Examen de Computabilidad y Complejidad

(CMC)

14 de junio de 2002

(I) Cuestiones (justifique formalmente las respuestas)

1. Sea el lenguaje $L = \{w_1w_2w_3w_4 : w_1w_3 = a^ib^i, w_2w_4 = c^jd^j, i,j \geq 0\}$. ¿ Es L incontextual ?

(1.5 ptos)

Solución

El lenguaje L no es incontextual. Lo demostraremos mediante el lema de bombeo. Tomemos n como la constante del lema y la cadena $z = a^nc^nb^nd^n$ que pertenece a L y cumple las condiciones de partida del lema.

Factorizamos, de la forma habitual, $z = uvwxy$ y veremos las distintas posibilidades de localización del las subcadenas v y x.

(a) vx formado por símbolos a con $|vx| = k \geq 1$. En este caso tomamos el valor $i = 0$ y formamos la cadena $uv^0wx^0y = a^{n-k}c^nb^nd^n$ que no pertenece a L ya que no hay igual número de símbolos a que de símbolos b.

(b) vx formado por símbolos c con $|vx| = k \geq 1$. En este caso tomamos el valor $i = 2$ y formamos la cadena $uv^2wx^2y = a^nc^{n+k}b^nd^n$ que no pertenece a L ya que no hay igual número de símbolos c que de símbolos d.

(c) vx formado por símbolos b con $|vx| = k \geq 1$. En este caso tomamos el valor $i = 2$ y formamos la cadena $uv^2wx^2y = a^nc^nb^{n+k}d^n$ que no pertenece a L ya que no hay igual número de símbolos b que de símbolos a.

(d) vx formado por símbolos d con $|vx| = k \geq 1$. En este caso tomamos el valor $i = 0$ y formamos la cadena $uv^0wx^0y = a^nc^nb^nd^{n-k}$ que no pertenece a L ya que no hay igual número de símbolos d que de símbolos c.

(e) vx formado por símbolos a y c, con $|vx|_a = k$ y $|vx|_c = j$ donde $k,j \geq 1$. Tomamos el valor $i = 0$ y se forma la cadena $a^{n-k}c^{n-j}b^nd^n$ que no pertenece a L ya que no hay coincidencia en el número de símbolos a y b ni en el de c y d.

(f) vx formado por símbolos c y b, con $|vx|_c = k$ y $|vx|_b = j$ donde $k,j \geq 1$. Tomamos el valor $i = 0$ y se forma la cadena $a^nc^{n-k}b^{n-j}d^n$ que no pertenece a L ya que no hay coincidencia en el número de símbolos a y b ni en el de c y d.

(g) vx formado por símbolos b y d, con $|vx|_b = k$ y $|vx|_d = j$ donde $k,j \geq 1$. Tomamos el valor $i = 0$ y se forma la cadena $a^nc^nb^{n-k}d^{n-j}$ que no pertenece a L ya que no hay coincidencia en el número de símbolos a y b ni en el de c y d.

Debido a las condiciones del lema ya no se pueden plantear más casos y al haber demostrado, en todos los casos posibles, que el lema no se cumple en su tercera condición entonces podemos concluir que L no es incontextual.

2. Sea la operación f definida entre cadenas como sigue: $f(x,y) = sucesor(x)sucesor(y)$, donde $sucesor(z)$ es el sucesor en orden lexicográfico de la cadena z. Se extiende la operación a lenguajes de la forma $f(L_1, L_2) = \{f(x,y) : x \in L_1, y \in L_2\}$.

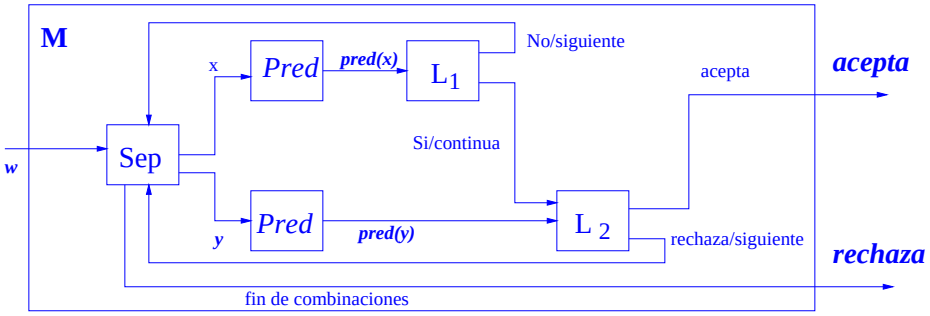
   (a) ¿ Es la operación f de cierre para la clase de los lenguajes recursivos ?

