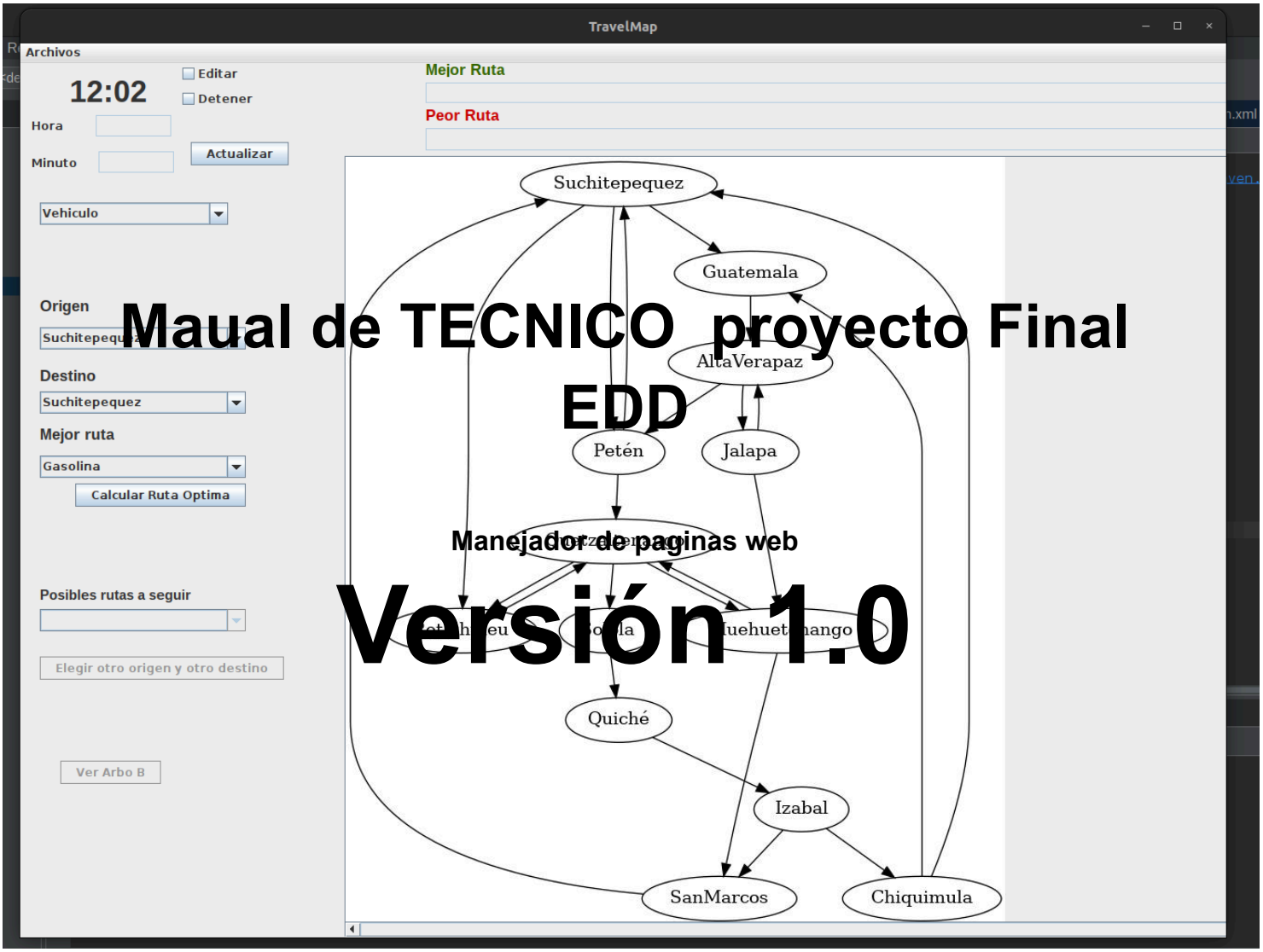
UNIVERSIDAD SAN CARLOS DE GUATEMALA
CENTRO UNIVERSITARIO DE OCCIDENTE
DIVISIÓN DE CIENCIAS DE LA INGENIERÍA
Ingeniería en Ciencias y Sistemas.
Ingeniería en Ciencias y Sistemas.
Ing. Christian Quiroa
Auxiliar: Yefer Rodrigo Alvarado

Estructura de Datos

Byron Fernando Torres Ajxup 201731523



Maual de TECNICO proyecto Final EDD

Manejador de paginas web

Versión 1.0

Quetzaltenango 05 de Mayo del 2024

Índice

Introducción

TravelMapGT es una aplicación diseñada para calcular rutas óptimas para viajar dentro de Guatemala, ya sea en vehículo o caminando. La aplicación utiliza estructuras de datos avanzadas, como grafos y árboles B, para representar y procesar la información de manera eficiente.

