Examen de Computabilidad y Complejidad

(CMC)

11 de junio de 2003

(I) Cuestiones (justifique formalmente las respuestas)

1. Sea el lenguaje $L = \{x\#y : x, y \in \{0,1\}^*, |x| = |y| \wedge |x|_0 = |y|_0\}$. ¿ Es L incontextual ?

(1.5 ptos)

Solución

El lenguaje L no es incontextual. Haremos una demostración mediante aplicación del lema de bombeo. Sea n la constante del lema y tomemos la cadena del lenguaje $z = 0^n1^n\#0^n1^n = uvwxy$. A continuación, procedemos a realizar un estudio por casos de las posibles localizaciones de las subcadenas v y x que cumplan las dos primeras condiciones del lema. Para facilitar el análisis denominaremos primer bloque a la subcadena anterior al símbolo # y segundo bloque a la subcadena posterior al símbolo #.

(a) v y x formadas por 0's del primer bloque
Suponiendo $|vx| = j \geq 1$, tomamos el valor $i = 0$ y obtenemos la nueva cadena $uv^0wx^0y = uwy = 0^{n-j}1^n\#0^n1^n$ que no pertenece al lenguaje ya que el primer bloque no coincide ni en longitud ni en número de símbolos 0 con el segundo bloque.

(b) v y x formadas por 1's del primer bloque
Suponiendo $|vx| = j \geq 1$, tomamos el valor $i = 0$ y obtenemos la nueva cadena $uv^0wx^0y = uwy = 0^n1^{n-j}\#0^n1^n$ que no pertenece al lenguaje ya que el primer bloque no coincide en longitud con el segundo bloque.

(c) v y x formadas por 0's del segundo bloque
Suponiendo $|vx| = j \geq 1$, tomamos el valor $i = 0$ y obtenemos la nueva cadena $uv^0wx^0y = uwy = 0^n1^n\#0^{n-j}1^n$ que no pertenece al lenguaje ya que el primer bloque no coincide ni en longitud ni en número de símbolos 0 con el segundo bloque.

(d) v y x formadas por 1's del segundo bloque
Suponiendo $|vx| = j \geq 1$, tomamos el valor $i = 0$ y obtenemos la nueva cadena $uv^0wx^0y = uwy = 0^n1^n\#0^n1^{n-j}$ que no pertenece al lenguaje ya que el primer bloque no coincide en longitud con el segundo bloque.

(e) v y x formadas por 0's y 1's del primer bloque
Supongamos $|vx|_0 = j \geq 1$ y $|vx|_1 = k \geq 1$. Si tomamos un valor $i = 0$, obtenemos la nueva cadena $uv^0wx^0y = uwy = 0^{n-j}1^{n-k}\#0^n1^n$ que no pertenece al lenguaje ya que el primer bloque no coincide ni en longitud ni en número de símbolos 0 con el segundo bloque.

(f) v y x formadas por 0's y 1's del segundo bloque
Supongamos $|vx|_0 = j \geq 1$ y $|vx|_1 = k \geq 1$. Si tomamos un valor $i = 0$, obtenemos la nueva cadena $uv^0wx^0y = uwy = 0^n1^n\#0^{n-j}1^{n-k}$ que no pertenece al lenguaje ya que el primer bloque no coincide ni en longitud ni en número de símbolos 0 con el segundo bloque.

(g) v y x formadas por 1's del primer bloque y 0's del segundo bloque
Supongamos $|vx|_1 = j \geq 1$ y $|vx|_0 = k \geq 1$. Si tomamos un valor $i = 0$, obtenemos la nueva cadena $uv^0wx^0y = uwy = 0^n1^{n-j}\#0^{n-k}1^n$ que no pertenece al lenguaje ya que el primer bloque no coincide en número de símbolos 0 con el segundo bloque.

(h) v y x que contengan el símbolo #
Tomando el valor de $i = 0$ formamos la cadena $uv^0wx^0y = uwy$ que no pertenece al lenguaje al haberse eliminado el único símbolo # que la cadena contenía.

Puesto que en todos los casos de análisis posibles hemos obtenido contradicciones al lema, podemos concluir que el lenguaje L no es incontextual.

2. Sea la operación P definida sobre cadenas del alfabeto $\{a, b\}$ como sigue: si la cadena termina en el símbolo a la operación cambia todos los símbolos a por símbolos b, si la cadena termina en el símbolo b la operación elimina todos los símbolos b de la cadena. ¿ Es la clase de los lenguajes incontextuales cerrada bajo P ?

(1.5 ptos)

Solución

La clase de los lenguajes incontextuales es cerrada bajo P. Para justificar la respuesta expresaremos P como el resultado de aplicar operaciones de cierre para la clase de los lenguajes incontextuales.