   (b) ¿ Es la operación f de cierre para la clase de los lenguajes recursivamente enumerables ?

(2 ptos)

Solución

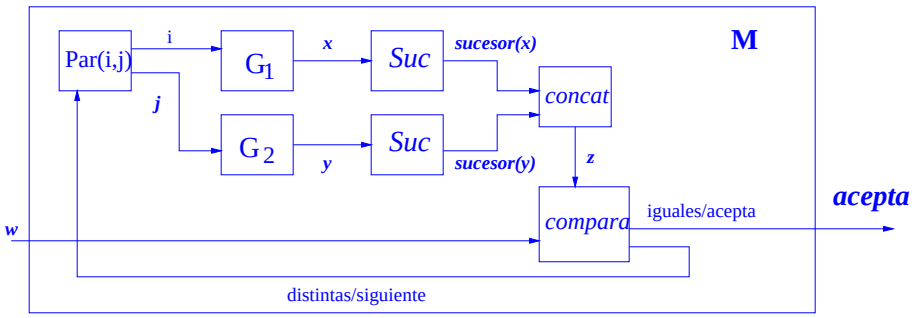
(a) La operación f es de cierre para la clase $\mathcal{L}_{rec}$. Para demostrarlo construiremos una máquina de Turing M que siempre para y que acepta $f(L_1, L_2)$. Para ello, partiremos de las máquinas que siempre paran y aceptan, respectivamente, a L_1 y L_2. El esquema para la máquina M se muestra a continuación



La máquina actúa como se explica a continuación. Tomemos una cadena de entrada w. El módulo *Sep* parte sucesivamente w en dos subcadenas x e y de forma creciente con longitud de x. Inicialmente se toma $|x| = 1$. Se cumple en todo momento que $w = xy$ y que $|y| \geq 1$. El módulo *Pred* calcula el predecesor en orden lexicográfico de la cadena de entrada. Esta subrutina termina en tiempo finito ya que se basa en un algoritmo que se ha visto en varias ocasiones. L_1 y L_2 son las máquinas a las que nos hemos referido anteriormente y que siempre paran. El criterio de aceptación de la cadena w es el siguiente: w debe separarse en dos cadenas x e y de forma que sus predecesores pertenecen respectivamente a L_1 y L_2. Es decir, w es el resultado de concatenar los sucesores de dos cadenas de L_1 y L_2, por lo tanto, w es aceptada si pertenece a $f(L_1, L_2)$. Obsérvese que el criterio de rechazo se establece cuando ya no hay más combinaciones posibles en la cadena w que cumplan las anteriores condiciones, es decir, cuando $w \notin f(L_1, L_2)$.

Al ser M una máquina que garantiza la parada podemos concluir que $f(L_1, L_2)$ es recursivo y que f es de cierre para la clase $\mathcal{L}_{rec}$.

(b) La operación f también es de cierre para la clase $\mathcal{L}_{re}$. Para demostrarlo, construiremos una máquina de Turing M que acepte las cadenas de $f(L_1, L_2)$. Contaremos con dos máquinas generadoras para L_1 y L_2 que denotaremos respectivamente por G_1 y G_2. De igual forma, utilizaremos el generador de pares visto en clase que denotaremos por $Par(i,j)$. A continuación se muestra el esquema de la máquina M



La máquina M funciona como se explica a continuación: Inicialmente se genera un par de enteros i y j que indican las cadenas que se tienen que generar de los lenguajes L_1 y L_2. Es decir, se genera en el módulo G_1 la i-ésima cadena de L_1, que hemos denotado por x, y se genera en el módulo G_2 la j-ésima cadena de L_2, que hemos denotado por y. El módulo Suc calcula el sucesor lexicográfico de su cadena de entrada, el módulo *concat* realiza el producto de dos cadenas de entrada (el orden es el de la cadena que proviene de G_1 con la cadena que proviene de G_2). Por último, el módulo *compara* realiza la comparación de dos cadenas y distingue si son iguales o no. Obsérvese, que la máquina especificada sólo acepta las cadenas de entrada w que pertenecen a $f(L_1, L_2)$. Es decir, aquellas cadenas que son el producto de los sucesores de dos cadenas de L_1 y L_2 respectivamente.

