Prueba tecnica

Jose Antonio Rodriguez Rivera

2023-06-23

Explicación del código para encontrar rutas entre elementos jurídicos

Para resolver este ejercicio, desarrollé una función llamada `encontrar_rutas` en Python. Esta función recibe tres parámetros: `elementoJuridicoInicial`, `elementoJuridicoFinal` y `relaciones`. Su objetivo es encontrar todas las rutas posibles entre los elementos jurídicos proporcionados.

Comencé definiendo una lista llamada `rutas`, que almacenará todas las rutas encontradas, y otra lista llamada `ruta_actual`, que representa la ruta que se está explorando en cada momento.

A continuación, creé una función auxiliar llamada `encontrar_rutas_recursivo` que realiza la exploración recursiva para encontrar las rutas. Esta función toma un parámetro `elemento_actual`, que representa el elemento actual en la exploración.

Dentro de `encontrar_rutas_recursivo`, verifiqué si `elemento_actual` es igual a `elementoJuridicoFinal`. Si es así, significa que se ha encontrado una ruta completa desde `elementoJuridicoInicial` hasta `elementoJuridicoFinal`. En este caso, agregué una copia de la `ruta_actual` a la lista de `rutas` utilizando `rutas.append(list(ruta_actual))`.

Si `elemento_actual` no es igual a `elementoJuridicoFinal`, se deben explorar más relaciones para encontrar la ruta completa. Iteré sobre cada relación en la lista `relaciones` utilizando un bucle `for`.

Dentro del bucle, verifiqué si el primer elemento de la relación es igual a `elemento_actual` y si el segundo elemento de la relación no está en la `ruta_actual`. Esto garantiza que no se repitan elementos en la ruta. Si se cumplen ambas condiciones, agregué el segundo elemento de la relación a la `ruta_actual` utilizando `ruta_actual <- c(ruta_actual, relacion[2])` y llamé recursivamente a `encontrar_rutas_recursivo` pasando el segundo elemento de la relación como `elemento_actual`. Esto permitió continuar explorando las relaciones y construyendo la ruta.

Después de la llamada recursiva, utilicé `ruta_actual <- ruta_actual[-length(ruta_actual)]` para retroceder y eliminar el último elemento agregado a la ruta. Esto fue necesario para explorar otras posibles rutas.

Al final de la función `encontrar_rutas_recursivo`, realicé una llamada inicial a la función pasando `elementoJuridicoInicial` como argumento. Esto inició la exploración recursiva desde el elemento inicial.

Finalmente, la función `encontrar_rutas` devolvió la lista de `rutas` como resultado.

En el ejemplo de uso, creé una lista de `relaciones` que contiene las tuplas proporcionadas en el ejercicio. Luego llamé a la función `encontrar_rutas` pasando `'Juez de Amparo'` como `elementoJuridicoInicial` y `'Sentencia 338/2018'` como `elementoJuridicoFinal`. Esto buscó todas las rutas posibles entre estos dos elementos utilizando las relaciones proporcionadas.

Por último, utilicé un bucle `for` para iterar sobre las `rutas` encontradas e imprimir cada ruta.

Este enfoque de búsqueda recursiva de grafos permitió encontrar todas las rutas entre los elementos jurídicos deseados. Fue necesario utilizar la recursividad para explorar todas las posibilidades y construir las rutas paso a paso.