Pauta de Corrección

Certamen Recuperativo
Introducción a la Informática Teórica

24 de julio de 2012

1. Por turno.

   (a) Es de contexto libre. Lo genera la gramática

   $$S \to aSaa \mid abaa$$

   No es regular. Supongamos que $\mathscr{L}_a$ fuera regular, sea N la constate del lema de bombeo. Elegimos $\sigma = a^N ba^{2N}$, para el cual $|\sigma| = 3N+1 \geq N$ por lo que podemos escribir$\sigma = \alpha\beta\gamma$ con $|\alpha\beta| \leq N$, $\beta \neq \epsilon$ tal que para todo $k \in \mathbb{N}_0$ $\alpha\beta^k\gamma \in \mathscr{L}_a$. Pero con estas condiciones β está formado únicamente por a, y al elegir $k = 0$ el número de a al comienzo y final no coincide.
   Otra manera de demostrarlo por contradicción usando propiedades de los lenguajes regulares. Supongamos que $\mathscr{L}_a$ fuera regular. Sea la substitución S_1:

   $$a \mapsto \{a, c\}$$
   $$b \mapsto \{b\}$$

   Entonces es regular el lenguaje:

   $$\mathscr{L}'_a = S_1(\mathscr{L}_a) \cap a^* bc^* = \{a^n bc^{2n} : n \geq 1\}$$

   Sea la substitución S_2 definida por:

   $$a \mapsto \{a\}$$
   $$b \mapsto \{\epsilon\}$$
   $$c \mapsto \{c\}$$

   Entonces es regular:

   $$\mathscr{L}''_a = S_2(\mathscr{L}'_a) = \{a^n c^{2n} : n \geq 1\}$$

   Sea el homomorfismo h:

   $$a \mapsto a$$
   $$b \mapsto cc$$

   Entonces es regular:

   $$\begin{aligned}\mathscr{L}'''_a &= h^{-1}(\mathscr{L}''_a) \\ &= h^{-1}(S_2(S_1(\mathscr{L}_a))) \\ &= \{a^n b^n : n \geq 1\}\end{aligned}$$

   Pero sabemos que $\mathscr{L}'''_a$ no es regular, con lo que tampoco lo es $\mathscr{L}_a$.

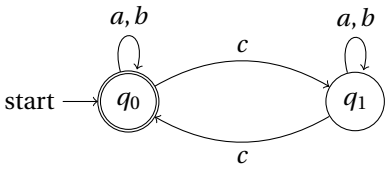

Figure 1: DFA para la pregunta 1c

(b) Podemos descomponer $\mathscr{L}_b$ de la siguiente forma:

$$\mathscr{L}_b = \{a^i b^j c^{i+j} : 0 \leq i, j \leq 10\} \cdot \{c^k : k \geq 3\}$$

El primer conjunto es finito, y por tanto regular. El segundo queda expresado por la expresión regular $cccc^*$, y también es regular. Siendo $\mathscr{L}_b$ la concatenación de dos lenguajes regulares, es regular. Siendo regular, es de contexto libre.

(c) Podemos construir $\mathscr{L}_c$ mediante lo siguiente:

- Sea $\mathscr{L}_1$ el conjunto de strings sobre Σ que contienen aba. $\mathscr{L}_1$ queda representado por la expresión regular $(a \mid b \mid c)^* aba (a \mid b \mid c)^*$, es regular.
- Sea $\mathscr{L}_2$ el conjunto de strings sobre Σ que contienen bab. $\mathscr{L}_1$ queda representado por la expresión regular $(a \mid b \mid c)^* bab (a \mid b \mid c)^*$, es regular. Como $\mathscr{L}_2$ es regular, lo es $\overline{\mathscr{L}_2}$.
- Sea $\mathscr{L}_3$ el conjunto de strings que contiene un número par de c. Es aceptado por el DFA de la figura 1, con lo que también es regular.

En términos de los anteriores:

$$\mathscr{L}_c = \mathscr{L}_1 \cap \overline{\mathscr{L}_2} \cap \mathscr{L}_3$$

Como los puntos de partida son regulares, y las operaciones preservan regularidad, $\mathscr{L}_c$ es regular. Siendo regular, es de contexto libre.

(d) Como son strings formados únicamente por a, da lo mismo usar el lema de bombeo para lenguajes regulares o de contexto libre para demostrar que no es de la clase del caso.

Supongamos $\mathscr{L}_d$ de contexto libre, y sea N la constante del lema. Sabemos que hay infinitos primos, así que podemos elegir p y q primos tal que $pq \geq N$, y tomar $\sigma = a^{pq}$. Por el lema, podemos escribir:

$$\sigma = uvxyz$$

donde $0 < |vy| \leq N$ tal que para todo $k \in \mathbb{N}_0$

$$uv^k xy^k z \in \mathscr{L}_d$$

Pero:

$$|uv^k xy^k z| = |uvxyz| + (k-1)|vy|$$

Si tomamos $k = pq + 1$:

$$\begin{aligned} |uv^k xy^k z| &= pq + pq|vy| \\ &= pq(1 + |vy|) \end{aligned}$$

Como sabemos que $|vy| > 0$, $1 + |vy|$ tiene al menos un factor primo, y $uv^{pq+1}xy^{pq+1}z \neq \mathscr{L}_d$. Concluimos que $\mathscr{L}_d$ no es de contexto libre, con lo que tampoco es regular.

