# Teoría de Autómatas y Lenguajes Formales

## Prueba de Evaluación de Lenguajes y Gramáticas

**Autores:**
**Araceli Sanchis de Miguel**
**Agapito Ledezma Espino**
**Jose A. Iglesias Martínez**
**Beatriz García Jiménez**
**Juan Manuel Alonso Weber**

| | |
|---|---|
| | UNIVERSIDAD CARLOS III DE MADRID<br>TEORÍA DE AUTÓMATAS Y LENGUAJES FORMALES.<br>GRADO EN INGENIERÍA INFORMÁTICA.<br>EVALUAC. CONTINUA |

Tiempo de examen: 45 minutos

1. Indica si las siguientes afirmaciones son Verdaderas o Falsas marcando con una X la casilla adecuada.

   Calificación: Respuesta correcta: +0,3. Respuesta incorrecta: -0.3. Sin respuesta: 0.

   Calificación máxima: 3 ptos. Calificación mínima: 0 ptos.

| | Verdadero | Falso |
|---|---|---|
| 1. La regla de producción Ca::=aaC, donde a Є $\Sigma_T$ y C Є $\Sigma_N$, pertenece a una gramática de tipo 2 en la jerarquía de Chomsky. | | X |
| 2. S::= λ es una regla en FNC | X | |
| 3. Si el axioma se sustituye por un símbolo no generativo, el lenguaje generado por la gramática es el lenguaje vacío. | X | |
| 4. Los alfabetos de terminales y de no terminales de una gramática son disjuntos. | X | |
| 5. En una forma sentencial sólo pueden aparecer símbolos no terminales. | | X |
| 6. S→A→B es una derivación de longitud 3. | | X |
| 7. Si una sentencia puede obtenerse en una G por medio de 2 o más árboles de derivación diferentes, la sentencia es ambigua. | X | |
| 8. Es posible que una Gramática tenga reglas superfluas que contribuyan a la formación de palabras. | | X |
| 9. Si no es posible encontrar ninguna gramática no ambigua que genere un determinado lenguaje, entonces decimos que éste es inherentemente ambiguo. | X | |
| | Verdadero | Falso |

| | | |
|---|---|---|
| 10. Dada la gramática: $G=(\Sigma_T , \Sigma_N , S, P)$, tal que $\Sigma_T = \{1,0\}$ $\Sigma_N = \{S,B\}$, P={ S::=1B, S::=1, B::=0S}, se puede transformar en la gramática equivalente G1 $=(\Sigma_T , \Sigma_N , S, P1)$, tal que $\Sigma_T = \{1,0\}$ $\Sigma_N = \{S,B,C\}$, P1={S::=1B, S::=1, B::=0C, C::=1B, C::=1} | X | |

2. Indica si las siguientes afirmaciones son Verdaderas o Falsas marcando con una X la casilla adecuada. Calificación: Respuesta correcta: +0,3. Respuesta incorrecta: -0.3.

   Sin respuesta: 0. Calificación máxima: 3 ptos. Calificación mínima: 0 ptos.

| | Verdadero | Falso |
|---|---|---|
| 1. El conjunto de reglas A::=BC, B::=λ, C::= λ, donde A,B,C Є $\Sigma_N$ y Axioma=A, puede transformarse en el conjunto equivalente A::=B, A::=C, A::=BC | | X |
| 2. La gramática cuyas reglas de producción son P={A::=BC \|B\|a, B::=b\|A , C::=c} es ambigua. | X | |
| 3. A::=A es una regla de redenominación. | | X |
| 4. Dos gramáticas son equivalentes si generan el mismo lenguaje. | X | |
| 5. A::=aBC es una regla en FNG. | X | |
| 6. Todo lenguaje generado por una G3 puede ser generado por una G2 equivalente. | X | |
| 7. A::= λ es una regla No generativa si y solo si A no es el axioma de la gramática. | X | |
| 8. Toda Gramática de Tipo 1 es también una Gramática de Tipo 2. | | X |
| 9. Si queremos eliminar la recursividad a izquierdas de la siguiente gramática: G = ({a,b},{S},S,P) donde P={S::=aSb \| SS \| λ}, podemos obtener la siguiente gramática equivalente: G = ({a,b},{S,X},S,P) donde P={S::=aSb \| aSbX \| λ, X::=SX \| S } | X | |
| 10. Sólo las gramáticas de tipo 2 son ambiguas. | | X |

