Teoría de Autómatas y Lenguajes Formales

Práctica 3: Máquina de Turing, funciones recursivas, lenguajes WHILE

Lázaro, Vargas García

18 de noviembre de 2022

1. Define una máquina de Turing que resuelva el ejercicio 3.4 de la lista de problemas y verifica su correcto funcionamiento

1.1. Máquina de Turing que suma 2 números naturales

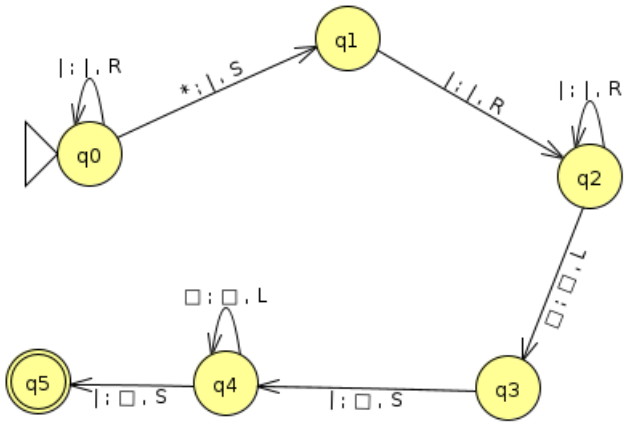
La máquina de Turing que vamos a definir realiza la operación suma de 2 números.

Los datos introducidos en la máquina se introducen de la siguiente manera:

Sean $n, m \in \{0, 1, 2, 3, 4....\}$, entonces la secuencia de entrada constará de $(n+1)$ y $(m+1)$ caracteres $'|'$ separados por un carácter '*'. (Nótese que $0 :=' |'$)

Por ejemplo, $1 + 4 := || * |||||$ y la salida debe ser $||||||$

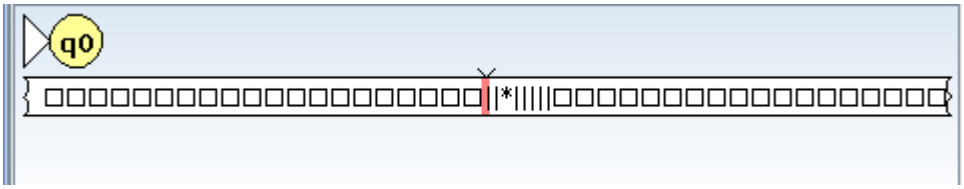
La máquina de Turing en JFLAP queda de la siguiente forma



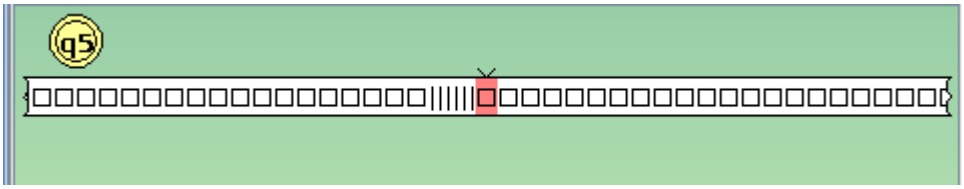
Observemos que asumimos que la cabeza lectora se sitúa inicialmente en el primer caracter $'|'$ empezando por la izquierda.

1.2. Verificación del correcto funcionamiento de la máquina

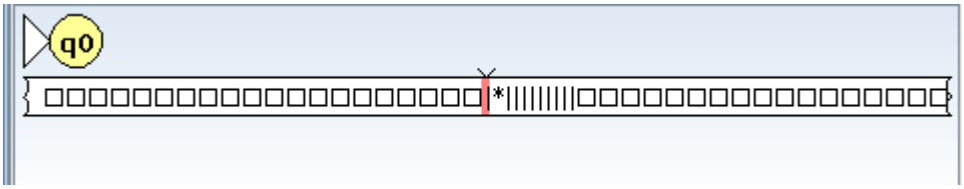
Si introducimos la operación $1 + 4$, la cadena que debe introducirse es:



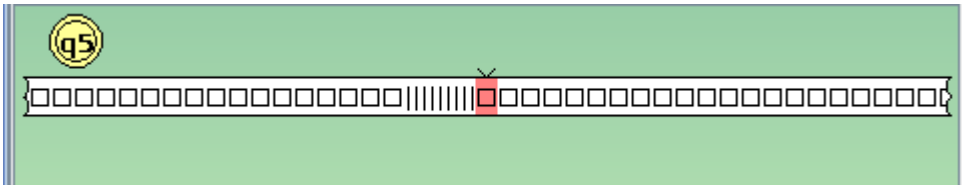
Y se recibe esta salida, que se corresponde con un 5, ya que es una cadena de 6 caracteres $'|'$:



Introduciendo la operación $0 + 8$, la cadena que debe recibirse en la máquina es:



