CC30B - Pauta Control 2

Prof. Ricardo Baeza Yates

P1 (2 ptos)

Escriba una gramática para el lenguaje de todas las palabras de la forma $a^n b^m c^p d^q$ tal que $m+n = p+q,\ m,n,p,q \geq 0$.

(1.5 ptos) Primero generamos los dos lados que son iguales: a y b por un lado y luego c y d. Usamos distintos no terminales para indicar que estamos colocando cada una de estas letras y para que el orden sea el correcto (T_1 no genera d's, T_2 no genera a's y T_3 no genera ni a's ni d's). Luego tenemos:

$$\begin{aligned}
S &\rightarrow \epsilon \mid aSd \mid aT_1c \mid bT_2d \mid bT_3c \\
T_1 &\rightarrow \epsilon \mid aT_1c \mid bT_3c \\
T_2 &\rightarrow \epsilon \mid bT_2d \mid bT_3c \\
T_3 &\rightarrow \epsilon \mid bT_3c
\end{aligned}$$

(0.5 ptos) Indique la derivación más a la izquierda para la palabra $abbccd$:

$$S \rightarrow aSd \rightarrow abT_3cd \rightarrow abbT_3ccd \rightarrow abbccd$$

P2 (2 ptos)

Suponga que tiene una calculadora lógica que usa notación polaca (tipo HP). Es decir, primero se ingresan los dos operandos y luego la operación. Considerando los valores V y F y las operaciones $+$ (or), $*$ (and) y $-$ (complemento), especifique un autómata de pila que reconoce una secuencia válida de operaciones y queda en un estado final si el último valor es V.

Sea $M = (K, \Gamma, \Sigma, \Delta, s, F)$ el autómata de pila. Usaremos un caracter extra C para marcar el comienzo del cálculo. Luego tenemos que $K = \{s, q, f\}$, $\Gamma = \Sigma \cup \{C\}$, $\Sigma = \{V, F, *, +, -\}$ y $F = \{f\}$. La relación de transición Δ es

$((s,\epsilon,\epsilon),(q,C))$	marcamos el comienzo
$((q,V,C),(q,VC))$	agregamos un valor al comienzo
$((q,F,C),(q,FC))$	agregamos un valor al comienzo
$((q,F,F),(q,FF))$	agregamos un valor al stack
$((q,F,V),(q,FV))$	agregamos un valor al stack
$((q,V,V),(q,VV))$	agregamos un valor al stack
$((q,V,F),(q,VF))$	agregamos un valor al stack
$((q,-,F),(q,V))$	complemento
$((q,-,V),(q,F))$	complemento
$((q,+,FF),(q,F))$	or
$((q,+,FV),(q,V))$	or
$((q,+,VF),(q,V))$	or
$((q,+,VV),(q,V))$	or

$$\begin{array}{ll}
((q,*,FF),(q,F)) & \text{and} \\
((q,*,FV),(q,F)) & \text{and} \\
((q,*,VF),(q,F)) & \text{and} \\
((q,*,VV),(q,V)) & \text{and} \\
((q,\epsilon,VC),(f,\epsilon)) & \text{resultado final es } V
\end{array}$$

P3 (2 ptos)

Escoja una de las dos siguientes preguntas:

a) Demuestre que el lenguaje $L = \{\ a^i b^j c^i d^j,\ i,j \geq 0\}$ no es libre del contexto.

Usando el teorema del bombeo visto en clases, no es posible demostrar que este lenguaje no es libre del contexto, pues hay un caso en que el bombeo funciona. Si demuestra esto hay 0.5 ptos. Pero hay otras alternativas.

Una de ellas es modificar el lema del bombeo notando que no solo $w = uvxyz$ tiene un largo mínimo, sino que también vxy tiene un largo máximo si ϵ no pertenece al lenguaje. Basta con hacer la misma demostración hecha en clases y ver el árbol de derivación resultante. En nuestro problema, podemos considerar el caso $i,j > 0$ (la diferencia es un lenguaje regular que contiene a ϵ). Luego, sea n el largo máximo y considere el string $a^n b^n c^n d^n$ que pertenece a L, luego como $|vxy| \leq n$, entonces vxy tiene a lo más dos letras distintas y estas deben ser consecutivas. Si vy tiene sólo a'sm entonces uxz tiene menos $a's$ que c's y entonces no está en L. Lo mismo si vy sólo contiene sólo b's, c's o d's. Si vy tiene a's y b's, entonces uxz todavía tiene menos a's que c's y entonces no está en L. Una contradicción similar ocurre si vy contiene b's y c's o c's y d's. Como éstas son todas las posibilidades, L no es LLC.

Otra posibilidad es manipular el lenguaje suponiendo que es LLC y llegar a un lenguaje que no es LCC vía propiedades de clausura de los LLC (unión, intersección, clausura de Kleene e intersección con un lenguaje regular). Luego por contradicción no puede ser LLC.

b) Si L es un LLC y R es regular entonces ¿$L - R$ es LLC? ¿Qué pasa con $R - L$? Justifique su respuesta.

Usamos una demostración similar a la demostración de que la intersección de un LLC y un lenguaje regular es LLC. Simulamos con un autómata de pila M, el autómata de pila que reconoce L y el autómata finito que reconoce R, en paralelo. Los estados de M entonces serán de la forma (q,p) donde q y p son, respectivamente, el estado en que está el autómata de pila y el autómata finito que estamos simulando. Si cuando leemos toda la entrada, el estado es (f, nf) donde f es final y nf no es final, pasamos a un estado especial que es el único estado final de M. Es decir, reconoce cuando la palabra pertenece a L pero no a R, es decir $L - R$. Por lo tanto, $L - R$ es LLC ya que es reconocido por un autómata de pila. El caso $R - L$ es similar, pues aceptamos cuando el estado es (nf, f).