Pauta de Corrección

Segundo Certamen
Introducción a la Informática Teórica

26 de noviembre de 2011

1. Primeramente, sabemos que los lenguajes regulares son un subconjunto de los de contexto libre. En consecuencia, si un lenguaje es regular automáticamente es de contexto libre; y el contrapositivo de ésto es que si no es de contexto libre no es regular.

   (a) Este lenguaje es finito, y por lo tanto regular. Como es regular, es de contexto libre.

   (b) Es de contexto libre. Una gramática que lo genera es:

   $$S \to aaaAc$$
   $$A \to aaaAc|B$$
   $$B \to bbB|bb$$

   No es regular (hay relación entre un par de números de símbolos). Aplicamos el lema de bombeo para lenguajes regulares: Supongamos que el lenguaje fuera regular, y sea N la constante del lema de bombeo para lenguajes regulares. Elegimos $\sigma = a^{3N}bbc^N$ en el lenguaje, suficientemente largo para que se aplique el lema y que así podemos dividir en $\sigma = xyz$ con $|xy| \leq N$ donde $y \neq \epsilon$ tal que para todo $k \geq 0$ el string xy^kz es parte del lenguaje. Con estas condiciones, xy está formado sólo por a, y eligiendo $k = 0$ queda corto de a, no pertenece al lenguaje. Esta contradicción demuestra que no es regular.

   (c) Pareciera no ser de contexto libre, así que usamos el lema de bombeo respectivo para demostrar esto. Supongamos que el lenguaje es de contexto libre, es aplicable el lema de bombeo para lenguajes de contexto libre. Sea N la constante del lema, elegimos $\sigma = a^Nb^{N^2}$ en el lenguaje y suficientemente largo. Entonces por el lema podemos escribir $\sigma = uvxyz$ tal que $vy \neq \epsilon$ y para todo $k \geq 0$ el string uv^kxy^kz está en el lenguaje. Veamos cada uno de los casos posibles:

   - Si v ó y contienen símbolos a y b, al repetirlo el resultado deja de estar en a^+b^+, y no pertenece al lenguaje.
   - Si v e y ambos contienen sólo a o sólo b, al repetirlos se rompe la relación entre el número de a y b. Nuevamente no pertenece al lenguaje.
   - En vista de los anteriores, v contiene sólo a, mientras y contiene sólo b, y ninguno de los dos es ϵ.. Usemos v, y para representar los largos de los strings respectivos. Entonces dado k debe ser:

     $$(N + (k-1)v)^2 = N^2 + (k-1)y$$

     Esto es absurdo, se reduce a una ecuación para k que a lo más tiene dos soluciones, cuando por el lema debiera valer para todo $k \geq 0$.

   Como no es de contexto libre, no puede ser regular.

Puntajes

Total			30
a)		10	
	Finito, luego regular	5	
	Regular, luego de contexto libre	5	
b)		10	
	De contexto libre (p.ej. vía CFG)	5	
	No es regular (lema de bombeo)	5	
c)		10	
	No es de contexto libre (lema de bombeo)	5	
	Como no es de contexto libre, no es regular	5	

2. (a) En la forma normal de Chomsky, todas las producciones tienen una de las formas $A \to BC$ con $A, B, C \in N$ ó $A \to a$, con $A \in N$ y $a \in \Sigma$. Cada vez que se aplica una de las producciones del primer tipo la forma sentencial se alarga en 1, para llegar a largo n se requieren $n-1$ de éstas; además deben aplicarse n producciones del segundo tipo para llevar los no-terminales a terminales. El número de producciones es fijo, y es $n - 1 + n = 2n - 1$.

   (b) Cada vez que se aplica una de las producciones de la gramática en forma normal de Greibach se introduce exactamente un terminal. Para introducir n terminales se requieren por lo tanto n pasos.

