# Pauta de Corrección

# Certamen Global
# Introducción a la Informática Teórica

19 de diciembre de 2011

1. Primeramente, si un lenguaje es regular, es de contexto libre. El contrapositivo es que si no es de contexto libre tampoco es regular.

   El lema de bombeo para lenguajes regulares: Si $\mathscr{L}$ es regular, hay una constante $N$ tal que para todo $\sigma \in \mathscr{L}$ si $|\sigma| \geq N$ puede escribirse $\sigma = \alpha\beta\gamma$, con $|\alpha\beta| \leq N$, $\beta \neq \epsilon$ tales que para todo $k \geq 0$ se cumple $\alpha\beta^k\gamma \in \mathscr{L}$.

   El lema de bombeo para lenguajes de contexto libre: Si $\mathscr{L}$ es de contexto libre, hay una constante $N$ tal que para todo $\sigma \in \mathscr{L}$ si $|\sigma| \geq N$ entonces puede dividirse $\sigma = uvxyz$ tales que $vy \neq \epsilon$, $|vxy| \leq N$ y para todo $k \geq 0$ la palabra $uv^kxy^kz \in \mathscr{L}$.

   (a) Este lenguaje es finito, por lo que es regular. Siendo regular es de contexto libre.

   (b) Hay una relación entre el número de $a$, $b$ y $c$, lo que hace pensar que no es regular.

   Supongamos que fuera regular, en tal caso es aplicable el lema de bombeo para lenguajes regulares. Sea $N$ la constante del lema, elegimos $\sigma = a^N bc^{2N+3} \in \mathscr{L}$, $|\sigma| = 3N + 4 \geq N$. Resulta $\beta \in a^+$ por estar "cerca del comienzo," al repetirlo (por ejemplo, de elegir $k = 2$) deja de cumplirse la relación entre $a$, $b$ y $c$, con lo que no pertenece al lenguaje. No es regular.

   Es de contexto libre. Una gramática que genera este lenguaje es:

   $$\begin{aligned} S &\to aScc \mid aAcc \\ A &\to bAccc \mid bccc \end{aligned}$$

   (c) Otro caso clásico de lenguaje no regular. Claro que si $|\Sigma| = 1$, son palabras de largo par, y el lenguaje es simplemente $(aa)^+$, que es regular y por tanto de contexto libre.

   En caso contrario, al menos habrán dos símbolos en $\Sigma$, llamémosles $a$ y $b$. Supongamos que el lenguaje es regular, y sea $N$ la constante del lema de bombeo. Consideremos $\sigma = a^N bba^N \in \mathscr{L}$, $|\sigma| = 2N + 2 \geq N$. Podemos dividir $\sigma = \alpha\beta\gamma$, donde $\beta \in a^+$. Al repetir $\beta$ dos veces, las $bb$ dejan de estar en el centro y el resultado no pertenece al lenguaje.

   El lenguaje es de contexto libre, una gramática es:

   $$S \to xSx \mid xx \qquad \text{para todo } x \in \Sigma$$

   (d) Nuevamente, si $|\Sigma| = 1$, el lenguaje es simplemente $(aa)^+$, que es regular y por tanto de contexto libre.

   En caso contrario, al menos habrán dos símbolos en $\Sigma$, llamémosles $a$ y $b$. En este caso no es de contexto libre, lo que demostramos mediante el lema de bombeo. Supongamos que es de contexto libre, sea $N$ la constante del lema y elegimos $\sigma = a^N b^N a^N b^N$. Por las condiciones de $vxy$, esto puede ser un solo símbolo, y al usar $k = 0$ el resultado tendrá menos de ese símbolo en la primera o segunda mitades, no pertenece al lenguaje. Si $v$ o $y$ están formados por dos tipos de símbolo: Si están en la primera mitad, al repetirlos deja de ser simétrico. Lo mismo si se encuentran en la segunda mitad, se produce la misma inconsistencia. La otra opción sería que estuviera en el centro, lo que al repetir aumenta las $b$ y las $a$, tampoco pertenece al lenguaje. No es de contexto libre, y por tanto tampoco regular.

## Puntajes

| | | |
|---|---|---|
| **Total** | | 20 |
| Regular $\Rightarrow$ Contexto libre | | 2 |
| Lema de bombeo, lenguajes regulares | | 4 |
| Lema de bombeo, lenguajes de contexto libre | | 4 |
| a) Finito, por tanto regular y contexto libre | | 3 |
| b) | | 5 |
| No regular | 3 | |
| Contexto libre | 2 | |
| c) | | 5 |
| No regular | 3 | |
| Contexto libre | 2 | |
| d) | | 5 |
| No contexto libre | 4 | |
| No regular | 1 | |

2. Un lenguaje regular puede estar dado por una expresión regular, un autómata finito no determinista (NFA), un autómata finito determinista (DFA) o una gramática regular. Si está dado por un DFA, podemos también suponer que es mínimo.

   (a) Supongamos dado el DFA mínimo para el lenguaje. Si no tiene estados finales, el lenguaje es vacío.

   (b) Nuevamente suponemos dado el DFA mínimo $M = (Q, \Sigma, \delta, q_0, F)$, donde $\mathscr{L} = \mathscr{L}(M)$. El DFA $\overline{M} = (Q, \Sigma, \delta, q_0, Q \smallsetminus F)$ claramente acepta exactamente si $M$ no lo hace, y así $\overline{\mathscr{L}} = \mathscr{L}(\overline{M})$. Para determinar si el lenguaje aceptado por $\overline{M}$ es finito, basta ver si hay ciclos en el autómata.