Obsérvese que, en el caso de que L_1 y L_2 sean finitos, entonces $f(L_1, L_2)$ también es finito y, por lo tanto, recursivamente enumerable. Si sólo uno de los dos lenguajes fuera finito, entonces el anterior esquema habría que modificarlo de forma que cada cadena generada del lenguaje infinito se combina con todas las cadenas del lenguaje finito. En cualquier caso, el esquema sería similar al anterior y llegaríamos de nuevo a la conclusión de que la operación f es de cierre para la clase $\mathcal{L}_{re}$.

3. Sean L_1 y L_2 dos lenguajes definidos sobre el mismo alfabeto de forma que $L_1 \cup L_2$, $L_1 - L_2$ y $L_2 - L_1$ son lenguajes recursivos. ¿ Es recursivo el lenguaje L_1 ? ¿ y el lenguaje L_2 ?

(1.5 ptos)

Solución

Veamos, en primer lugar, que L_1 es recursivo. Mediante propiedades de conjuntos podemos expresar $L_1 = (L_1 \cup L_2) - (L_2 - L_1)$ que, aplicando la diferencia, queda como $L_1 = (L_1 \cup L_2) \cap \overline{(L_2 - L_1)}$. Sabemos que $L_2 - L_1$ es recursivo, luego $\overline{(L_2 - L_1)}$ también lo es. Por otra parte, la clase de los lenguajes recursivos es cerrada bajo intersección, luego $L_1 = (L_1 \cup L_2) \cap \overline{(L_2 - L_1)}$ es recursivo, tal y como se quería demostrar.

El razonamiento para ver que L_2 es recursivo es similar al anterior. Mediante propiedades de conjuntos podemos expresar $L_2 = (L_1 \cup L_2) - (L_1 - L_2)$ que, aplicando la diferencia, queda como $L_2 = (L_1 \cup L_2) \cap \overline{(L_1 - L_2)}$. Sabemos que $L_1 - L_2$ es recursivo, luego $\overline{(L_1 - L_2)}$ también lo es. Por otra parte, la clase de los lenguajes recursivos es cerrada bajo intersección, luego $L_2 = (L_1 \cup L_2) \cap \overline{(L_1 - L_2)}$ es recursivo, tal y como se quería demostrar.

(II) PROBLEMAS:

4. Dada una gramática incontextual, diremos que una producción es *recursiva* si el símbolo auxiliar de la parte izquierda de la producción también aparece en su parte derecha. Se pide escribir un módulo *Mathematica* que, dada una gramática incontextual como parámetro de entrada, devuelva como salida el símbolo auxiliar que más producciones recursivas presenta en la gramática y el número de las citadas reglas para dicho auxiliar. Si hay más de un símbolo que devuelva cualquiera de ellos.

(2 ptos)

Solución

```
Solucion[G_List]:=Module[{ P,k, prod, j, cont, auxcont, sol },
    P=G[[3]];
    cont=0;
    For[k=1, k≤Length[P], k++,
        prod=P[[k]];
        auxcont=0;
        For[j=1, j≤Length[prod[[2]]], j++,
            If[MemberQ[prod[[2,j]],prod[[1]]], auxcont++]
        ];
        If[auxcont ≥ cont, sol={prod[[1]],auxcont}; cont=auxcont]
    ];
    Return[sol]
]
```

5. Sea el lenguaje $L = \{a^n b^n c^m : n, m \geq 0\}$. Sea la sustitución f definida como $f(a) = L^*$, $f(b) = L^r L$ y $f(c) = \{\lambda\}$. Se pide obtener una gramática incontextual para el lenguaje $f(L)$.

(1 pto)

Solución

En primer lugar obtendremos una gramática G de forma que $L(G) = L$. La gramática queda definida por las siguientes producciones

$S \to AB$

$A \to aAb \mid \lambda$

$B \to cB \mid \lambda$

A partir de ahora trabajaremos con la anterior gramática. Procedemos a obtener gramáticas de sustitución para f.