En primer lugar, el cambio de los símbolos a por símbolos b de una cadena se puede formalizar por el homomorfismo $h(a) = b$ y $h(b) = b$. La eliminación de los símbolos b de una cadena la podemos formalizar por el homomorfismo $g(a) = a$ y $g(b) = \lambda$. La selección de las cadenas de un lenguaje L que terminan por el símbolo a se expresa mediante la intersección $L \cap (a+b)^*a$. De igual forma, la selección de las cadenas de un lenguaje L que terminan por b se expresa como $L \cap (a+b)^*b$. Finalmente, seleccionar la cadena λ de un lenguaje L se expresa como $L \cap \lambda$. En definitiva, la operación P, actuando sobre un lenguaje L, la podemos expresar como sigue

$$P(L) = h(L \cap (a+b)^*a) \cup g(L \cap (a+b)^*b) \cup (L \cap \lambda)$$

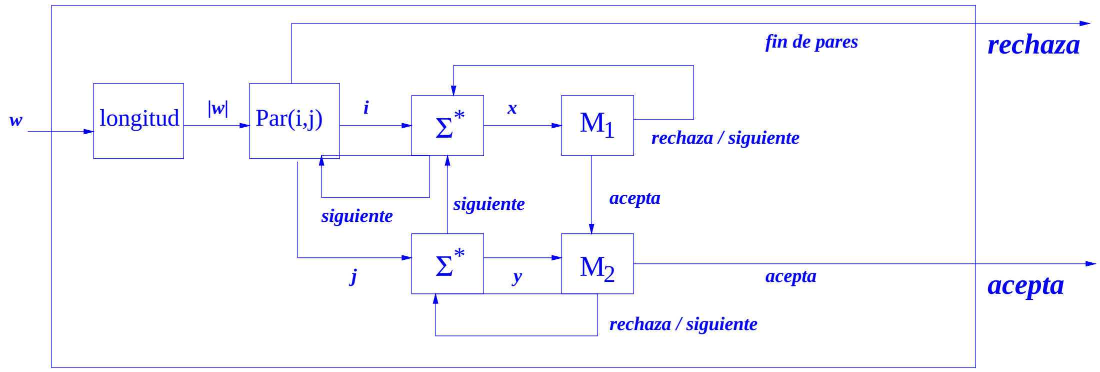
Obsérvese que todas las operaciones que hemos aplicado son de cierre para la clase de los lenguajes incontextuales, por lo que P también lo es.

3. Dados dos lenguajes L_1 y L_2, definidos sobre el alfabeto $\{a, b\}$, definimos la operación P como $P(L_1, L_2) = \{w \in \{a, b\}^* : |w| = |x| + |y|, x \in L_1, y \in L_2\}$. ¿ Es la clase de los lenguajes recursivos cerrada bajo P ?

(2 ptos)

Solución

La clase de los lenguajes recursivos es cerrada bajo P. Propondremos un esquema de aceptación para el lenguaje $P(L_1, L_2)$ que garantice siempre la parada. Para ello, contaremos con un módulo M_1 que acepta L_1, contaremos también con un módulo M_2 que acepta L_2. Obsérvese que, tanto M_1 como M_2 garantizan siempre la parada al ser L_1 y L_2 lenguajes recursivos. También contaremos con un generador de pares, $Par(i, j)$, que se ha utilizado en numerosas ocasiones y cuyo mecanismo interno se ha explicado en clase. En este caso, el generador de pares toma una entrada numérica y únicamente genera aquellos pares cuya suma sea igual a la entrada. Contaremos con un módulo que calcula la longitud de una cadena de entrada, *longitud*, que se ha utilizado en diversas ocasiones. También contaremos con un módulo Σ^* que, tomando como entrada un valor entero positivo, genera las cadenas definidas sobre $\Sigma = \{a, b\}$ cuya longitud sea igual al valor de entrada. Este módulo se fundamenta en que el conjunto generado es finito y, por lo tanto, recursivo, por lo que existe una máquina de Turing que lo puede generar en orden lexicográfico. El esquema que se propone es el siguiente



El funcionamiento del anterior esquema se explica a continuación. Inicialmente, dada una cadena de entrada w se calcula su longitud y se activa el generador de pares que emite el par $< i, j >$ que cumple $i + j = |w|$. El valor i se proporciona como entrada al módulo Σ^* que proporciona como salida una cadena x de longitud i. La cadena x se pasa como entrada al módulo M_1 que establece si pertenece a L_1. En caso negativo, se solicita otra cadena de longitud i al módulo Σ^*. Si $x \in L_1$, entonces a partir del generador de pares y del módulo Σ^* se genera una cadena y de longitud j. La cadena y se somete como entrada al módulo M_2 que establece si pertenece a L_2. En caso afirmativo se acepta la cadena de entrada w y finaliza el proceso. En caso negativo, se solicita una nueva cadena de longitud j y se repite el anterior proceso. Cuando ya se han generado todas las cadenas y de longitud j sin éxito en la aceptación en M_2 se solicita una nueva cadena x de longitud i y se repite el proceso desde el principio (esto es, estableciendo en primer lugar si $x \in L_1$). Si se han generado todas las cadenas de longitud i sin éxito en el proceso anterior, se solicita un nuevo par $< i, j >$ y se repite el proceso completo (tomando cadenas x e y). Si se han generado todas los posibles pares cuya suma sea $|w|$, y no se ha tenido ningún éxito en el proceso descrito anteriormente, se rechaza la cadena de entrada.

