PROYECTO 1 IPC 2

201905750 – Rony Omar Miguel López

Resumen

En el presente documento se explica el problema a resolver y los métodos utilizados para la resolución de este.

El problema a resolver o desarrollar explicado fue la optimización de un camino en base de una matriz de tamaño nxm en la cual cada cuadrante o celda tiene un valor de combustible que el robot r2e2 debe utilizar, el trabajo realizado permite definir mediante el uso de TDAs y el algoritmo de Dijkstra el camino que menos combustible utilice.
.

Palabras clave

1. TDA
2. *Nodo*
3. *Algoritmo*
4. *Push*
5. *Pop*

Abstract

In the present document it will be explained the problem taken to resolve and the methods used to resolve it.

The problem to be resolved or developed explained was the optimization of a path based on a matrix of dimensions nxm where each square or cell has a gas value that the robot r2e2 should use, the work done allows to define trough the usage of ADT and the Dijkstra algorithm the path that uses the minimum gas quantity.

Keywords

1. *ADT*
2. Node
3. Algorithm
4. Push
5. Pop

Introducción

El tema desarrollado en este proyecto fue la lectura de un documento de extensión XML así como la implementación de TDAs para poder crear listas dinámicas para poder almacenar la información obtenida como terrenos, celdas, incluso otras listas, así como el análisis de los terrenos para crear una matriz.

Con toda esa información analizada y almacenada se creó un programa que mediante el análisis de datos sea capaz de optimizar y proporcionar un camino para el robot r2e2 así como graficarlo, escribir esta ruta en un archivo o graficar la matriz del terreno.

Desarrollo del tema

Para el análisis del documento XML se utilizó la librería Element Tree siendo esta en base al modelo DOM, el proceso para analizar el archivo ocurre en el orden en que se presentan los datos en el archivo, se utilizó un método recursivo para ir obteniendo cada uno de los atributos y características del terreno.

La matriz se crea a partir de una lista de listas que se crea al ingresar las dimensiones del terreno y se modifican sus datos al analizar las posiciones y valores de cada celda.

Para la obtención del camino se utiliza el algoritmo de Dijkstra, el cual usa un método recursivo para analizar cada celda y sus cuatro posibles movimientos es decir norte, sur, este y oeste, cada iteración calcula la cantidad de combustible necesaria para llegar hasta cada una de las cuatro celdas de la celda analizada y toma el valor del que requiera menos combustible, cuando se sabe que celda se debe usar para llegar a otra celda con la mínima cantidad de combustible esta se guarda en un atributo de la celda, este algoritmo se hace en base a una celda específica que es la celda inicial.

Luego de modificar e identificar cada una de las celdas se utiliza otro método recursivo para a partir de la celda destino buscar la celda anterior a esta dentro de la matriz que permite utilizar la menor cantidad de combustible y estas se guardan en una lista que es la lista que proporciona el camino. Para graficar esto simplemente se hace una verificación en la matriz y se imprime el carácter indicado cuando la posición analizada contiene una de las celdas que forman el camino.

Para la creación del archivo XML se utilizan los datos del terreno para crear la parte del archivo que se mantiene igual, para la escritura de la ruta se utiliza un ciclo que permita ir agregando los datos de cada celda que forma parte del camino solución del problema.

Para la creación de la gráfica se utiliza un ciclo que permita recorrer la matriz y crear un nodo por cada celda y crear uniones entre estos.

.

Descripción de Métodos

A continuación, se explican algunos de los métodos más relevantes utilizados para poder obtener este camino.

Métodos de Matriz:

1. Imprimir(): Este método perteneciente a cada matriz permite imprimir de manera gráfica la matriz en consola.
2. add(): Este método permite agregar una celda dentro de la consola esto en base a sus posiciones ‘x’ y ‘y’.
3. buscar(): Este método permite buscar una celda dentro de la matriz y retornarla.
4. obtenerRuta(): Este método es el encargado de calcular todos los caminos dentro de la matriz a partir de una celda especifica y modificar cada celda para indicar la menor cantidad de combustible e indicar que celda es la que permite llegar a la siguiente celda con el menor consumo de combustible.
5. Ruta(): Este método permite obtener la lista de celdas que forma el camino y retornarlo en forma de lista.
6. imprimirRuta(): Este método es el encargado de imprimir la matriz modificada en consola verificando que se modifiquen las posiciones de acuerdo a la validación de si existe cada posición dentro de la lista de la ruta obtenida.

Métodos de Listas Simples:

1. add(): Este método permite agregar objetos a la lista.
2. Imprimir(): Método utilizado para imprimir los datos cuando se almacenan objetos de tipo terreno.
3. imprimirCelda(): Método utilizado para imprimir los datos cuando se almacenan objetos de tipo celda.
4. Buscar(): Método utilizado para buscar objetos tipo terreno dentro de la lista.
5. buscarCelda(): Método utilizado para buscar objetos de tipo celda dentro de la lista.
6. Pop(): Método utilizado para retornar el último dato guardado.

Métodos de Cola de prioridades:

1. Add(): Método utilizado para agregar celdas en la cola de prioridades utilizado para el algoritmo de dijkstra.
2. Pop(): Método utilizado para retornar el primer valor en la cola.
3. eliminar(): Método utilizado para eliminar cualquier celda dentro de la cola.

Métodos principales

1. cargarArchivos(): Método utilizado para obtener toda la información del archivo XML.
2. escribirArchivo(): Método utilizado para crear un archivo.
3. crearXML(): Método que permite crear el archivo xml de la resolución de la matriz de un terreno con las especificaciones requeridas.
4. crearGrafica(): Método utilizado para crear la grafica de la matriz utilizando GraphViz.
5. Menú(): Método encargado de organizar todo el funcionamiento y selección de opciones para ejecutar el programa.

Conclusiones

Los tipos de datos abstractos y la memoria dinámica son muy útiles para resolver muchos tipos de problemas ayudando a hacerlos de una manera eficiente y efectiva sin utilizar mas recursos de los necesitados.

Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

Diagrama de clases explicando la estructura utilizada para la resolución del proyecto.