Universidad Autónoma

Prácticas de AUTLEN

Práctica 2

Memoria

Autores:
Aitor Arnaiz del Val
Santiago González-Carvajal
Centenera

Pareja 9
Grupo 1401

19 de noviembre de 2018

Índice

1. Descripción de la implementación

En esta práctica hemos tenido que implementar una función de mayor dificultad que las implementadas anteriormente ***cierre_transitivo*** que calcula el cierre transitivo de una matriz de transiciones lambda. Nuestra manera de implementarla la comentaremos a continuación. En cuanto al resto de funciones, comentaremos las que han supuesto una mayor dificultad.

1.1. cierre_transitivo:

En esta función, como hemos comentado anteriormente, calculamos el cierre transitivo de, en nuestro caso, una matriz cuadrada. Para ello, hemos necesitado implementar una función auxiliar encargada de buscar transiciones recursivamente (de acuerdo a la propiedad transitiva). Por lo tanto, la función principal, simplemente busca transiciones lambda en un paso, y a continuación, por cada transición lambda encontrada, llama a la función recursiva que se encarga de buscar relaciones lambda de manera transitiva, es decir, en más de un paso. Por ejemplo, si hay una transición lambda entre el estado 1 y el 3, sería encontrado por la función principal, y a continuación, esta llamaría a la función recursiva para que añada a las transiciones lambda posibles desde 1 todas las que sean posibles desde 3, o desde cualquier estado accesible desde 3 directa o indirectamente.

1.2. cierre_reflexivo:

Con el fin de optimizar nuestro programa, como el cierre reflexivo únicamente consistía en insertar ünos.[en] la diagonal de nuestra matriz (al ser esta cuadrada) no hemos empleado dos bucles anidados para recorrer la matriz e insertar estas transiciones reflexivas, sino que lo que hemos hecho ha sido recorrer con un solo índice el tamaño del lado de la matriz e insertar en matriz[índice][índice] el '1' que indica que existe una transición en esa posición de la matriz, en este caso representando la transición reflexiva.

1.3. is_in_actuales:

Función encargada de comprobar si el estado (dado por su índice) se encuentra entre los estados actuales de autómata.

1.4. AFNDInicializaCadenaActual:

Esta función ya la habíamos implementado para la práctica anterior (en el módulo "palabra"). Por ello, simplemente hemos tenido que llamar a la función ya implementada *reset_word*.

1.5. Cambios en AFNDInicializaEstado:

Esta función ha tenido que pasar a inicializar la lista de estados actuales teniendo en cuenta todas las posibles transiciones lambda.

1.6. Cambios en AFNDProcesaEntrada:

En esta función hemos tenido que cambiar la lógica para adecuarnos a la salida esperada. Esto se debe a que en la salida esperada cuando la lista de estados actuales está vacía (aunque quede palabra por procesar) termina la ejecución.

1.7. Cambios en AFNDTransita:

En esta función hemos tenido que comprobar, además de las transiciones que ya comprobábamos en la práctica anterior, las posibles transciones lambda. Eso sí, evitando que los estados actuales aparecieran repetidos (lo cual podría pasar si no hubiéramos implementado la funcionalidad que nos brinda *is_in_actuales* de una manera u otra), ya que las transiciones lambda son transitivas, como hemos visto anteriormente.