   (c) Sabemos que la intersección entre lenguajes regulares es regular. En efecto, si $\mathscr{L}_1 = \mathscr{L}(M_1)$ y $\mathscr{L}_2 = \mathscr{L}(M_2)$, donde $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ y $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ son DFAs, entonces $M = (Q_1 \times Q_2, \delta, (q_1, q_2), F_1 \times F_2)$ donde $\delta((q', q''), a) = (\delta_1(q', a), \delta_2(q'', a))$ acepta $\mathscr{L}_1 \cap \mathscr{L}_2$. Entonces basta determinar si $\mathscr{L} \cap (\Sigma\Sigma)^+$ no es vacío para saber si $\mathscr{L}$ contiene palabras de largo par.

## Puntajes

| | | | |
|---|---|---|---|
| **Total** | | | 20 |
| a) | | 5 | |
| b) | | 7 | |
| c) | | 8 | |
| Intersección | 5 | | |
| Determinar vacío | 3 | | |

3. Usaremos la estrategia de ir contabilizando la diferencia entre el número de $a$ y $b$ vistos hasta el momento. Una diferencia positiva se representa mediante $A$s en la pila, una negativa mediante $B$s. Si viene una $a$ en la entrada, y en el stack hay $Z_0$ (corresponde a diferencia cero) o $A$, hay que contabilizar una $a$ adicional, agregando una $A$ a la pila; si en la pila hay una $B$, la eliminamos (la diferencia absoluta disminuye en uno). El caso de $b$ en la entrada es simétrico. Siempre que la diferencia sea nula ($Z_0$ en la pila) damos la posibilidad de aceptar.

   Un autómata construido de esta forma lo muestra la figura 1.

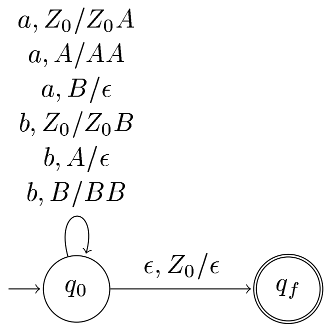


Figure 1: El autómata apilador

## Puntajes

| | |
|---|---|
| **Total** | 15 |
| Discusión | 15 |

4. El número posible de descripciones instantáneas de un LBA es finito. Por lo tanto, es posible generar las descripciones instantáneas del autómata durante el cómputo y almacenarlas. Si aparece una descripción instantánea ya generada antes, el autómata está en un ciclo y no acepta; si llega a "trabarse" o aceptar tenemos la respuesta.

### Puntajes

| | |
|---|---|
| **Total** | 25 |
| Discusión | 25 |

5. Cada término por turno.

   (a) Una *reducción* de un problema $P_1$ al problema $P_2$ es un algoritmo que traduce instancias de $P_1$ en instancias de $P_2$ con la misma solución.

   (b) Una *reducción polinomial* es una reducción que se ejecuta en tiempo acotado por un polinomio en el largo de los datos de entrada.

   (c) Un problema está en $\mathscr{NP}$ si el lenguaje correspondiente es aceptado por una máquina de Turing en tiempo polinomial en el largo de la entrada,

   (d) Un problema es *$\mathscr{NP}$-completo* si está en $\mathscr{NP}$ y es $\mathscr{NP}$-duro (todo problema en $\mathscr{NP}$ puede reducirse polinomialmente a él).

   (e) Un problema está en $\mathscr{P}$ si el lenguaje correspondiente es aceptado por una máquina de Turing determinista en tiempo polinomial en el largo de la entrada.

   (f) Un problema es *decidible* si el lenguaje correspondiente es aceptado por una máquina de Turing determinista que siempre se detiene. En caso contrario, es *no decidible*.

## Puntajes

| | | |
|---|---|---|
| **Total** | | 20 |
| a) | 2 | |
| b) | 2 | |
| c) | 4 | |
| d) | 5 | |
| e) | 3 | |
| f) | 4 | |

6. Primeramente, definimos nuestros términos. Ya definimos *reducción* y *reducción polinomial* antes.

   - Un *problema en $\mathscr{P}$* es uno que puede resolver una máquina de Turing determinista en tiempo acotado por un polinomio en el largo de la entrada
   - Un *problema en $\mathscr{NP}$* es uno que puede resolver una máquina de Turing no determinista en tiempo acotado por un polinomio en el largo de la entrada
   - Un problema es *$\mathscr{NP}$-duro* si todo problema en $\mathscr{NP}$ puede reducirse polinomialmente a él
   - Un problema es *$\mathscr{NP}$-completo* si es $\mathscr{NP}$-duro y está en $\mathscr{NP}$
   - Un problema es *no decidible* si no hay máquina de Turing determinista que siempre se detiene que lo resuelva

   Con estas definiciones en mente:

   (a) No dice mucho, la reducción puede perfectamente tomar mucho tiempo.

   (b) Los problemas en $\mathscr{P}$ son un subconjunto de los en $\mathscr{NP}$, esta reducción no es nada sorprendente.

   (c) Esto es imposible, los problemas en $\mathscr{NP}$ son decidibles, y esta reducción haría decidible a $I$.

   (d) Es parte de la definición, todo problema en $\mathscr{NP}$ puede reducirse polinomialmente a cualquiera que sea $\mathscr{NP}$-completo.

   (e) Esta reducción significa que $D$ está en $\mathscr{NP}$, cxon lo que $D$ es $\mathscr{NP}$-completo.

## Puntajes

| **Total** | | 30 |
|---|---|---|
| a) | 5 | |
| b) | 5 | |
| c) | 5 | |
| d) | 5 | |
| e) | 5 | |
| f) | 5 | |