Theory of Automata and Formal languages

Fernando Javier López Cerezo
Taller 3

November 23, 2022

Lema. La función $factorial(n) = n!$ es recursiva y WHILE-calculable.

Recordemos que es la recursión primitiva:

Definición. Recursión primitiva
Sea $k \geq 0$ y las funciones
$g : \mathbb{N}^k \to \mathbb{N}$
$h : \mathbb{N}^{k+2} \to \mathbb{N}$
Si la función $f : \mathbb{N}^{k+1} \to \mathbb{N}$ es

$$f(\vec{n}, m) = \begin{cases} g(\vec{n}) & \text{si} \quad m = 0 \\ h(\vec{n}, m-1, f(\vec{n}, m-1)) & \text{si} \quad m > 0 \end{cases}$$

entonces f se obtiene de g y h por recursión primitiva.

Lo expresaremos como $f(\vec{n}) = \langle g|h\rangle(\vec{n})$, o simplemente $f = \langle g|h\rangle$.

Demostración. Veamos primero que es una función recursiva. Para ello demostraremos que $factorial(n)$ se puede expresar como recursión primitiva de otras dos funciones. Como $factorial(n) : \mathbb{N} \to \mathbb{N}$ siguiendo la definición anterior tenemos que tomar $k = 0$. Luego hemos de encontrar una función $g : \mathbb{N}^0 \to \mathbb{N}$ y una función $h : \mathbb{N}^2 \to \mathbb{N}$ tales que $factorial = \langle g|h\rangle$. Una forma elemental de expresar la función $factorial$ de forma recursiva es

$$factorial(m) = \begin{cases} 1 & \text{si} \quad m = 0 \\ m factorial(m-1) & \text{si} \quad m > 0 \end{cases}$$

Siguiendo la definición es claro que la función g debe ser la función constante $g() = 1$ y que $h(m, n) = (m+1)n$ ya que así $h(m-1, factorial(m-1)) = mfactorial(m-1)$. Usando la notación usada en los apuntes, $g = \sigma(\theta)$ y $h = \langle\langle\theta|\pi_2^2\rangle|\langle\pi_1^1|\sigma(\pi_3^3)\rangle(\sigma(\pi_1^3), \pi_3^3)\rangle$. Veáse que h no es más que la función producto con una pequeña modificación ($\sigma(\pi_3^3)$ en vez de π_3^3) para que a la hora de multiplicar dos números se sume 1 al primer número antes.

Luego finalmente $factorial$ es una función recursiva ya que se puede expresar de la siguiente manera: $factorial = \langle\sigma(\theta)|\langle\langle\theta|\pi_2^2\rangle|\langle\pi_1^1|\sigma(\pi_3^3)\rangle(\sigma(\pi_1^3), \pi_3^3)\rangle\rangle$.

Veamos ahora que $factorial$ es WHILE-calculable. Para ello basta que esta función se puede expresar como un programa WHILE (en este caso usaremos WHILE extendido):

```
Factorial=(1,s)
s:
   X2 := X1 - 1;
   while X2 ≠ 0 do
        X1 := Product(X1,X2);
        X2 := X2 - 1;
   od


Product=(2,s)
s:
   X3 := 0
   while X2 ≠ 0 do
      X3 := Sum(X3,X1)
      X2 := X2 - 1;
   od
   X1 := X3


Sum=(2,s)
s:
   while X2 ≠ 0 do
      X1 := X1 + 1
      X2 := X2 - 1;
   od
```

□