Documento de análisis Reto 4

Juan David Guevara Arévalo y Kevin David Hernández Tapiero

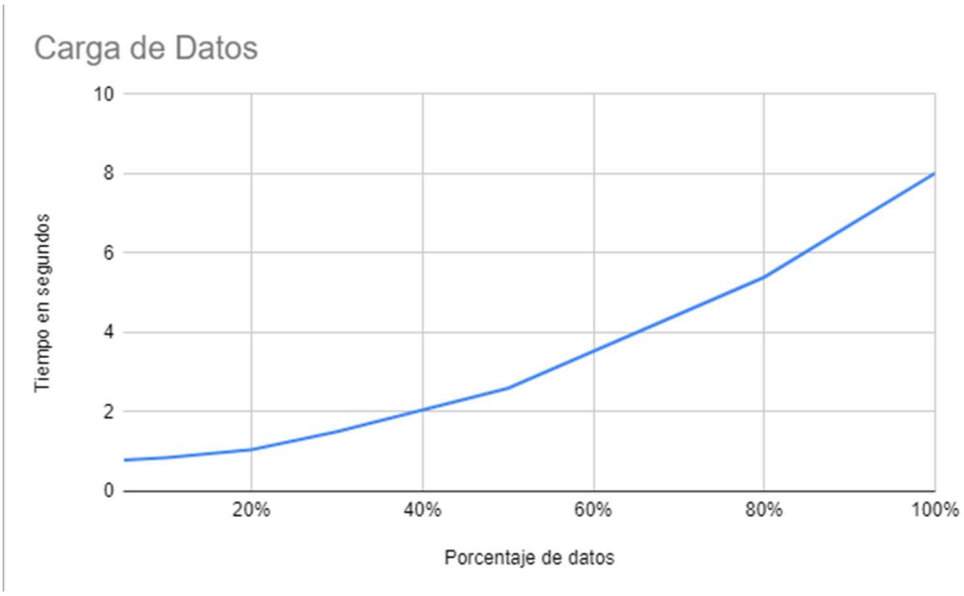
Departamento de Ingeniería de Sistemas y Computación, Universidad de los Andes

8 de diciembre de 2021

A continuación, se presenta el documento de análisis del Reto 4, el reto fue realizado por **Kevin David Hernández Tapiero** con el código **202111724** (**k.hernandezt@uniandes.edu.co**) y por **Juan David Guevara Arévalo** con el código **202116875** (**jd.guevaraa1@uniandes.edu.co**). Ambos estudiantes pertenecen al Grupo 1 de la Sección 1 de Estructura de Datos y Algoritmos.

Carga de Datos:

La carga de datos consiste en crear un grafo dirigido y uno no dirigido a partir de un archivo de rutas, de aeropuertos y de ciudades, debido a que la cantidad de rutas y aeropuertos cada vez es mayor, el tiempo de ejecución de la función cada vez es mayor, el siguiente grafico muestra el comportamiento con el 5%, 10%, 20%, 30%, 50%, 80% y 100% de los datos:



