Pauta de Corrección

Recuperación Segundo Certamen
Introducción a la Informática Teórica

6 de diciembre de 2011

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

   (a) Este lenguaje no es regular. Supongamos que lo fuera, entonces es aplicable el lema de bombeo para lenguajes regulares. Con N la constante del lema elegimos $a^N b^N c^{2N}$ en el lenguaje y suficientemente largo, lo que permite dividirlo en xyz con $|xy| \leq N$ (o sea, xy consta sólo de a), $y \neq \epsilon$ tal que $xy^k z$ es parte del lenguaje para todo $k \geq 0$. Pero al elegir $k = 0$ deja de cumplirse la relación entre las a, las b y las c, con lo que el resultado no pertenece al lenguaje. Esta contradicción demuestra que no es regular.
   Es de contexto libre, una gramática es como sigue:

   $$S \longrightarrow aSc \mid aBc$$
   $$B \longrightarrow bBc \mid bc$$

   (b) No hay relación entre el número de a, b y c. Esto es simplemente el lenguaje denotado por la expresión regular $(aaa)^+(bb)^+c^+$, y por tanto es regular. Como es regular, es de contexto libre.

   (c) Esta no es de contexto libre, al tener atadas las a, b y c.
   Supongamos que fuera de contexto libre, en tal caso es aplicable el lema de bombeo para lenguajes de contexto libre. Sea N la constante del lema, elegimos $\sigma = a^N b^{2N} c^{3N}$, parte del lenguaje y suficientemente largo. Podemos entonces escribir:

   $$\sigma = uvxyz$$

   tal que $vy \neq \epsilon$, y para todo $k \geq 0$ la palabra $uv^k xy^k z$ es parte del lenguaje. Ahora bien, tanto v como y están formadas por un único símbolo, porque de otra manera uv^2xy^2z no tendría la forma $a^*b^*c^*$ y no pertenece al lenguaje. Pero entonces al repetir v e y podemos a lo más aumentar el número de dos tipos de símbolos, jamás de los tres, y la palabra uv^2xy^2z no pertenece al lenguaje. Esta contradicción demuestra que el lenguaje no es de contexto libre. Como no es de contexto libre, no es regular.

Puntajes

Total		30
a)		10
No es regular	5	
Es de contexto libre	5	
b)		10
Regular	5	
Contexto libre	5	
c)		10
No de contexto libre	5	
No es regular	5	

2. Para minimizar el número de pasos, usaremos siempre la opción de generar k terminales en una producción, con lo que el número mínimo de pasos es:

$$P_{\min} = \left\lceil \frac{n}{k} \right\rceil$$

Para maximizar el número de pasos, debemos usar en los últimos pasos producciones de la forma $A \to a$, cada una de las cuales aporta un terminal. Aplicar alguna de las producciones con no-terminales siempre aporta k terminales, la manera de minimizar ésto es aplicar una sola de ellas. De esta forma el número de pasos máximo es:

$$P_{\max} = \begin{cases} n - k + 1 & \text{si } n \geq k \\ 1 & \text{caso contrario} \end{cases}$$

Puntajes

Total	20
a)	10
b)	10

3. La idea es ir acumulando una marca (por ejemplo A) cada vez que se lee ab (para esto basta contabilizar las a, asegurándose que vienen en pares con b), para luego descontar una marca con cada c. Finalmente consumimos las a finales. Un autómata apilador que funciona con este criterio lo da la figura 1.

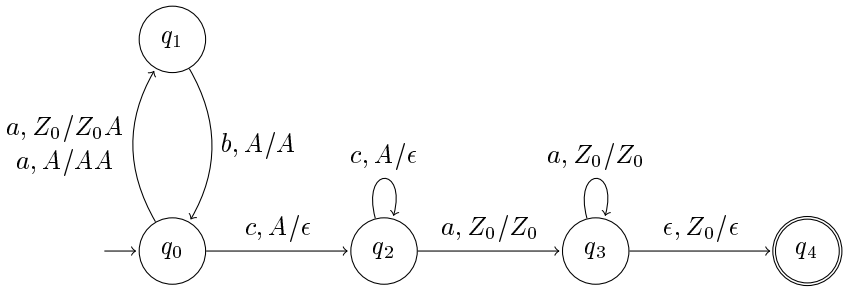

Figure 1: El autómata apilador

Puntajes

Total	15
– Explicación diseño	5
– Autómata	10

4. Una gramática para ésto es la siguiente:

$$S \longrightarrow Sa \mid Aa$$
$$A \longrightarrow abAc \mid abc$$

La idea es que S genere A con la cola de a, luego A genera pares ab con su correspondiente c.

Puntajes

Total	15
– Explicación diseño	5
– Autómata	10

5. Cada punto por turno.

   (a) Si $\mathscr{L}$ y su complemento son recursivamente enumerables, basta monitorear la salida de las máquinas que enumeran esos dos lenguajes. La palabra σ dada aparecerá en una de las dos tarde o temprano, indicando si $\sigma \in \mathscr{L}$ o no.

   (b) Por el punto 5a, si fuera recursivamente enumerable $\overline{\mathscr{L}}$, $\mathscr{L}$ sería recursiva.

   (c) Como indica la pista, podemos generar programas en orden lexicográfico, y cada vez que se genera un nuevo programa ejecutar una instrucción más de cada uno de los programas que ya tenemos. Si esta instrucción hace que el programa escriba "Hola, mundo", lo copiamos a la cinta de salida.

Puntajes

Total	35
a)	13
b)	10
c)	12

6. Primeramente, un problema es determinar si una palabra σ pertenece o no a un lenguaje $\mathscr{L}$ dado, con lo que podemos identificar problemas con lenguajes. A la palabra σ le llamaremos instancia del problema.

   (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 mediante un algoritmo determinista que toma tiempo acotado por un polinomio en el largo de la instancia.

   (c) Un problema se dice que está en $\mathscr{P}$ si hay una máquina de Turing determinista que reconoce ese lenguaje en tiempo acotado por un polinomio en el largo de la instancia.

   (d) Un problema se dice que está en $\mathscr{NP}$ si hay una máquina de Turing no determinista que reconoce ese lenguaje en tiempo acotado por un polinomio en el largo de la instancia.

   (e) Un problema se llama $\mathscr{NP}$-*duro* si todos los problemas en $\mathscr{NP}$ pueden reducirse polinomialmente a él.

   (f) Un problema se llama $\mathscr{NP}$-*completo* si es $\mathscr{NP}$-duro y está en $\mathscr{NP}$.

Puntajes

Total	30
a)	5
b)	5
c)	5
d)	5
e)	5
f)	5