Programación de estructuras de datos y algoritmos fundamentales (Gpo 604)

Leon Daniel Vilchis Arceo A01641413

Álvaro González Martínez A01646343

Gregorio Alejandro Orozco Torres A01641967

Daniel Hernández Gutiérrez A01640706

Lorenzo Orrante Román A01641580

Campus Guadalajara

12 de noviembre del 2025

Implementación Dijkstra's Algorithm

- **¿Qué es algoritmo Dijkstra?**

El algoritmo de Dijkstra es un método utilizado en teoría de grafos para encontrar el camino más corto desde un nodo origen hacia todos los demás nodos del grafo, siempre que las aristas tengan pesos no negativos.

Su funcionamiento se basa en asignar progresivamente la distancia mínima conocida a cada nodo, partiendo del origen y actualizando esas distancias conforme se descubren rutas más cortas. En cada paso, el algoritmo selecciona el nodo con la distancia más pequeña aún no procesada, y revisa sus vecinos para verificar si puede mejorarse la distancia hacia ellos. De esta forma, el algoritmo expande el conjunto de los nodo visitados hasta que termine de cubrir todo el grafo y garantizar que haya encontrado la ruta más corta entre el nodo de inicio.

- **Justificación del uso de la estructura de grafo seleccionada.**

La estructura de grafo que nosotros usamos en este programa está hecha con un diccionario, donde cada nodo tiene una lista de tuplas que nos van a representar sus vecinos junto con el peso de cada conexión, es decir a cuánta distancia están. Escogimos este tipo de estructura porque es muy práctica ya que nos permite acceder rápido a las conexiones de cualquier nodo y, además, es más fácil de modificar. Por ejemplo, si queremos agregar una nueva conexión entre dos puntos, solo necesitamos añadir una nueva tupla en la lista correspondiente, sin tener que cambiar toda la estructura. Esta forma de representar un grafo es muy útil cuando se trabaja con algoritmos como Dijkstra, ya que hace más fácil recorrer los nodos y actualizar las distancias cuando se encuentran caminos más cortos.

Como cada nodo guarda sus conexiones y los pesos asociados, el programa puede analizar todas las posibles rutas y escoger la más eficiente sin necesidad de usar estructuras más complicadas. Además, al estar implementado con un diccionario, el acceso a la información es rápido y directo, y Python maneja muy bien este tipo de datos, lo que ayuda a que el código sea más limpio, fácil de entender y de mantener.

- **Complejidad del algoritmo (tiempo y espacio).**

En cuanto a la complejidad, el algoritmo de Dijkstra que implementamos tiene una complejidad de tiempo aproximada de $O(N^2)$, donde N representa el número de nodos., porque en cada iteración se busca el nodo no visitado con la distancia más corta, y para eso hay que revisar todos los nodos que aún no se han visitado, para grafos pequeños o medianos como el nuestro, la versión actual es suficiente y mantiene el código mucho más claro.
En cuanto al uso de memoria, el algoritmo necesita O(N), ya que se guardan las distancias de cada nodo, los nodos visitados y el camino anterior para poder reconstruir la ruta al final.

En resumen podemos concluir que el algoritmo de Dijstra es una herramienta que puede resultar fundamental para resolver problemas en donde necesitemos generar rutas más óptimas dentro de grafos ponderados. A través de la implementación desarrollada con diccionarios y listas se representa de forma clara y eficiente la red de conexiones que gracias a su estructura, es posible aplicar el algoritmo de forma intuitiva y obtener las rutas más cortas desde un punto de partida hasta cualquier destino seleccionado dentro del grafo, así como es demostrado en el programa con los distintos caminos utilizados.

Referencias APA:

Lagares, J. (2022, 27 agosto). How does Dijkstra's Algorithm work? Easy explanation in Less than 5 Minutes. *Medium*. https://medium.com/codex/how-does-dijkstras-algorithm-work-easy-explanation-in-less-than-5-minutes-e27f46795c18