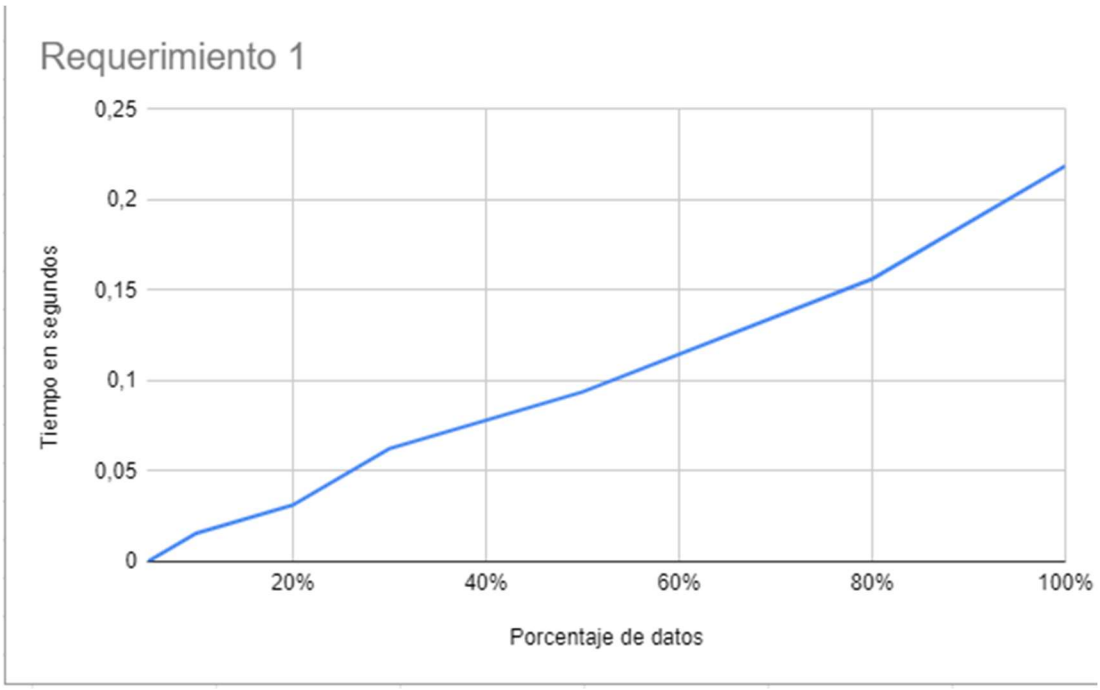
Análisis del requerimiento 1:

Problema. Como analista de vuelos deseo encontrar el (los) aeropuerto(s) que sirven como punto de interconexión a más rutas aéreas en la red en cada uno de los grafos.

Complejidad Temporal

Para este requerimiento la complejidad temporal es O(V**2), donde V es la cantidad de vértices del grafo, porque en la solución del problema se recorrieron todos los vértices y se comparan para saber cuál es el que tiene la mayor cantidad de arcos vinculados usando el TAD map y ordered map junto con las funciones del TAD graph.

El siguiente grafico muestra el comportamiento con el 5%, 10%, 20%, 30%, 50%, 80% y 100% de los datos:



Análisis del requerimiento 2:

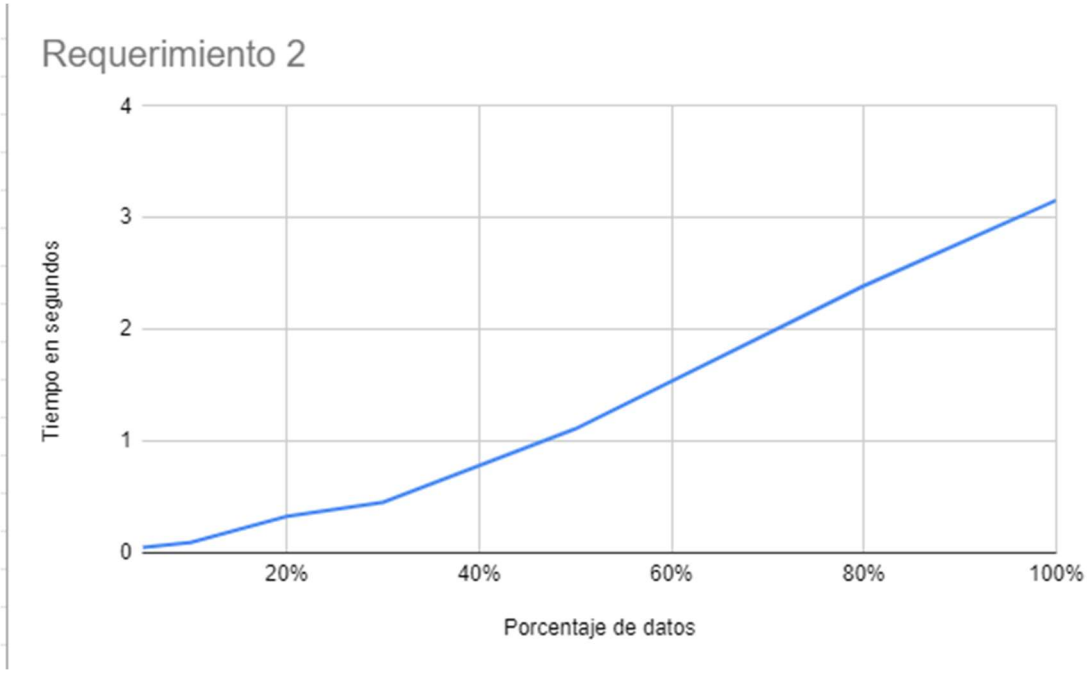
Problema. Como analista de vuelos deseo encontrar la cantidad de clústeres (componentes fuertemente conectados) dentro de la red de tráfico aéreo y si dos aeropuertos pertenecen o no al mismo clúster.

