Modelos de Computación

grado en ingeniería informática

Preguntas de examen resueltas

Autor
Carlos Sánchez Páez

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Curso 2019-2020

Índice

Índice de figuras

1. Tema 1

1. Determinar si la gramática $G = (S, A, B, a, b, c, d, P, S)$ donde P es el conjunto de reglas de producción:

$$S \implies AB; A \implies Ab; A \implies a; B \implies cB; B \implies d$$

genera un lenguaje de tipo 3.

Solución

Comenzamos a generar:

$$S \implies AB \underset{A \implies Ab}{\Longrightarrow} AbB \underset{A \implies Ab}{\Longrightarrow} AbbB \underset{\dots}{\Rightarrow} Ab^iB \underset{B \implies cB}{\Longrightarrow} Ab^icB$$
$$\underset{\dots}{\Rightarrow} Ab^ic^jB \underset{A \implies a}{\Longrightarrow} ab^ic^jB \underset{B \implies d}{\Longrightarrow} ab^ic^jd$$

Vemos que generamos el lenguaje ab^ic^jd, que también se puede generar mediante la siguiente gramática:

$$S \implies aB; B \implies bB; B \implies C; C \implies cC; C \implies d$$

Como la gramática es de tipo 3 (sólo hay como máximo una variable a la derecha en todas las producciones), el lenguaje también lo es.

2. Diseñar una máquina de estados que calcule el complemento a dos de un número binario.

Solución

El complemento a dos de un número binario se calcula obteniendo su complemento a uno y sumándole uno. Veamos algunos ejemplos:

- $C_2(1100)$=$C_1(1100)$+1=0011+1=0100
- $C_2(1110)$=$C_1(1110)$+1=0001+1=0010
- $C_2(11101100)$=$C_1(11101100)$+1=00010011+1=00010100

Tras realizar varias operaciones nos damos cuenta de que existe una codificación que se mantiene:

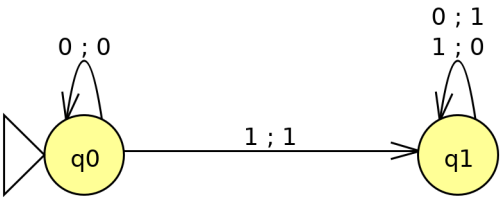
a) Comenzamos leyendo el número de derecha a izquierda y escribimos lo que leemos en la salida (también de derecha a izquierda).

b) Cuando encontremos el primer 1 lo escribimos en la cinta y a partir de ahí escribimos el complemento a uno del número que leamos (cambiamos 0 por 1 y viceversa).

Esta codificación se puede expresar mediante una máquina de *Mealy*:

- La cabeza lectora y escritora se desplazará de derecha a izquierda.
- Estados
  - q_0: todavía no he leído el primer 1. Si leo 0, escribo 0 y me mantengo. Si leo 1, paso al estado q_1 y escribo 1.
  - q_1: ya he leído el primer 1. Ahora debo aplicar el complemento a 1 (si leo 1 escribo 0 y viceversa). En ambos casos me mantengo.

  Es decir, la máquina sería la siguiente:



3. Si es posible, construir un autómata finito para el siguiente lenguaje:

$$L = \{pcp^{-1} : p \in \{0,1\}^*\}$$

En caso contrario, demostrar que no existe dicho autómata finito.

Solución

Para resolver este problema emplearemos el lema de bombeo. De un primer vistazo podemos ver que el lenguaje no es regular, ya que la relación sintáctica establecida (capicúa) es bastante fuerte.

- Suponemos que el lenguaje es regular, por lo que debe satisfacer el lema de bombeo.
- Comienzo eligiendo una cadena $z \in L$, por ejemplo, $z = 0^n c 0^n$. El lema de bombeo dice que $\exists n \in \mathbb{N} : z = uvw$.
- Comienzo a determinar el valor de u, v y w. Como la primera condición nos dice que $|uv| \leq n$, uv debe ser una parte de 0^n (si fuera más no se cumpliría la condición). En este punto podemos determinar que $\boldsymbol{w = 0^{n-k}c0^n}$ tal que $k < n \in \mathbb{N}$.
- La segunda condición nos dice que $\boldsymbol{|v| \geq 1}$, por lo que $\boldsymbol{v = 0^l : l \geq 1}$ y $\boldsymbol{u = 0^{k-l}}$. Por ahora tenemos que $\boldsymbol{z = 0^n c 0^n = uvw = 0^{k-l}0^l 0^{n-k} c 0^n}$.
- Ahora intentaremos buscar la contradicción bombeando, de forma que se incumpla que $\forall i \in \mathbb{N} : uv^i w \in L$. Para $i = 0$:

$$uv^0 w = 0^{k-l}0^{n-k}c0^n = \boldsymbol{0^{n-l}c0^n} \notin L$$

(el resultado tiene un número distinto de ceros a cada lado de c).
Por tanto, queda demostrado que no existe un autómata finito para el lenguaje propuesto.