rightarrow \mathbf{nia}\underline{A}\mathbf{c}BD \\
&\Rightarrow \mathbf{niai}\underline{C}\mathbf{c}BD \\
&\Rightarrow \mathbf{niai}\underline{B}\mathbf{c}BD \\
&\Rightarrow \mathbf{niai}\underline{A}D\mathbf{c}BD \\
&\Rightarrow \mathbf{niain}\underline{B}D\mathbf{c}BD \\
&\Rightarrow \mathbf{niain}\underline{D}\mathbf{c}BD \\
&\Rightarrow \mathbf{niainm}\underline{B}\mathbf{c}BD \\
&\Rightarrow \mathbf{niainmc}\underline{B}D \\
&\Rightarrow \mathbf{niainmc}\underline{A}DD \\
&\Rightarrow \mathbf{niainmci}\underline{C}DD \\
&\Rightarrow \mathbf{niainmci}\underline{B}DD \\
&\Rightarrow \mathbf{niainmci}\underline{D}D \\
&\Rightarrow \mathbf{niainmcip}\underline{B}D \\
&\Rightarrow \mathbf{niainmcip}\underline{D} \\
&\Rightarrow \mathbf{niainmcipm}\underline{B} \\
&\Rightarrow \mathbf{niainmcipm}
\end{aligned}$$

3. **(3 puntos extra)** Demuestre que el conjunto de los Lenguajes Libres de Contexto es cerrado sobre la operación de reverso, i.e. si $L$ es Libre de Contexto entonces $L^R = \{w^R | w \in L\}$ también es Libre de Contexto. **Nota:** si la sumatoria de los puntos obtenidos en las preguntas 1 y 2 **no** son suficientes para aprobar el examen, esta pregunta **no** será tomada en cuenta.

   Sea $L$ un Lenguaje Libre de Contexto, por tanto existe $G = (V, \Sigma, S, P)$ una Gramática Libre de Contexto tal que $L = L(G)$. Entonces se propone la gramática $G' = (V, \Sigma, S, P')$ tal que $L(G') = L(G)^R$, pues las producciones de $G'$ se obtienen *invirtiendo* los lados derechos de las producciones de $G$, esto es $A \to w^R \in P'$ si y sólo si $A \to w \in P$.

   Es necesario demostrar que $u$ es una forma sentencial de $G$ si y sólo si $u^R$ es una forma sentencial de $G'$. Se consideran formas sentenciales izquierdas en $G$ y formas sentenciales derechas en $G'$, y usaremos inducción sobre la longitud de las derivaciones para la demostración.

   **Caso Base** Consiste de las formas sentenciales producidas por derivaciones de **un** paso. Esas formas sentenciales son producidas exclusivamente por las derivaciones $S \Rightarrow u$ de $G$ y $S \Rightarrow u^R$ de $G'$. Como $S \to u \in P$, entonces $S \to u^R \in P'$ por lo tanto es cierto.

   **Hipótesis Inductiva** Asumimos que $u$ es una forma sentencial de $G$ derivable en $n$ o menos pasos si y sólo si $u^R$ es una forma sentencial de $G'$ derivable en $n$ o menos pasos.

   **Inducción** Sea $u = xwy$ una forma sentencial de $G$ derivada en $n+1$ pasos. Entonces la derivación tiene la forma

   $$S \overset{n}{\Rightarrow} xAy \Rightarrow xwy$$

   donde $A$ es el símbolo no terminal más a la izquierda en $xAy$ y $A \to w \in P$. Si $A \to w \in P$ entonces por construcción de $G'$, $A \to w^R \in P'$. Por otro lado, según la Hipótesis Inductiva, $y^R A w^R$ es una forma sentencial en $G'$ derivable en $n$ pasos. Entonces en $G'$ se puede derivar

   $$S \overset{n}{\Rightarrow} y^R A x^R \Rightarrow y^R w^R x^R$$

y por la definición de reverso $y^R w^R x^R = u^R$.

De manera similar se demuestra que $u$ es una forma sentencial de $G'$ si y sólo si $u^R$ es una forma sentencial de $G$. Con *ambas* demostraciones se concluye que $L(G') = L(G)^R$.