Representación del mapa de Guatemala

El mapa de Guatemala se representa utilizando un grafo dirigido, donde cada nodo representa una ubicación (ciudad, pueblo, etc.), y las aristas representan las rutas posibles entre esas ubicaciones. Cada arista contiene información adicional, como la distancia, el tiempo de recorrido en vehículo, el tiempo de recorrido a pie, el consumo de gasolina y el desgaste físico.

La estructura de datos utilizada para representar el grafo es una lista de adyacencia, donde cada nodo tiene una lista de los nodos adyacentes a los que está conectado. Esto permite un acceso eficiente a los vecinos de un nodo determinado y facilita operaciones como la búsqueda de rutas.

GRAFO

Grafo en Estructuras de Datos:

Un grafo en el contexto de las estructuras de datos es una colección de nodos o vértices, que están conectados entre sí mediante enlaces llamados aristas. En términos técnicos, un grafo está definido como G = (V, E), donde V es un conjunto de vértices y E es un conjunto de aristas que conectan esos vértices. Dependiendo de la dirección de las aristas y la presencia de pesos asociados, los grafos pueden ser dirigidos o no dirigidos, y ponderados o no ponderados.

Nodos (Vértices): Representan entidades individuales dentro del grafo. Cada nodo puede contener información adicional según las necesidades del problema.
Aristas: Representan las conexiones entre los nodos. Pueden tener direcciones y pesos asociados, lo que indica la dirección de la conexión y una medida adicional (como distancia o costo) respectivamente.
Los grafos son una estructura de datos versátil y se utilizan en una variedad de aplicaciones, incluidas las redes de computadoras, las redes sociales, la logística, la optimización y más.

Para encontrar rutas más cortas en grafos, hay varios algoritmos populares, cada uno con sus propias características y aplicaciones. Aquí tienes una descripción breve de algunos de los más comunes:

Algoritmo de Dijkstra:

Este algoritmo encuentra el camino más corto desde un nodo de inicio hacia todos los demás nodos en un grafo ponderado y no dirigido o dirigido, con pesos no negativos.
Funciona de manera iterativa, seleccionando el nodo con la distancia mínima al nodo de inicio en cada iteración y actualizando las distancias de los nodos adyacentes.
Es eficiente en grafos densos o dispersos, pero no puede manejar pesos negativos ni detectar ciclos negativos.

Algoritmo de Bellman-Ford:

Similar a Dijkstra, pero puede manejar grafos con pesos negativos y detectar ciclos negativos.
Funciona relajando repetidamente todas las aristas en el grafo para mejorar las estimaciones de la distancia más corta entre el nodo de inicio y todos los demás nodos.
Es menos eficiente que Dijkstra para grafos densos, pero es más versátil debido a su capacidad para manejar pesos negativos y ciclos negativos.