$f(a) = L^*$ queda definida por la siguiente gramática

$S_a \to S_1 S_a \mid \lambda$

$S_1 \to A_1 B_1$

$A_1 \to aA_1 b \mid \lambda$

$B_1 \to cB_1 \mid \lambda$

$f(b) = L^r L$ queda definida por la siguiente gramática

$S_b \to S_r S_2$

$S_r \to B_r A_r$

$A_r \to bA_r a \mid \lambda$

$B_r \to B_r c \mid \lambda$

$S_2 \to A_2 B_2$

$A_2 \to aA_2b \mid \lambda$

$B_2 \to cB_2 \mid \lambda$

Por último, $f(c) = \{\lambda\}$ se define por la gramática

$S_c \to \lambda$

A continuación, aplicamos la sustitución f sobre la gramática G y obtenemos la gramática que se nos pide

$S \to AB$

$A \to S_a A S_b \mid \lambda$

$B \to S_c B \mid \lambda$

$S_a \to S_1 S_a \mid \lambda$

$S_1 \to A_1 B_1$

$A_1 \to aA_1b \mid \lambda$

$B_1 \to cB_1 \mid \lambda$

$S_b \to S_r S_2$

$S_r \to B_r A_r$

$A_r \to bA_r a \mid \lambda$

$B_r \to B_r c \mid \lambda$

$S_2 \to A_2 B_2$

$A_2 \to aA_2b \mid \lambda$

$B_2 \to cB_2 \mid \lambda$

$S_c \to \lambda$

6. Dada la gramática G definida por las siguientes producciones se pide obtener una gramática simplificada y en Forma Normal de Chomsky que genere $L(G) - \{\lambda\}$

$S \to ASA \mid B \mid ABC \mid AFA \mid H$ $\quad C \to CC \mid AFB \mid AFD \mid ABC$ $\quad H \to ASA \mid aAb \mid \lambda$

$A \to aAb \mid B \mid b \mid FA \mid A$ $\quad D \to AA \mid BB \mid a$

$B \to DBC \mid H$ $\quad F \to AC \mid FA \mid BBC$

(2 ptos)

Solución

En primer lugar procedemos a simplificar la gramática.

Eliminación de símbolos no generativos

Símbolos no generativos: $\{C, F\}$

Gramática sin símbolos no generativos

$S \to ASA \mid B \mid H$ $\quad H \to ASA \mid aAb \mid \lambda$

$A \to aAb \mid B \mid b \mid A$ $\quad D \to AA \mid BB \mid a$

$B \to H$

Eliminación de símbolos no alcanzables

Símbolos no alcanzables: $\{D\}$

Gramática sin símbolos no alcanzables

$S \to ASA \mid B \mid H$ $\quad H \to ASA \mid aAb \mid \lambda$

$A \to aAb \mid B \mid b \mid A$ $\quad B \to H$

Eliminación de producciones vacías

Símbolos anulables: $\{S, A, H, B\}$

Gramática sin producciones vacías

$S \to ASA \mid SA \mid AA \mid AS \mid A \mid S \mid B \mid H$ $\quad H \to ASA \mid SA \mid AA \mid AS \mid A \mid S \mid aAb \mid ab$

$A \to aAb \mid ab \mid B \mid b \mid A$ $\quad B \to H$

Eliminación de producciones unitarias

$\mathcal{C}(S) = \{S, A, H, B\}$ $\mathcal{C}(A) = \{S, A, H, B\}$ $\mathcal{C}(H) = \{S, A, H, B\}$ $\mathcal{C}(B) = \{S, A, H, B\}$

Gramática sin producciones unitarias

$S \to ASA \mid SA \mid AA \mid AS \mid aAb \mid ab \mid b$

$H \to ASA \mid SA \mid AA \mid AS \mid aAb \mid ab \mid B$

$A \to ASA \mid SA \mid AA \mid AS \mid aAb \mid ab \mid b$

$B \to ASA \mid SA \mid AA \mid AS \mid aAb \mid ab \mid b$

Eliminación de símbolos no alcanzables

Símbolos no alcanzables: $\{H, B\}$

Gramática sin símbolos no alcanzables

$S \to ASA \mid SA \mid AA \mid AS \mid aAb \mid ab \mid b$

$A \to ASA \mid SA \mid AA \mid AS \mid aAb \mid ab \mid b$

Paso a Forma Normal de Chomsky

Sustitución de símbolos terminales

$S \to ASA \mid SA \mid AA \mid AS \mid C_aAC_b \mid C_aC_b \mid b$

$A \to ASA \mid SA \mid AA \mid AS \mid C_aAC_b \mid C_aC_b \mid b$

$C_a \to a$

$C_b \to b$

Factorización de las producciones y obtención de la gramática definitiva en FNC

$S \to AD_1 \mid SA \mid AA \mid AS \mid C_aD_2 \mid C_aC_b \mid b$

$A \to AD_1 \mid SA \mid AA \mid AS \mid C_aD_2 \mid C_aC_b \mid b$

$D_1 \to SA$

$D_2 \to AC_b$

$C_a \to a$

$C_b \to b$