Trabajo práctico especial

Integrantes:

Arleo, Agustin.

Duana, Lucas

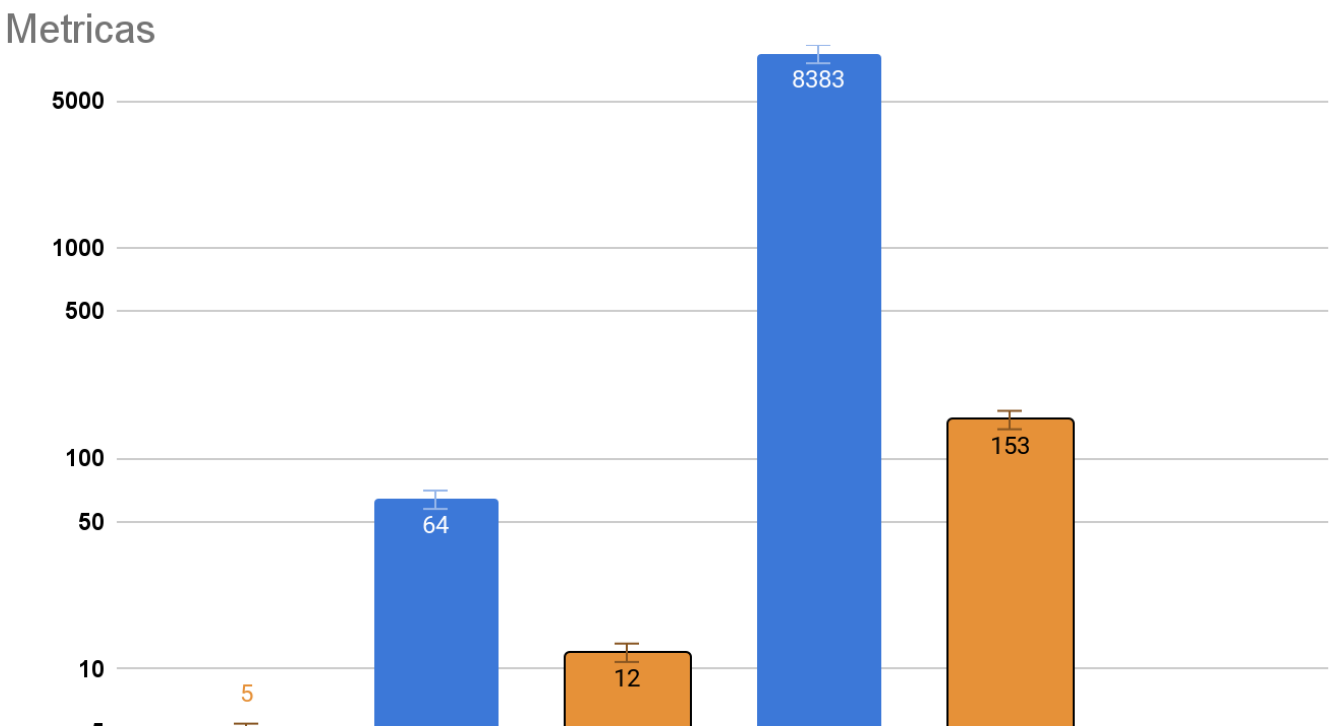
Para poder cumplir con la parte dos del trabajo práctico especial utilizamos las dos técnicas algorítmicas solicitadas (Backtracking y Greedy). En los siguientes párrafos confeccionamos un resumen de cada algoritmo y cómo se resolvió.