3. Obtener la gramática G' en FNC equivalente a G, explicando brevemente las transformaciones en la gramática, paso a paso:
   G=({a,b,d}, {A, B,C,D,E,F,G,H}, A, P)
   P = {A ::= aaDB | G| λ | aC
   B ::= Bb | b
   C ::= a | λ
   D ::= b | D
   E ::= E
   F ::= Bb | D | λ
   G ::= Ga | dHb
   H ::= bbG}

Recordad: Antes de pasar a FNC es necesario *bien formar* G. Para ello, se deberá eliminar: 1. Reglas Innecesarias, 2. Símbolos inaccesibles, 3. Reglas superfluas y símbolos no generativos, 4. Reglas no generativas y 5. Reglas de Redenominación. Calificación máxima: 4 puntos.

**SOLUCIÓN:**

**Antes de pasar a FNC, bien formamos G:**

1. Eliminar Reglas innecesarias (Reglas del tipo *A::=A*)
   En este caso, eliminamos las siguientes reglas de producción:
   D::=D y E::=E

2. Eliminar Símbolos Inaccesibles (Todo símbolo $U \in \Sigma_N$ no inaccesible cumple S $\stackrel{*}{\rightarrow}$ xUy.)
   En este caso, los símbolos No Terminales: E y F son inaccesibles, ya que no puede accederse a ellos desde el Axioma.
   Así, eliminamos estos símbolos y sus reglas (F::=Bb | D | λ).

3. Eliminar Reglas superfluas y símbolos no generativos (Reglas que no contribuyen a la formación de palabras $x \in \Sigma_T{}^*$).
   Para ello, marcamos primeramente los símbolos No Terminales generativos (aquellos que en la parte derecha de al menos una de sus producciones tienen únicamente terminales).
   Así, L0 = {A, B, C, D}.

   A continuación, marcamos aquellos símbolos que en la parte derecha de una de sus producciones tienen tanto terminales como los símbolos previamente marcados ({A, B, C, D}).
   Así, L1 = {A,B,C,D}

   Con lo cual, los símbolos G y H son no generativos, y podremos eliminarlos de $\Sigma_N$. Además, eliminaremos las reglas en las que aparecen (reglas supérfluas):
   *G ::= Ga | dHb , H ::= bbG* y *A::=G*

4. <u>Reglas No Generativas. (Reglas del tipo A::= λ (donde A≠Axioma))</u>
   Eliminamos C::= λ,
   Añadiendo así la regla A::=a (porque A.:=aC)

5. <u>Reglas de Redenominación (Reglas del tipo *A::=B* (donde A ≠ B))</u>
   No hay, la gramática no se modificará en este punto.

Así, la gramática Bien Formada será:

**$G_{\text{Bien Formada}}$=({a,b}, {A, B,C,D}, A, $P'_{bf}$)**
**$P'_{bf}$ = {A ::= aaDB | λ | aC | a**
**B ::= Bb | b**
**C ::= a**
**D ::= b }**

**Una vez bien formada G, podremos transformarla a FNC.**

Para ello, modificamos las reglas de producción del siguiente modo:

A ::= aaDB se transforma en:
**A :::= CE** (donde E::=aDB, pero esta regla no está en FNC)
(en este caso, no es necesario añadir un nuevo NT, ya que *C::=a* ya está en las reglas de producción, y <u>C sólo genera a.</u> Si no fuera así, deberíamos crear una nueva regla de producción).
**E ::= CF**
**F ::= DB**

A ::= aC se transforma en:
**A ::= CC**

B::=Bb se transforma en:
**B ::= BD**
(en este caso, no es necesario añadir un nuevo NT, ya que *D::=b* ya está en las reglas de producción, y <u>D sólo genera b.</u> Si no fuera así, deberíamos crear una nueva regla de producción).

**Así, la Gramática EQUIVALENTE a G en FNC será:**
**$G_{FNC}$=({a,b}, {A, B,C,D,E,F}, A, $P'_{FNC}$)**
**$P'_{FNC}$ = {A ::= λ | a | CE | CC**
**B ::= Bb | b**
**C ::= a**
**D ::= b**
**E ::= CF**
**F ::= DB}**