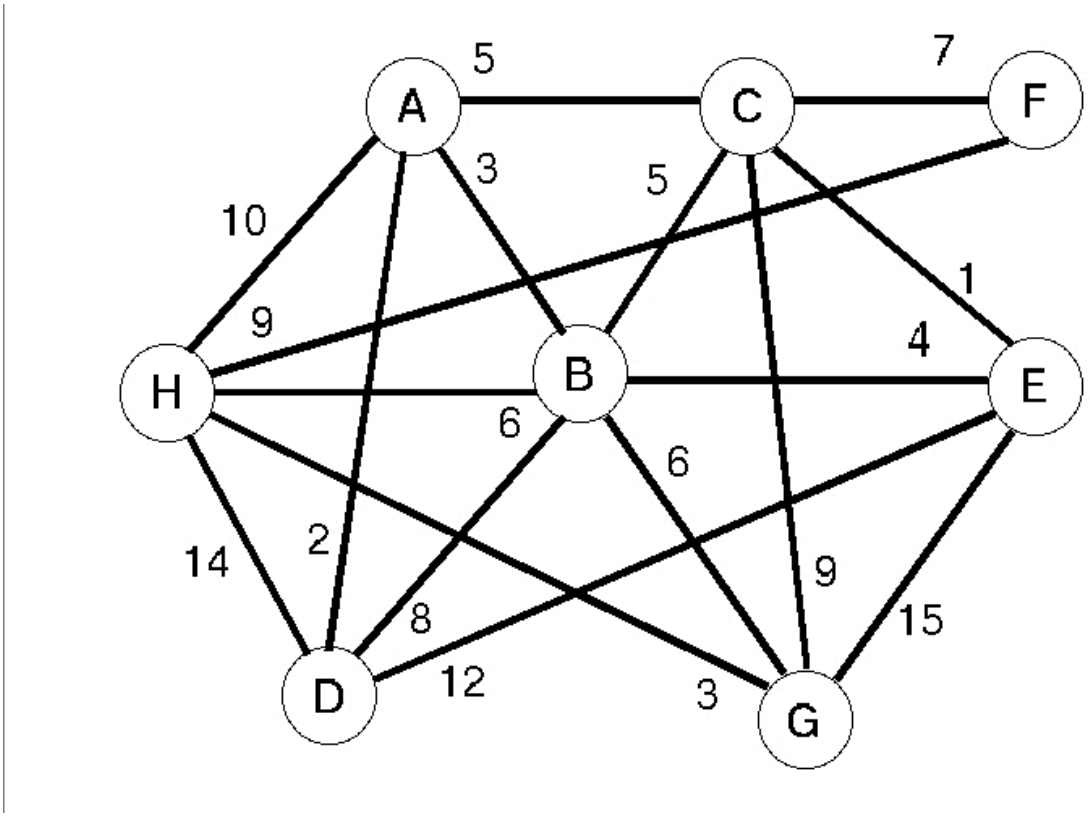
Algoritmo de Floyd-Warshall:

Encuentra los caminos más cortos entre todos los pares de nodos en un grafo dirigido y ponderado (incluso con pesos negativos), sin ciclos negativos.
Utiliza una matriz de adyacencia para almacenar las distancias entre todos los pares de nodos y actualiza iterativamente estas distancias considerando todos los nodos intermedios posibles.
Es menos eficiente que Dijkstra o Bellman-Ford para encontrar el camino más corto entre un solo par de nodos, pero es más eficiente cuando se necesita encontrar los caminos más cortos entre todos los pares de nodos.

Algoritmo A*:

Es una extensión del algoritmo de búsqueda de Dijkstra que utiliza una heurística para priorizar los nodos que probablemente conduzcan a la solución más rápida.
Utiliza una función de evaluación que combina el costo real del camino desde el nodo de inicio hasta el nodo actual con una estimación del costo restante (heurística) hasta el nodo objetivo.

Es especialmente útil en problemas de búsqueda de rutas en gráficos con muchos nodos y aristas, donde la eficiencia es crítica.