Puntajes

Total		25
a)		15
	Forma normal de Chomsky	5
	Razonamiento	10
b)		10
	Razonamiento	10

3. Un autómata de pila que acepta el lenguaje indicado, con la convención que la pila crece hacia la derecha se ilustra en la figura 1. Consume a sin cambiar la pila, luego acumula sucesivas b en la pila.

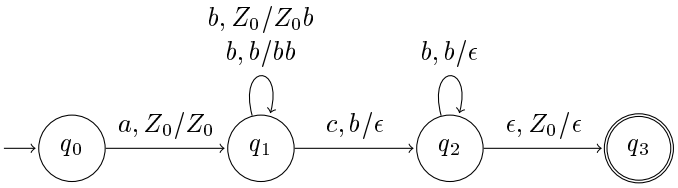

Figure 1: Autómata apilador que acepta $\{ab^k c^k : k \geq 1\}$

Con la primera c elimina una b, por cada c consecutiva consume una b de la pila. Sólo si el número de b y c coinciden tendrá la posibilidad de eliminar Z_0 de la pila y pasar al estado final. Como debe consumir al menos una c, debe consumir al menos una b.

Puntajes

Total 20

4. Primeramente, un *problema* consiste en determinar si un string σ pertenece a un lenguaje $\mathscr{P}$.

   (a) En este punto, por turno:

   - Una *reducción polinomial* de un problema P_1 a un problema P_2 es un algoritmo (máquina de Turing determinista que siempre se detiene) que traduce $\sigma_1 \in P_1$ en $\sigma_2 \in P_2$ y $\sigma_1 \notin P_1$ en $\sigma_2 \notin P_2$, y en el proceso toma a un número de pasos acotado por un polinomio en $|\sigma_1|$.
   - Un problema está en $\mathscr{NP}$ si hay una máquina de Turing que reconoce el strings en el lenguaje en un número de pasos acotado por un polinomio en el largo de la entrada.
   - Un problema es $\mathscr{NP}$*-duro* si todos los problemas en $\mathscr{NP}$ pueden reducirse polinomialmente a él.
   - Un problema es $\mathscr{NP}$*-completo* si es $\mathscr{NP}$-duro y está en $\mathscr{NP}$.

   (b) Como P_c es $\mathscr{NP}$-completo, todo problema en $\mathscr{NP}$ puede reducirse polinomialmente a él, y por tanto (como la composición de polinomios es un polinomio) a P. Esto hace que P sea $\mathscr{NP}$-duro, y al estar en $\mathscr{NP}$ es $\mathscr{NP}$-completo.

   (c) Esto no permite concluir nada nuevo, al ser P_c $\mathscr{NP}$-completo es $\mathscr{NP}$-duro, y todos los problemas en $\mathscr{NP}$ pueden reducirse polinomialmente a él.

Puntajes

Total			30
a) Definiciones		15	
Reducción polinomial	5		
$\mathscr{NP}$-duro	5		
$\mathscr{NP}$-completo	5		
b)		10	
P es $\mathscr{NP}$-duro	5		
P es $\mathscr{NP}$-completo	5		
c) No dice nada nuevo		10	

5. Reduciremos el problema de detectar "Hola, mundo" al problema de determinar si se le asigna un valor a la variable a. Si pudiéramos resolver este último, podríamos reolver el primero.

   Tomemos un programa cualquiera. Lo modificaremos de tal forma que sólo cuando escribe "Hola, mundo" le asigna un valor a la variable a. Para evitar "accidentes," cambiamos el nombre a todas las variables a, de forma de no alterar el significado del programa. Por ejemplo, podríamos anteponer x a cada identificador del programa. Esto es simple de hacer. Luego identificamos todos los puntos en que escribe, y registramos lo escrito en una nueva variable auxiliar. Si lo último que ésta contiene es "Hola, mundo", ejecutamos la sentencia a = 1.

   Si pudiéramos determinar si se asigna un valor a a, podríamos determinar si un programa escribe "Hola, mundo", lo que sabemos imposible.

Puntajes

Total	20
Reducción de detección de "Hola, mundo" a este problema	15
Conclusión	5