Estrategia Backtracking:

La estrategia de backtracking utilizada es un enfoque recursivo para encontrar un árbol de costo mínimo en el grafo. El algoritmo explora todas las posibles combinaciones de arcos para construir una solución factible y luego realiza una búsqueda exhaustiva para encontrar la solución óptima.
El algoritmo comienza con un conjunto vacío de estaciones visitadas y el vértice inicial como la estación actual. Luego, itera sobre todos los arcos disponibles y verifica si se puede agregar un arco a la solución actual cumpliendo ciertas condiciones, como que el arco sea válido, la distancia actual sea menor que la menor distancia encontrada hasta el momento y el vértice destino no haya sido visitado antes. Si se cumplen estas condiciones, se agrega el arco a la solución actual y se realiza una llamada recursiva al método con el vértice destino como el nuevo vértice actual.
Después de la llamada recursiva, se revierten los cambios realizados para explorar otras posibilidades.
La recursión continúa hasta que todas las estaciones hayan sido visitadas. En ese punto, se verifica si la solución actual es la mejor solución encontrada hasta el momento y se actualiza en caso afirmativo. Finalmente, el método retorna la mejor solución encontrada.
El algoritmo realiza un recorrido exhaustivo de todas las posibles combinaciones de arcos y estaciones, lo que implica que cada arco se puede seleccionar o descartar, y cada estación puede ser visitada o no visitada. Esto genera un espacio de búsqueda de tamaño 2^V.
En cada paso del backtracking, se itera sobre todos los arcos restantes en la lista arcos, lo cual tiene una complejidad de O(E)
Dado que el algoritmo debe explorar todas las posibles combinaciones, el número total de iteraciones será proporcional a la cantidad de arcos (E) y el tamaño del espacio de búsqueda (2^V), lo que resulta en una complejidad de O(E * 2^V).

Estrategia Greedy:

La estrategia Greedy implementada para resolver este ejercicio es el algoritmo de Prim.
El algoritmo encuentra un subconjunto de aristas que forman un árbol con todos los vértices, donde el peso total de todas las aristas en el árbol es el mínimo posible.
Se incrementa continuamente el tamaño de un árbol,comenzando por un vértice inicial al que se le van agregando sucesivamente vértices cuya distancia a los anteriores es mínima.
Esto da a entender que, en cada paso, las aristas a considerar son aquellas que inciden en vértices que ya pertenecen al árbol.
El árbol recubridor mínimo está completamente construido cuando no quedan más vértices por agregar.
El bucle principal del algoritmo se ejecuta hasta que todos los vértices hayan sido visitados, lo cual ocurre a lo sumo V veces. En cada iteración del bucle, se extrae el arco de menor costo de la cola de prioridad, lo cual tiene una complejidad de O(log V). Además, se recorren todos los arcos del grafo para encontrar los arcos que conectan el vértice visitado con un vértice no visitado, lo cual tiene una complejidad de O(A). Por lo tanto, la complejidad del bucle principal es O((V + A) log V).
En resumen, la complejidad total del algoritmo de Prim implementado es O((V + A) log V), donde V es el número de vértices y A es el número de arcos en el grafo.



Estos son los datos arrojados en cada ejecución para cada dataset (DsX). Como se puede observar la métrica de backtracking en el dataset 3 está vacía ya que al momento de ejecutar el algoritmo con estos datos, esta lleva mucho tiempo en terminar.

Conclusión

Como conclusión, hemos utilizado dos estrategias algorítmicas distintas, Backtracking y Greedy, para resolver el problema de construir una red de subterráneos que conecte todas las estaciones de una ciudad con la menor cantidad de metros de túnel posibles.

En base a los resultados obtenidos, pudimos evaluar la eficiencia de ambas técnicas. La estrategia Greedy (algoritmo de Prim) puede proporcionar una solución rápida y razonablemente buena, aunque no garantiza la solución óptima en todos los casos. Por otro lado, la estrategia Backtracking garantiza la obtención de la solución óptima, pero a expensas de un mayor tiempo de ejecución en instancias más grandes debido a su costo computacional exponencial.

Link al editable de este documento:

https://docs.google.com/document/d/1n3DVA1DLRwSh_rYfRBa4w3QvdR5bXlVhe6zPU25EE2k/edit

Link al repositorio:

https://github.com/ArleoAgustin/TPE_Programacion3_Parte2