Y la cadena resultante está formada por 9 caracteres $'|'$, que se corresponde con un 8:



2. Define una función recursiva para la suma de 3 valores

2.1. Definición de la función

En teoría hemos visto que la función suma de 2 números se implementa de la siguiente forma:

$suma(n,m) = \langle \pi_1^1 | \sigma(\pi_3^3) \rangle (n,m)$ Es decir, es una recursión primitiva, por lo que se expresará así:

$$suma(n,m) = \begin{cases} \pi_1^1(n) & si \quad m = 0 \\ \sigma(\pi_3^3)(n, m-1, suma(n, m-1)) & si \quad m > 0 \end{cases}$$

Es lógico, ya que el caso base es cuando $m = 0$, en el que el resultado es el propio n por ser 0 el neutro para la suma, y el otro caso consiste simplemente en sumar 1 al resultado de sumar n y m-1.

Para el caso de sumar tres números, hemos de considerar como caso base cuando recibamos un vector de la forma $(n, k, 0)$ donde n y m son naturales, y este caso consistirá simplemente en aplicar $suma(n, k)$ que ya está definida.

Es inmediato ver entonces que para sumar 3 números habrá que hacer una recursión primitiva con caso base la suma de los 2 primeros, y donde la otra opción sea sumar 1 al resultado de sumar los dos primeros con el anterior del último, de este modo:

$$suma3(n,k,m) = \begin{cases} suma(n,k) = \langle \pi_1^1 | \sigma(\pi_3^3) \rangle & si \quad m = 0 \\ \sigma(\pi_4^4)(n, k, m-1, suma3(n, k, m-1)) & si \quad m > 0 \end{cases}$$

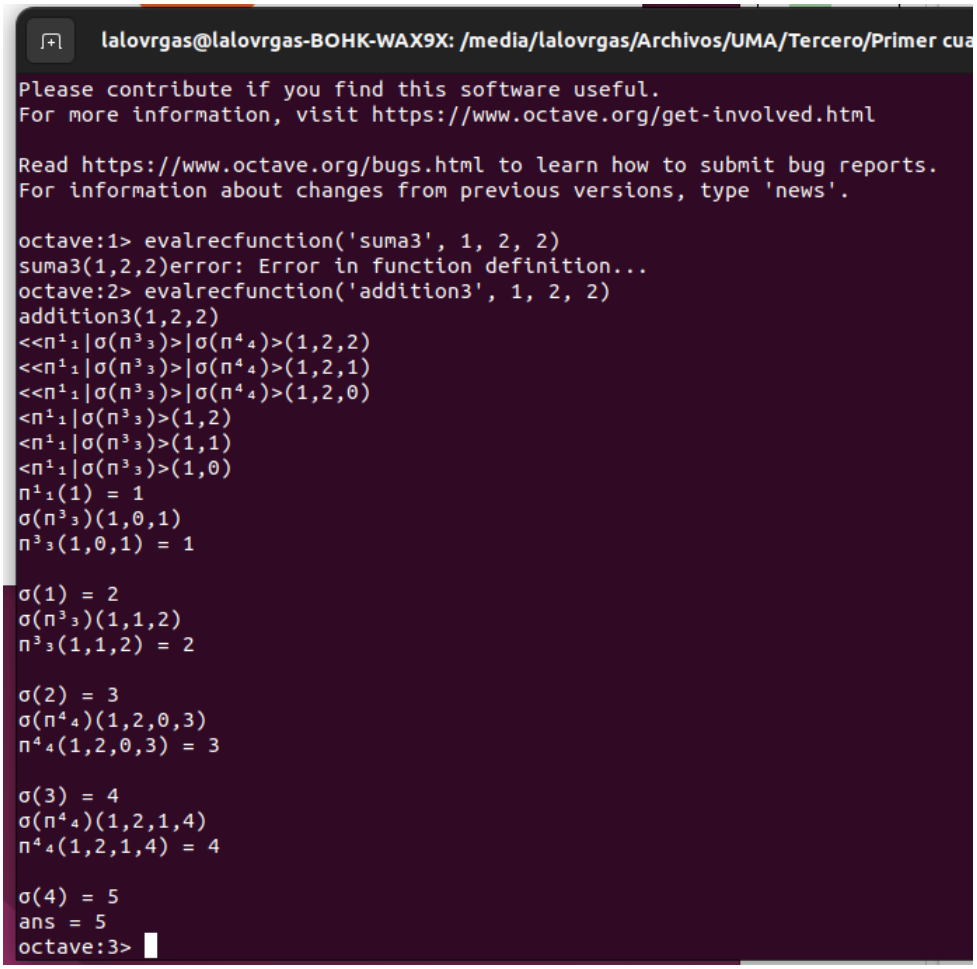
Luego, tenemos lo siguiente: $suma3(n,k,m) = \langle suma(n,k) | \sigma(\pi_4^4) \rangle (n,k,m)$
$= \langle \langle \pi_1^1 | \sigma(\pi_3^3) \rangle | \sigma(\pi_4^4) \rangle (n,k,m)$