Complejidad Temporal

La complejidad temporal de este requerimiento es la misma que la del algoritmo de Kosaraju, es decir, O(V+E), donde V es la cantidad de vértices y E la cantidad de arcos del grafo.

Para resolverlo se utilizó el algoritmo de Kosaraju para saber la cantidad de componentes fuertemente conectados y la función stronglyConnected para saber si los dos aeropuertos entrantes por parámetro estaban fuertemente conectados.

El siguiente grafico muestra el comportamiento con el 5%, 10%, 20%, 30%, 50%, 80% y 100% de los datos:



Análisis del requerimiento 3:

Problema. Como analista de vuelos deseo encontrar la ruta mínima en distancia para viajar entre dos ciudades, los puntos de origen y de destino serán los nombres de las ciudades.

Complejidad Temporal

Para este requerimiento inicialmente se selecciona cada ciudad según el número de ciudades homónimas, la complejidad temporal de este proceso es $O(2H)$ donde H es el número de ciudades Homónimas.

El segundo proceso en este requerimiento es buscar el aeropuerto más cercano a las ciudades previamente seleccionadas. La complejidad temporal será:

$$O(/\frac{D}{10}+1-\left(\frac{D}{10}\%1\right)+N)$$

siendo d la distancia en kilómetros del aeropuerto más cercano y n el número de aeropuertos encontrados en el rango, por ejemplo: se encontró que hay 2 aeropuertos a una distancia de 79 kilómetros reemplazando sería:

$$O\left(\left(\frac{79}{10}+1\right)-\left(\frac{79}{10}\%1\right)+2\right)=O(10)$$

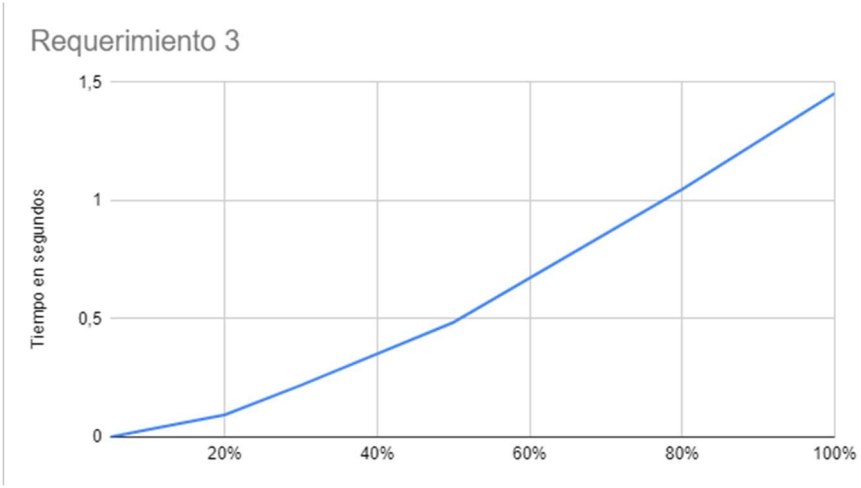
El ultimo proceso es calcular la ruta mínima entre los aeropuertos determinados anteriormente mediante el algoritmo de Dijkstra, la complejidad temporal de este paso es $O(ElogV)$ donde V es el número de vértices (aeropuertos) y E el número de arcos (rutas).

