Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 2do. Semestre 2021.

ALGORITMO DE MOVIMIENTO PARA R2E2

202001053 – Kevin Mark Hernández Chicol

Resumen

Se presenta una manera para dar solución para el manejo y creación de archivos XML, mediante el cual se hizo usa de nodos, listas enlazadas y la creación de una matriz para poder efectuar lo requerido con los datos, para ello se utilizó la herramienta para manejo de ficheros xml programado en el lenguaje Python usando el modulo element tree, usando modulos que representan información estructurada en el fichero xml y como se puede obtener la información de dicha estructura, como añadir, eliminar elementos o atributos y a la modificación de la información almacenada.

La aplicación esta desarrollada mediante el paradigma de objetos para almacenar la información y manejarla mediante un TDA, el cual nos servirá para hallar la solución al problema mediante la programación, se utilizaron de muchas herramientas para encontrar la solución para la app

Palabras clave

- Matriz
- Nodo
- Lista Enlazada
- XML
- Graphviz

Abstract

A way to provide a solution for the management and creation of XML files is presented, through which the use of nodes, linked lists and the creation of a matrix to be able to carry out what is required with the data was used, for this the management tool was used of xml files programmed in the Python language using the tree of module elements, using modules that represent structured information in the xml file and how the information of said structure can be obtained, such as adding, removing elements or attributes and modifying the stored information.

The application is developed through the object paradigm to store the information and handle it an ADT, which will help us to find the solution to the problem through programming, many tools were used to find the solution for the app

Keywords

- Matrix
- Node
- Linked List
- XML
- Graphviz.

Introducción

A continuación, se presenta la solución para el movimiento del robot R2E2, el cual se utilizará el lenguaje de programación Python y el paradigma de programación orientado a objetos, para el manejo de memoria se utilizará de los TDA's.

El objetivo del manejo de la memoria se realizo mediante a las 2 herramientas mencionadas anteriormente y así poder manejarla en diferentes clases y de esta manera tener toda la estructura ordenada para un manejo más eficiente de nuestras secuencias de código.

En este proyecto se utilizaron varias clases y se necesito de su respectiva importación para el manejo de estas en otras clases

Desarrollo del tema

El problema que se nos plantea es sobre el desarrollo de un algoritmo para que r2e2 pueda identificar el camino con menor consumo de combustible para moverse por el terreno de exploración el cual se plantea con la estructura de una matriz.

Para la resolución de este problema se usó el lenguaje de programación Python el cual importamos nuestras clases para leer el archivo, usamos la librería para manejo de archivos XML "element tree" y así poder realizar una matriz, el cual se hace mediante 2 ciclos for para ir recorriendo y llenando dicha matriz.

Para todo el proceso se utilizo el paradigma orientado a objetos debido al cual creamos distintas clases-

Esto fue de ayuda para la creación de nuestras estructuras de datos por diferentes áreas, las clases fueron las siguientes:

a. Nodos
b. Linked List
c. Menu
d. App
c. Dijkstra

El ejemplo de la gráfica se encuentra en la figura 1. Pero para hacer las manipulaciones de la matriz tuvimos que crear un nodo, el cual sirvió parar recorrer diferentes listas

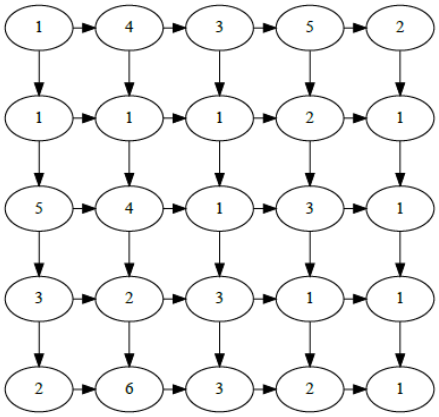

Figura 1. Grafica.

Fuente: elaboración propia.

De forma específica para realizar la lectura del archivo se utilizo una Matriz, la cual se conforma de una lista enlazada, la cual contiene dentro de si varias listas, simulando de esta forma la estructura de datos de una matriz para esto se utilizó de una función la cual contenía 2 for los cuales se encargaban de llenar la matriz

```
def matrix():
    x = Size()[0]
    y = Size()[1]
    data = Lista_new.First
    array = [[0]*x for i in range(y)]

    for n in range(x):
        for m in range(y):
            array[m][n] = str(data)
            data = data.Next
```

Figura 2. Creación de la matriz.