Puntajes

Total		30
a)		7
No es regular	4	
Es de contexto libre	3	
b)		7
Es regular	5	
Es de contexto libre	2	
c)		8
Es regular	6	
Es de contexto libre	2	
d)		8
No es de contexto libre	6	
No es regular	2	

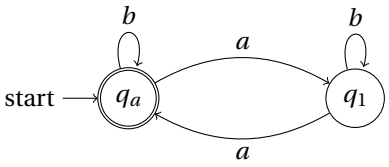

Figure 2: DFA M_a

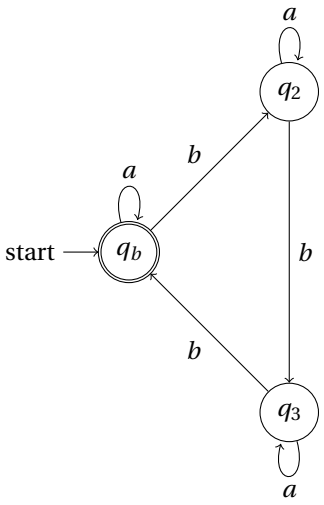

Figure 3: DFA M_b

2. Podemos construir directamente el autómata, pero resulta engorroso. Construiremos autómatas para strings sobre $\Sigma = \{a, b\}$ que tengan un número par de a ($M_a = (Q_a, \Sigma, \delta_a, q_a, F_a)$, ver figura 2) y otro para strings con un número de b divisible por 3 ($M_b = (Q_b, \Sigma, \delta_b, q_b, F_b)$, ver figura 3), y los combinamos el el DFA para la intersección $M = (Q_a \times Q_b, \Sigma, \delta, (q_a, q_b), F_a \times F_b)$, donde para todo $x \in \Sigma$:

$$\delta((q', q''), x) = (\delta_a(q', x), \delta_b(q'', x))$$

Puntajes

Total 15

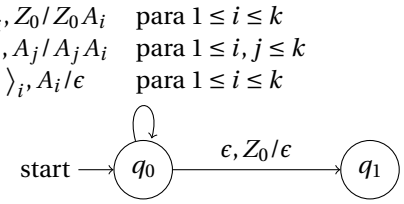

Figure 4: PDA para la pregunta 3

3. Una opción es construir un PDA que acepta por pila vacía que carge los abre paréntesis en su pila y los verifique con los cierre paréntesis respectivos, cuidando no trabarse en el proceso. La figura 4 da un posible diseño (la pila crece hacia la derecha), usando $\langle_i$ y $\rangle_i$ para abrir y cerrar el i-ésimo tipo de paréntesis en la entrada, respectivamente; y representa un paréntesis de tipo i mediante el símbolo A_i en la pila.

   Otra posibilidad es partir de la gramática para paréntesis balanceados:

   $$S \to (S)S \mid \epsilon$$

   y replicar la primera producción:

   $$S \to \epsilon$$
   $$S \to \langle_i S \rangle_i S \qquad \text{para todo } 1 \le i \le k$$

Puntajes

Total 25

4. La forma básica de demostrar que un problema no es decidible es reducir un problema que sabemos no es decidible a él. En nuestro caso, reducimos el problema de detectar `"Hola, mundo!"` al problema de determinar si el programa divide por cero.

   Tomamos un programa que podría escribir `"Hola, mundo!"`. Para evitar posibles "accidentes," revisamos el programa y antes de cada división verificamos que el divisor no es cero y en caso de serlo simplemente detenemos el programa. Enseguida, modificamos todas las instrucciones que generan salida de forma de escribir en un área interna, y luego de cada escritura verificamos si esa área contiene `"Hola, mundo!"`. En caso de ser así, ejecutamos la línea `1/0` (perfectamente legal en C). Si pudiéramos detectar división por cero, mediante este artilugio detectaríamos `"Hola, mundo!"`, lo que sabemos imposible.

Puntajes

Total	20
- Reducción demuestra no decidible	10
- Esbozar reducción	10

5. Un problema está en $\mathcal{NP}$ si basta "adivinar" la solución y luego verificarla, ambas operaciones en tiempo polinomial en el tamaño de la entrada. La entrada serán los grafos, por lo que podemos suponer que su tamaño es simplemente $|V_1| + |V_2| + |E_1| + |E_2|$.

   En nuestro caso, dados grafos $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$, verificamos que $|V_1| = |V_2|$ y que $|E_1| = |E_2|$ en tiempo proporcional a $|V_1| + |E_1|$. Podemos proponer un isomorfismo $\phi\colon V_1 \to V_2$ en tiempo proporcional a $|V_1|$, para luego verificar que para cada $(x, y) \in E_1$ se cumple $(\phi(x), \phi(y)) \in E_2$ y que para todo $(x, y) \in E_2$ se cumple $(\phi^{-1}(x), \phi^{-1}(y)) \in E_2$ en tiempo proporcional a $|E_1| + |E_2|$. El tiempo total claramente está acotado por un polinomio (incluso uno lineal) en el tamaño de la entrada).

   No se requieren respuestas formales, una explicación informal razonablemente precisa es suficiente.

Puntajes

Total 20

6. Claro que no. Sólo si la reducción es polinomial sabríamos que $\mathscr{P} = \mathscr{NP}$.

Puntajes

Total 10