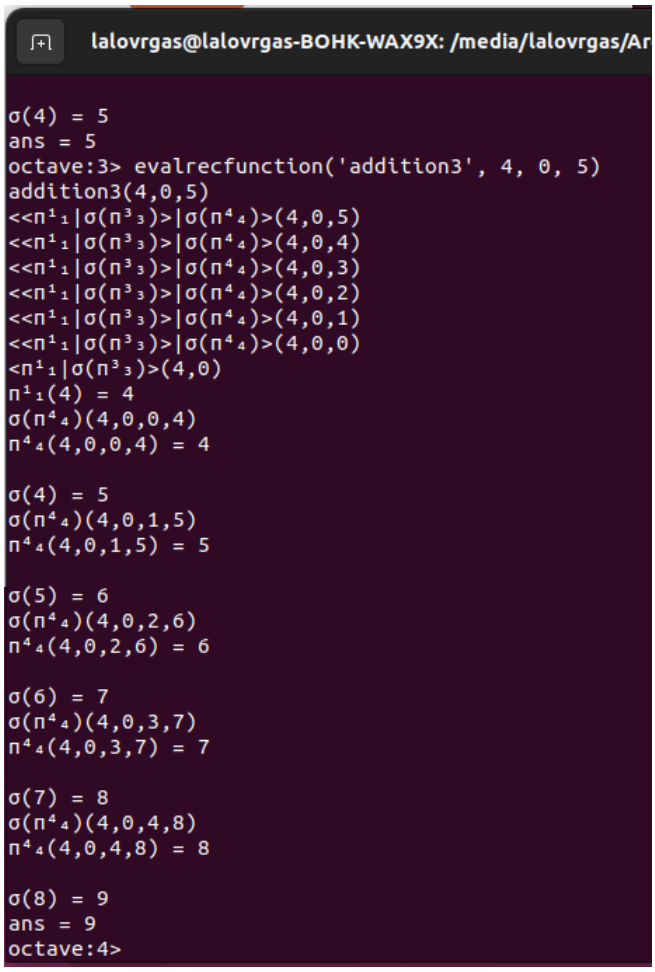
2.2. Pruebas en octave con la función que hemos definido

Para hacer esta prueba, en primer lugar vamos a definir nuestra función en el fichero $recursivefunctions$:

```
12 addition3            <<n^1_1|σ(n^3_3)>|σ(n^4_4)>
```

Usando el script de octave $evalrecfunction.m$, vemos que hemos obtenido los resultados correctos para las datos que hemos introducido:


lalovrgas@lalovrgas-BOHK-WAX9X: /media/lalovrgas/Archivos/UMA/Tercero/Primer cua
Please contribute if you find this software useful.
For more information, visit https://www.octave.org/get-involved.html
Read https://www.octave.org/bugs.html to learn how to submit bug reports.
For information about changes from previous versions, type 'news'.
octave:1> evalrecfunction('suma3', 1, 2, 2)
suma3(1,2,2)error: Error in function definition...
octave:2> evalrecfunction('addition3', 1, 2, 2)
addition3(1,2,2)
<<π¹₁|σ(π³₃)>|σ(π⁴₄)>(1,2,2)
<<π¹₁|σ(π³₃)>|σ(π⁴₄)>(1,2,1)
<<π¹₁|σ(π³₃)>|σ(π⁴₄)>(1,2,0)
<π¹₁|σ(π³₃)>(1,2)
<π¹₁|σ(π³₃)>(1,1)
<π¹₁|σ(π³₃)>(1,0)
π¹₁(1) = 1
σ(π³₃)(1,0,1)
π³₃(1,0,1) = 1
σ(1) = 2
σ(π³₃)(1,1,2)
π³₃(1,1,2) = 2
σ(2) = 3
σ(π⁴₄)(1,2,0,3)
π⁴₄(1,2,0,3) = 3
σ(3) = 4
σ(π⁴₄)(1,2,1,4)
π⁴₄(1,2,1,4) = 4
σ(4) = 5
ans = 5
octave:3>



3. Implementa un programa WHILE que calcule la suma de 3 valores. Debes usar una variable auxiliar que guarde el valor de la suma

Nuestro programa $Suma3$ usará las variables $\{X1, X2, X3, X4\}$, siendo las 3 primeras los números que hay que sumar, y la última, la variable donde se guarda el resultado de la suma.

El código quedaría así:

```
X4 := X1;
while X2 ≠ 0 do
    X4 := X4 + 1;
    X4 := X2 − 1;
od
while X3 ≠ 0 do
    X4 := X4 + 1;
    X4 := X3 − 1;
od
```