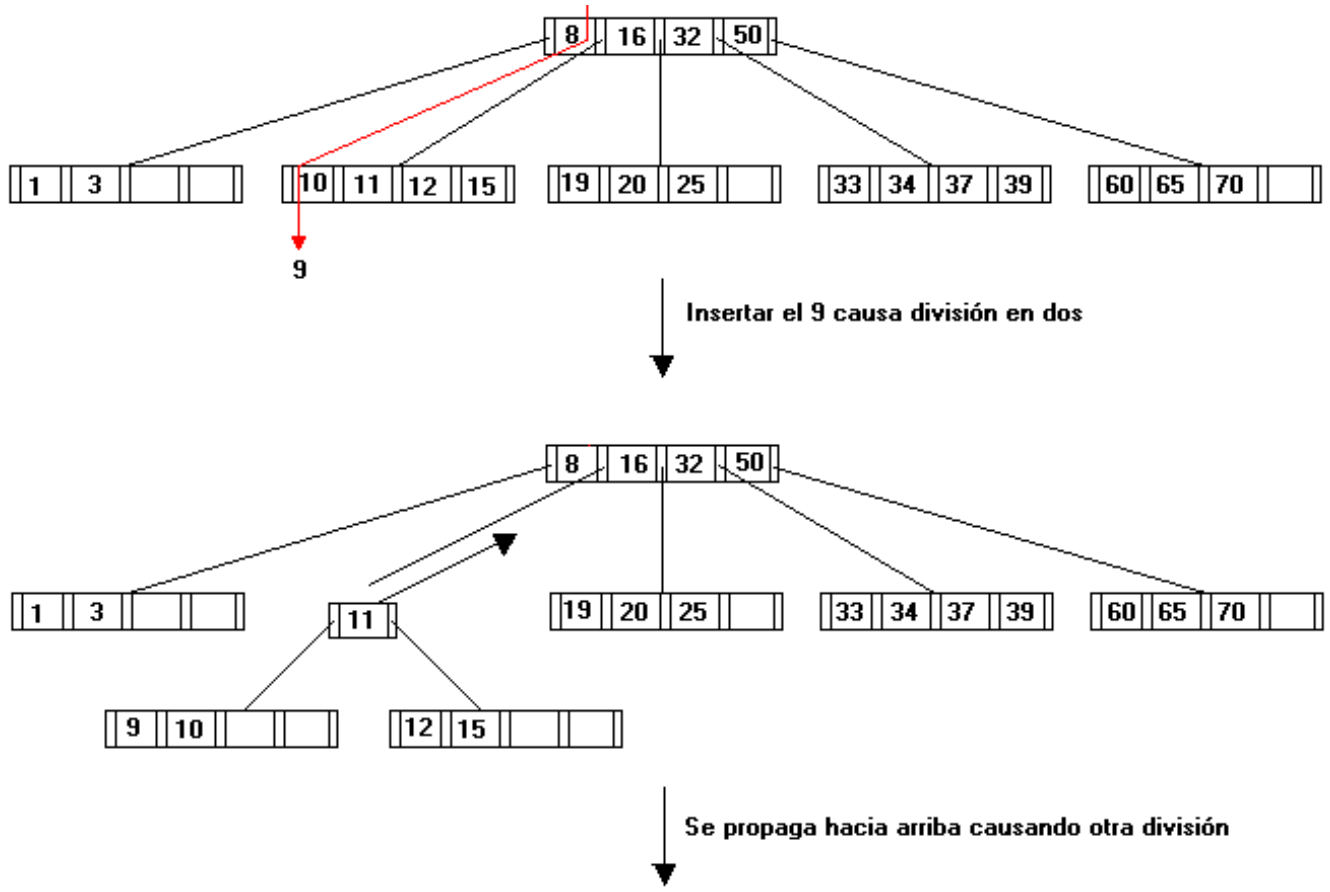
ÁRBOL B

Árbol B en Estructuras de Datos:

Un árbol B es una estructura de datos de tipo árbol que se utiliza comúnmente en la implementación de bases de datos y sistemas de archivos. Se caracteriza por su capacidad para manejar grandes conjuntos de datos y su eficiencia en la búsqueda, inserción y eliminación de elementos. Un árbol B se define por las siguientes propiedades:

Balanceo: Todos los nodos en un árbol B tienen aproximadamente la misma profundidad, lo que garantiza un tiempo de búsqueda eficiente.
Ordenación: Las claves en cada nodo están organizadas en orden ascendente o descendente, lo que facilita las búsquedas binarias.

División y Fusión Automática: Cuando un nodo alcanza su capacidad máxima, se divide en dos nodos, y cuando un nodo queda por debajo de un umbral mínimo de ocupación, se fusiona con un vecino.
Los árboles B se utilizan en aplicaciones donde se requiere un acceso eficiente a datos ordenados, como bases de datos y sistemas de archivos. Son especialmente útiles cuando se trabaja con conjuntos de datos grandes que no caben en la memoria principal, ya que minimizan la cantidad de accesos a disco necesarios para realizar operaciones de búsqueda, inserción o eliminación.

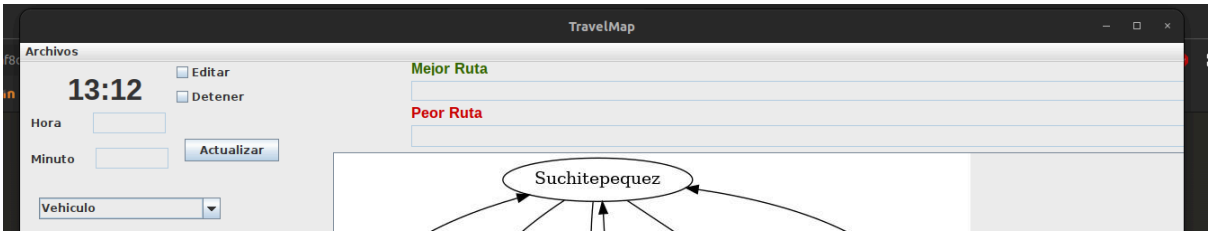


Interfaz de Usuario

Descripción de los componentes de la interfaz gráfica:

