Desarrollo en Python de algoritmo de optimización para el problema de enrutamiento del vehículo (VRP)

Gonzales Bahena José Maximiliano, López Arévalo José Luis, Moisés Sotelo Rodríguez

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS, (CUCEI, UDG)

jose.gonzales6217@alumnos.udg.mx
jose.lopez1291@alumnos.udg.mx
moises.sotelo5881@alumnos.udg.mx

***Abstract*—** Este documento presenta el desarrollo en Python de un algoritmo de optimización para resolver el problema del enrutamiento del vehículo, que consiste en designar un camino a 'n' representado por un grafo pesado cantidad de vehículos que parten de un mismo origen, donde cada uno debería llegar a un destino y regresar al origen con el camino más corto posible, sin utilizar alguna calle más de una vez. Se muestra cómo se optó por un algoritmo (inserte nombre del algoritmo), y su uso dentro de una base de datos obtenida en Openstreetmap.

***Palabras claves* –** Optimización, VPR, Python

Repositorio de código:
https://github.com/Maaaxgz/Vehicle-routing-problem

Versión actual del código: Ver.4

Licencia legal código: Open Source

I. Introducción

Este documento busca abordar una solución al problema de optimización conocido como problema del enrutamiento del vehículo (de ahora en adelante llamado VRP).

II. Trabajos Relacionados

Debido a la naturaleza del problema que se está planteando resolver en este documento, anteriormente se han realizado numerosas investigaciones y/o soluciones para dicho problema.

A. Enfoque de Ventanas de Tiempo

En el año 2020, Uriel Pineda muestra un planteamiento para resolver el problema del represamiento de entregas, utilizando técnicas como el análisis de Pareto y el método Clarke-Weight, el cual consiste en encontrar una solución inicial de alto costo pero simple, para continuar con su mejora mediante la consolidación de rutas. Esto, sumado a la implementación de ventanas de tiempo, significaron una mejora en la cantidad de servicios realizados respecto a métodos anteriormente utilizados [1].

B. SimpliRoute

SimpliRoute es una plataforma dedicada a la gestión de rutas y logística. Se encargan de optimizar las rutas de entrega de vehículos mediante algoritmos avanzados, con el objetivo de minimizar tiempos de entrega y costos de gestión Utiliza varios tipos de algoritmos entre los cuales destaca el algoritmo del vecino más cercano (NN, comienza en un punto y selecciona el cliente más cercano en cada paso que da, aunque no puede garantizar la solución más óptima), optimización por colonias de hormigas (ACO, encuentra rutas mediante la exploración de soluciones) y la programación lineal entera mixta (MILP, considera restricciones y objetivos) [2].

III. Descripción del Desarrollo del Proyecto

Para el desarrollo de este algoritmo se decidió dividir la creación de este en cuatro secciones principales:

A. Base de datos

Con este algoritmo se requirió de una base de datos o información que contuviera información sobre una zona delimitada, como calles y avenidas, y la distancia y dirección de cada una de ellas. Para ello se optó por utilizar OpenStreetMap, el cual es un sitio web colaborativo donde los usuarios pueden proporcionar datos de mapas a nivel mundial. Para poder manipular esta información, se utilizó la librería 'osmnx', la cual permite modelar, analizar y visualizar en forma de grafo

redes de calles de esta plataforma en particular.

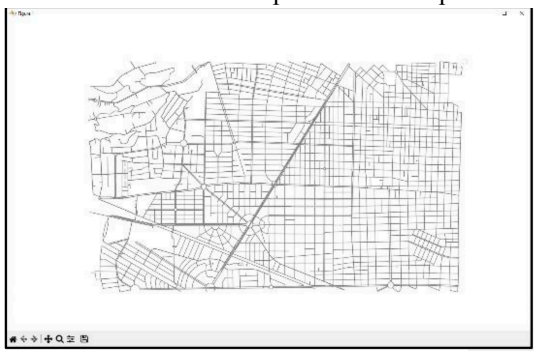

Figura 1: Visualización de una zona de Zapopan con la librería 'osmnx'

Utilizando esta librería, es posible manejar la información de un mapa como un grafo, donde los nodos o vértices corresponden a las esquinas y los arcos o aristas a las calles en sí. Teniendo una zona delimitada y un mapa con el que trabajar, pasamos al siguiente punto para el desarrollo del algoritmo.

B. Creación de la ruta inicial

Debido a que nuestro problema debe de resolverse con un algoritmo de optimización, este presenta ciertas características. Algunas de ellas son encontrar la mejor solución posible maximizando o minimizando una función, y partir de un resultado con alto coste temporal o espacial, para conseguir mejorarlo. En nuestro caso, se decidió partir por una ruta inicial simple, la cual solo debía ir desde el origen y regresar sin utilizar la misma calle dos veces.

