Pauta de Corrección

Certamen Recuperativo
Introducción a la Informática Teórica

5 de diciembre de 2014

1. Por turno.

   a) Los lenguajes sensibles al contexto son generados por gramáticas cuyas producciones tienen la forma:

   $$\alpha \to \beta$$

   donde $\alpha \in V^* N V^*$ (o sea, contiene al menos un no-terminal) y $\beta \in V^*$ con la restricción que $|\alpha| \leq |\beta|$. Consideremos $L = \mathscr{L}(G)$, donde $G = (N, \Sigma, P, S)$ es una gramática sensible al contexto. Construimos la gramática $G' = (N \cup \{S'\}, \Sigma, P', S')$ con un nuevo no-terminal S', agregando las producciones:

   $$S' \to S'S \mid \epsilon$$

   Entonces $L^* = \mathscr{L}(G')$. En rigor, esta gramática no es sensible al contexto (ninguna gramática sensible al contexto puede generar ϵ), pero si cambiamos las producciones agregadas a:

   $$S' \to S'S \mid S$$

   genera $L^+ = L^* \smallsetminus \{\epsilon\}$, que podemos considerar suficientemente cercano.

   b) Sean los lenguajes:

   $$\begin{aligned} A &= \mathscr{L}(((b \mid c)^* a (b \mid c)^* a (b \mid c)^* a (b \mid c)^*)^*) \\ B &= \mathscr{L}((a \mid c)^* b ((a \mid c)^* b (a \mid c)^* b)^*) \\ C &= \mathscr{L}((a \mid b \mid c)^* abc (a \mid b \mid c)^*) \end{aligned}$$

   En palabras, A consta de palabras con un número de a que es divisible por 3, B son palabras con un número impar de b, y C son las palabras que contienen abc. Como los tres se expresan en términos de expresiones regulares, los tres son regulares. Ahora bien, el lenguaje solicitado puede expresarse:

   $$L_1 = (A \cap B) \cap \overline{C}$$

   Como los lenguajes regulares son cerrados respecto de intersección y complemento, L_1 es regular.

   c) Si L_a y L_b están en NP, sabemos que hay máquinas de Turing no deterministas M_a y M_b que aceptan la palabra ω en tiempos acotados respectivamente por polinomios $p_a(|\omega|)$ y $p_b(|\omega|)$. La construcción dada por la figura 1 muestra que el tiempo total no-determinista para aceptar ω si pertenece a la intersección está acotado por

   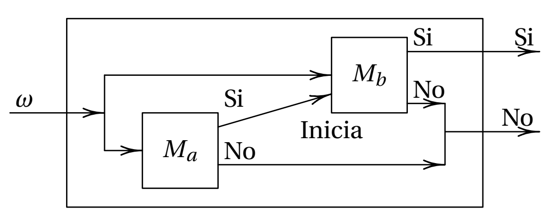

   Figura 1: Construcción para intersección de lenguajes en NP

   $p_1(|\omega|) + p_2(|\omega|)$, que claramente es un polinomio en $|\omega|$.

Puntajes

Total		30
a) Construcción/explicación	10	
b) Construcción/explicación	10	
c) Construcción/explicación	10	

2. Este lenguaje no es de contexto libre, cosa que demostramos por contradicción. Supongamos que L_2 es de contexto libre, y sea N la constante del lema de bombeo de lenguajes de contexto libre. Consideremos:

   $$\sigma = a^N b^{N^2}$$

   Entonces $|\sigma| = N + N^2 \geq N$, por lo que por el lema podemos escribir:

   $$\sigma = uvwxy$$

   con $|vx| > 0$, $|vwx| \leq N$ tales que para todo $k \in \mathbb{N}_0$:

   $$uv^k wx^k y \in L'$$

   Es claro que v ni x pueden estar formados por más de un símbolo, ya que de de caso contrario la palabra bombeada no estaría en $\mathscr{L}(a^* b^*)$. Además, v debe estar formada únicamente por a y x únicamente por b, y ninguna de las dos puede ser vacía, ya que ambas partes deben crecer juntas. Concentrémonos en la parte formada por b, y tomemos $k = 2$. Entonces el número de b en $uv^2 wx^2 y$ cumple:

   $$N^2 < b \leq N^2 + N < N^2 + 2N + 1 = (N+1)^2$$

   El número de b no es un cuadrado perfecto, y esta palabra no pertenece a L_2, contradiciendo el lema de bombeo.

   Como todo lenguaje regular es de contexto libre, L tampoco es regular.

Puntajes

Total		20
Demostrar que L_2 no es de contexto libre	12	
Como L no es de contexto libre, no es regular	8	

Puntajes

3.

Total	25
Llevar a NFA	20

4. Que el algoritmo SIMPLEX en el peor caso tome tiempo exponencial no impide que en los casos de interés práctico su tiempo de ejecución esté en rango práctico.

   A pesar de su peor caso, y de conocerse algoritmos que demuestran que el problema está en P, sigue siendo popular ya que en muchos casos de interés práctico es más eficiente.

Puntajes

Total		20
Explicación	20	

5. El problema está en NP, ya que podemos adivinar los valores de las variables (estarán acotadas por el máximo de los b_i) y verificar que es factible.

   Podemos reducir 3-SAT a INTEGER LINEAR PROGRAMMING asignando una variable entera v_j a la variable lógica x_j, agregando restricciones $0 \leq v_j \leq 1$ (que al ser entera v_j hacen que solo pueda tomar esos valores). La idea es que $v_j = 0$ si x_j es falso, y $v_j = 1$ si x_j es verdadero. Para cada cláusula agregamos una restricción. Por ejemplo, para $x_1 \vee \overline{x_2} \vee x_3$ usamos $v_1 + (1 - v_2) + v_3 \geq 1$. Esto se cumple solo si alguno de los tres literales es verdadero, cumplir todas las restricciones significa que la fórmula es satisfacible. Vale decir, es NP-duro.

   Como el problema es NP-duro y está en NP, es NP-completo.

Puntajes

Total		25
El problema está en NP	10	
Reducción de 3-SAT a INTEGER LINEAR PROGRAMMING	10	
Conclusión	5	