Fuente: elaboración propia.

Para realizar la manipulación de lo datos se hizo mediante el uso de distintas listas enlazadas las cuales contenían el terreno que nosotros escogemos mediante un input de tipo string y a su vez listas auxiliares las cuales se encargaban de contener los datos específicos que necesitábamos y así almacenar los terrenos y los valores para el tamaño de cada matriz

```
def procesar_terreno(nombre):
    node = Lista.First
    PosX = ListaX.First
    PosY = ListaY.First
    Lista_new.clear()
    X_new.clear()
    Y_new.clear()
    while node != None:
        if str(node) == nombre:
            node = node.Next
            PosX = PosX.Next
            PosY = PosY.Next
            while node != None and -1 == str(node).find("terreno"):
                Lista_new.Insert(node)
                X_new.Insert(PosX)
                Y_new.Insert(PosY)
                node = node.Next
                PosX = PosX.Next
                PosY = PosY.Next
        elif node!= None:
            node = node.Next
            PosX = PosX.Next
            PosY = PosY.Next

def Size():
    PosX = X_new.First
    PosY = Y_new.First
    while PosX.Next != None:
        PosX = PosX.Next
        PosY = PosY.Next
    j = [int(str(PosX)), int(str(PosY))]
    return(j)
```

Figura 3. Procesar el terreno.

Fuente: elaboración propia.

Para realizar la gráfica utilizamos la herramienta graphviz, la cual debe estar instalada en el ordenador al igual que un programa, y posteriormente incluido en la variables PATH, de manera que pueda ser ejecutada por medio del interprete de Python con lo cual se realizan 3 funciones declarar nodos, enlazar filas y enlazar columnas

```
matriz = matrix()

def declararNodos(listaFila):#Aca se definen las filas al mismo nivel
    global d
    global counter_mx
    with d.subgraph() as s:
        s.attr(rank='same')
        for i in listaFila:
            s.node(str(counter_mx), str(i))
            counter_mx+=1

def apuntarNodosHorizontales(lista):#Aca se apuntan los nodos de cada fila
    global d
    contador = 0
    for i in lista:
        if contador <= len(lista) and contador >= 1:
            d.edge(str(lista[contador - 1]), str(lista[contador]))
        contador += 1

def apuntarNodosVerticales(matriz):#Aca se apuntan los nodos de cada columna
    contador1 = 0
    for i in matriz:
        contador2 = 0
        for j in i:
            if (contador2 < len(i) and contador2 >= 1 ):
                if(contador2 == 1) and contador1 < len(matriz) - 1:
                    d.edge(str(i[contador2-1]), str(matriz[contador1 + 1][contador2-1]))
                if contador1 < len(matriz) - 1:
                    d.edge(str(i[contador2]), str(matriz[contador1 + 1][contador2]))
            contador2 += 1
        contador1 += 1
```

Figura 4. Realizar Grafica.

Fuente: elaboración propia.

Conclusiones

La programación orientada objetos es fundamental e importante para el desarrollo de cualquier app. Ya que debido a sus distintas ventajas ayuda a poder tener un mejor orden y manejo del programa.

El algoritmo de Dijkstra es un algoritmo desarrollado para la búsqueda de caminos mas corto en distintas estructuras de datos.

La herramienta Graphviz puede realizar gráficas de distintos TDA según la manera en que nosotros le demos uso.

Referencias bibliográficas

- https://runestone.academy/runestone/static/pythoned/Graphs/ElAlgoritmoDeDijkstra.html
- https://realpython.com/linked-lists-python/
- https://runestone.academy/runestone/static/pythoned/Trees/NodosYReferencias.html
- https://graphviz.readthedocs.io/en/stable/manual.html
- https://graphviz.readthedocs.io/en/stable/examples.html

Extensión: de cuatro a siete páginas como máximo

Adicionalmente, se pueden agregar apéndices con modelos, tablas, etc. Que complementan el contenido del trabajo.