Obsérvese que el anterior esquema garantiza la parada por la finitud de los procesos de generación de pares $< i, j >$ y de las cadenas en Σ^* y por la condición de parada en M_1 y M_2. Por otra parte, el anterior esquema acepta exactamente $P(L_1, L_2)$ ya que sólo son aceptadas aquellas cadenas de entrada para las que existe un par de cadenas de L_1 y L_2 cuya suma de longitudes iguale a la longitud de la cadena de entrada. En conclusión, podemos afirmar que P es una operación de cierre para la clase de los lenguajes recursivos.

(II) PROBLEMAS:

4. Se pide construir un módulo *Mathematica* que, tomando como parámetro de entrada una gramática incontextual y un valor entero n, devuelva como salida una lista con todos aquellos símbolos auxiliares que aparecen más de n veces en total contando las partes derechas de las reglas.

(2 ptos)

Solución

```
Solucion[G_List, n_Integer]:=Module[{ P,Salida,aux,k,contador,j,i,m },
    P=G[[3]];
    Salida={};
    aux=G[[1]];
    For[k=1, k≤Length[aux], k++,
        contador=0;
        For[ j=1, j≤Length[P], j++,
            regla=P[[j]];
            For[i=1, i≤Length[P[[j,2]]], i++,
                For[m=1, m≤Length[P[[j,2,i]]], m++,
                    If[P[[j,2,i,m]] == aux[[k]], contador++]
                ];
```

```
            ];
        ];
        If[contador>n, AppendTo[Salida,aux[[k]]]];
    ];
    Return[Salida]
]
```

5. Sea G la gramática con las reglas $S \to AA1 \mid 0S$; $A \to 1S0 \mid \lambda$ y sea h el homomorfismo definido como $h(0) = ab$ y $h(1) = \lambda$. Obtenga una gramática incontextual para el lenguaje $h(L(G) \cup (L(G))^r)$

(1 pto)

Solución

En primer lugar, una gramática para $L(G)^r$

$S_r \to 1A_rA_r \mid S_r0$

$A_r \to 0S_r1 \mid \lambda$

A continuación, una gramática para $L(G) \cup (L(G))^r$

$S_1 \to S \mid S_r$

$S \to AA1 \mid 0S$

$A \to 1S0 \mid \lambda$

$S_r \to 1A_rA_r \mid S_r0$

$A_r \to 0S_r1 \mid \lambda$

Por último, la gramática para $h(L(G) \cup (L(G))^r)$ que se solicitaba en el enunciado

$S_1 \to S \mid S_r$

$S \to AA \mid abS$

$A \to Sab \mid \lambda$

$S_r \to A_rA_r \mid S_rab$

$A_r \to abS_r \mid \lambda$

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

$S \to AB01 \mid 0B11 \mid 0$
$A \to 00A \mid AS$
$B \to BSB \mid AC \mid \lambda$
$C \to SAS \mid 01 \mid \lambda$

(2 ptos)

Solución

En primer lugar procedemos a simplificar la gramática.

Eliminación de símbolos no generativos

Símbolos no generativos: $\{A\}$

Gramática sin símbolos no generativos

$S \to 0B11 \mid 0$

$B \to BSB \mid \lambda$

$C \to 01 \mid \lambda$

Eliminación de símbolos no alcanzables

Símbolos no alcanzables: $\{C\}$

Gramática sin símbolos no alcanzables

$S \to 0B11 \mid 0$

$B \to BSB \mid \lambda$

Eliminación de producciones vacías

Símbolos anulables: $\{B\}$

Gramática sin producciones vacías

$S \to 0B11 \mid 011 \mid 0$

$B \to BSB \mid BS \mid SB \mid S$

Eliminación de producciones unitarias

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

Gramática sin producciones unitarias

$S \to 0B11 \mid 011 \mid 0$

$B \to BSB \mid BS \mid SB \mid 0B11 \mid 011 \mid 0$

La gramática anterior ya está totalmente simplificada ya que todos sus símbolos son útiles (generativos y alcanzables).

Paso a Forma Normal de Chomsky

Sustitución de símbolos terminales

$S \to C_0BC_1C_1 \mid C_0C_1C_1 \mid 0$

$B \to BSB \mid BS \mid SB \mid C_0BC_1C_1 \mid C_0C_1C_1 \mid 0$

$C_0 \to 0$

$C_1 \to 1$

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

$S \to C_0D_1 \mid C_0D_2 \mid 0$

$D_1 \to BD_2$

$D_2 \to C_1C_1$

$B \to BD_3 \mid BS \mid SB \mid C_0D_1 \mid C_0D_2 \mid 0$

$D_3 \to SB$

$C_0 \to 0$

$C_1 \to 1$