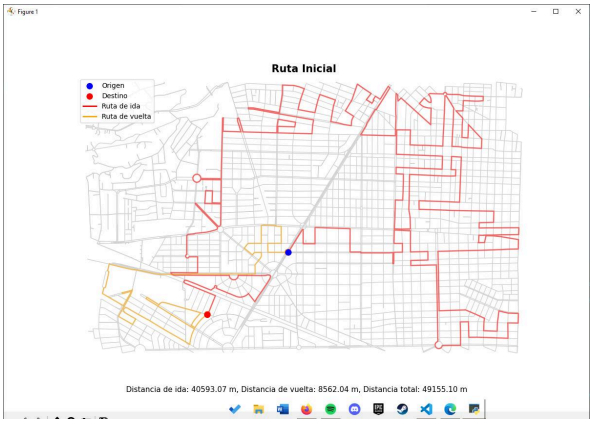

figura 2: Definición de la ruta inicial

C. Optimización de la ruta

Para la optimización de la ruta se optó por utilizar un código similar al utilizado para genera la ruta inicial, solo que ahora también se hace una verificación para la distancia que hay entre el final de la ruta y el destino, se busca encontrar el camino que más distancia reduzca entre el final de la ruta y el destino, o en su defecto, el camino que menos se aleje del mismo.

D. Interfaz gráfica

Se utilizó la biblioteca de tkinter para desarrollar una interfaz gráfica de usuario que facilita la visualización de la información de manera clara y eficiente. Esta interfaz permite a los usuarios interactuar con los datos de forma intuitiva, mejorando así la comprensión y el análisis de la información presentada. Además, la biblioteca de tkinter ofrece múltiples opciones de personalización para adaptarse a diversas necesidades.

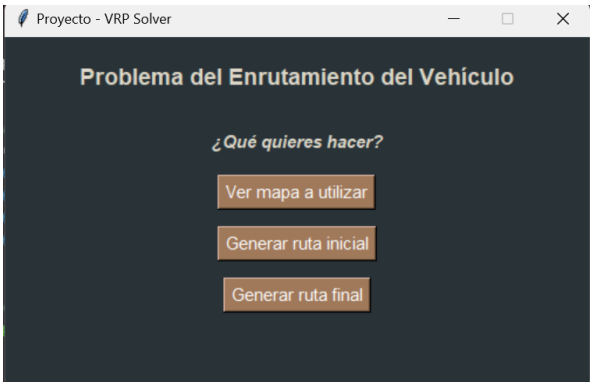

Figura 3: GUI para el usuario

En el código, se puede obtener una ruta inicial generada aleatoriamente que permite ir del punto A al punto B de la manera más corta posible. Esta ruta optimizada facilita el desplazamiento eficiente entre ambos puntos, maximizando la eficacia y minimizando el tiempo de recorrido. Con este método, logramos mejorar la planificación y ejecución de rutas de manera significativa.

Al generar la ruta final podemos observar una mejoría respecto a la ruta inicial, aunque este resultado no se puede garantizar que sea el más óptimo.

Los requisitos para poder llevar la correcta ejecución del código además de tener Python 3 fueron las siguientes librerías como base:

- Tkinter
- Random
- Osmnx
- Networkx
- Matplotlib

De este modo, es posible generar representaciones gráficas, calcular las distancias mínimas entre cada nodo existente y determinar la mejor ruta posible para cada caso particular, optimizando así la eficiencia del recorrido.

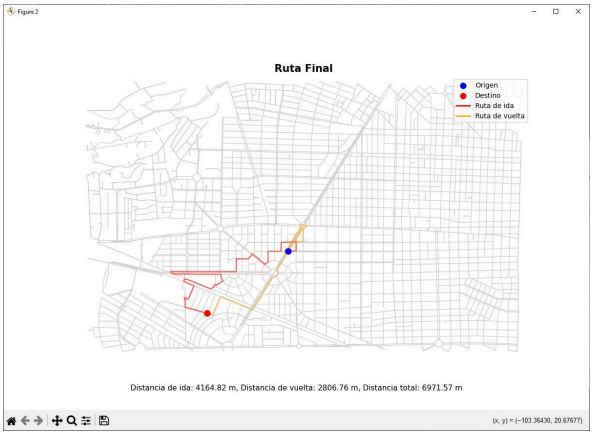

Figura 4: Ruta optimizada

IV. Resultados Obtenidos del Proyecto

Se logra el objetivo de la optimizar el problema, aunque no se garantiza que el 100% de las veces la respuesta sea la mas optima ya que por la naturaleza de nuestro algoritmo va resolviendo el problema uniendo soluciones locales a un problema global.

V. Conclusiones y Trabajo a Futuro

Aunque este algoritmo presenta una solución al problema presentado, consideramos que existen ciertas oportunidades de mejora, como la consideración de ventanas de tiempo, o considerar variables como el tráfico.

Reconocimientos

Gracias a la UDG y a Kanye West por hacer famosa a Taylor Swift..

Referencias

[1] Pineda Zapata, U. and Carabalí Ararat, H. 2020. Un Problema de Enrutamiento del Vehículo con Enfoque de Ventanas de Tiempo para Mejorar el Proceso de Entregas. Ingeniería. 25, 2 (Jul. 2020), 117–143. DOI: https://doi.org/10.14483/23448393.15271.

[2] (2018). SimpliRoute. Cómo SimpliRoute resuelve el problema de Ruteo de Vehículos [Online]. Avaliable: https://simpliroute.com/es/blog/como-simplirouteresuelve-el-problema-de-ruteo-de-vehiculos