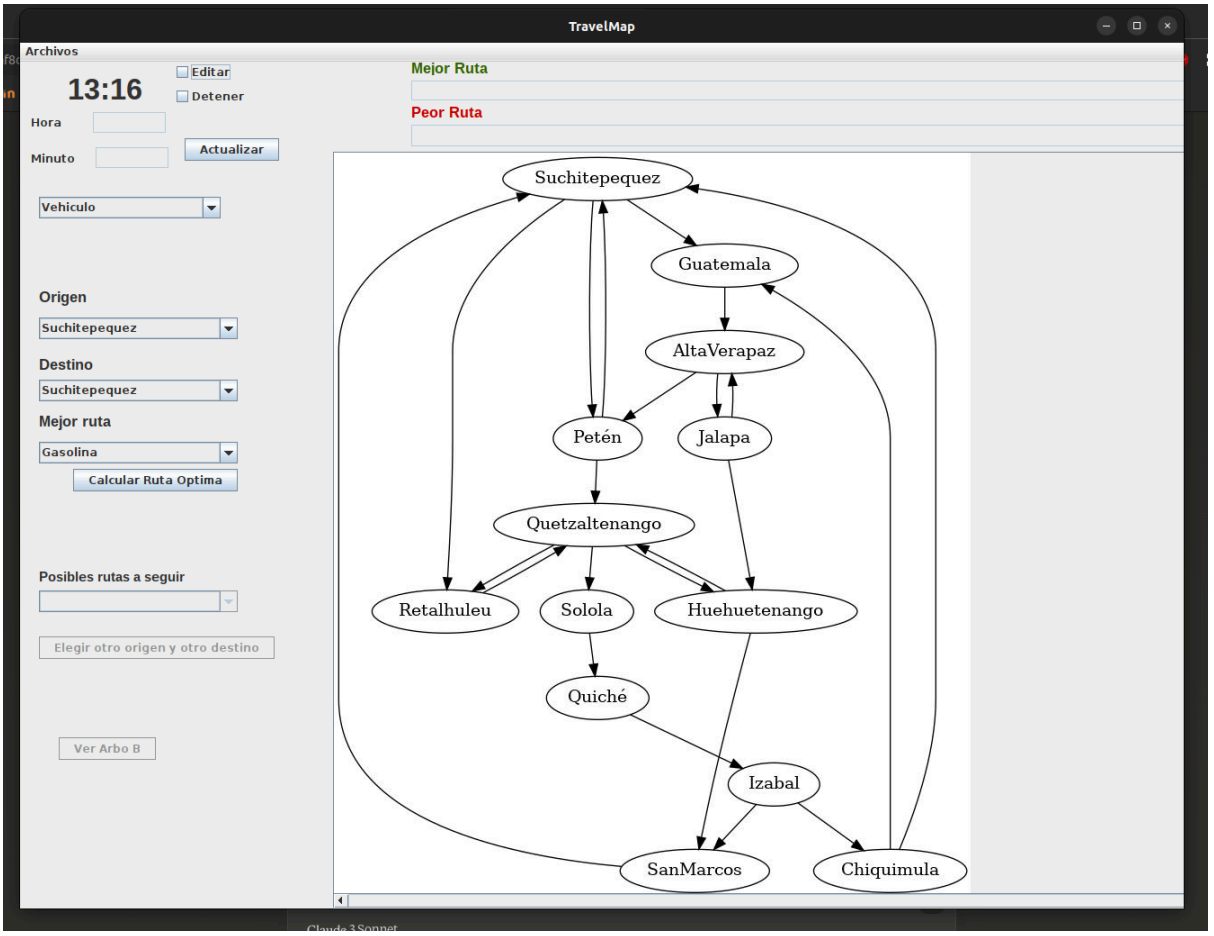
- Existe un menu en la parte superior izquierda, el cual cargara los archivos csv para generar el trafino, las rutas, etc.

Ejemplo:



- Una ves cargado los archivos debera cargar el grafo representativo de las rutas:

  Ejemplo:



Archivos de entrada:
Carga de datos

Los datos del mapa de Guatemala se cargan desde un archivo de entrada con el siguiente formato:

code<origen>|<destino>|<tiempo_vehiculo>|<tiempo_pie>|<consumo_gas>|<desgaste_persona>|<distancia>

Gestión del tráfico

La aplicación también considera las condiciones de tráfico al calcular las rutas óptimas. Los datos del tráfico se cargan desde un archivo de entrada con el siguiente formato:

Copy code<origen>|<destino>|<hora_inicio>|<hora_finaliza>|<probabilidad_trafico>