Dadas las anteriores complejidades la complejidad total del requerimiento es:

$$O\left(2H+2\left(\left(\frac{D}{10}+1\right)-\left(\frac{D}{10}\%1\right)+N\right)+ElogV\right)$$

Donde H es el número de ciudades Homónimas, D la distancia en kilómetros del aeropuerto más cercano a una ciudad, N la cantidad aeropuertos en el rango, E la cantidad de arcos (rutas) y V la cantidad de vértices (aeropuertos).

El siguiente grafico muestra el comportamiento con el 5%, 10%, 20%, 30%, 50%, 80% y 100% de los datos:



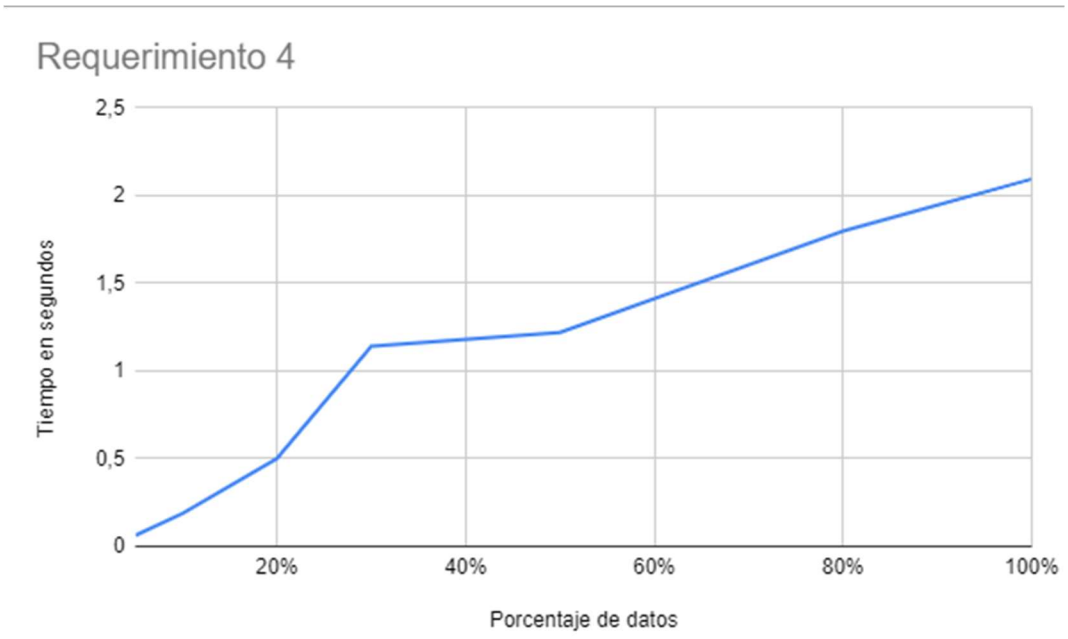
Análisis del requerimiento 4:

Problema. Como viajero desea utilizar sus millas para realizar un viaje redondo que cubra la mayor cantidad de ciudades que él pueda visitar. Para ello se necesita identificar el árbol de expansión mínima en cuanto a distancia que maximice la cantidad de ciudades de la red (representadas por los aeropuertos).

Complejidad Temporal

Este requerimiento tiene una complejidad temporal aproximada de O ((V + E) Log V), debido a que se utiliza el algoritmo Prim para encontrar el árbol de expansión mínima del grafo, y cada vértice debe entrar a la cola de prioridad solo una vez y este proceso toma un tiempo logarítmico.

El siguiente grafico muestra el comportamiento con el 5%, 10%, 20%, 30%, 50%, 80% y 100% de los datos:



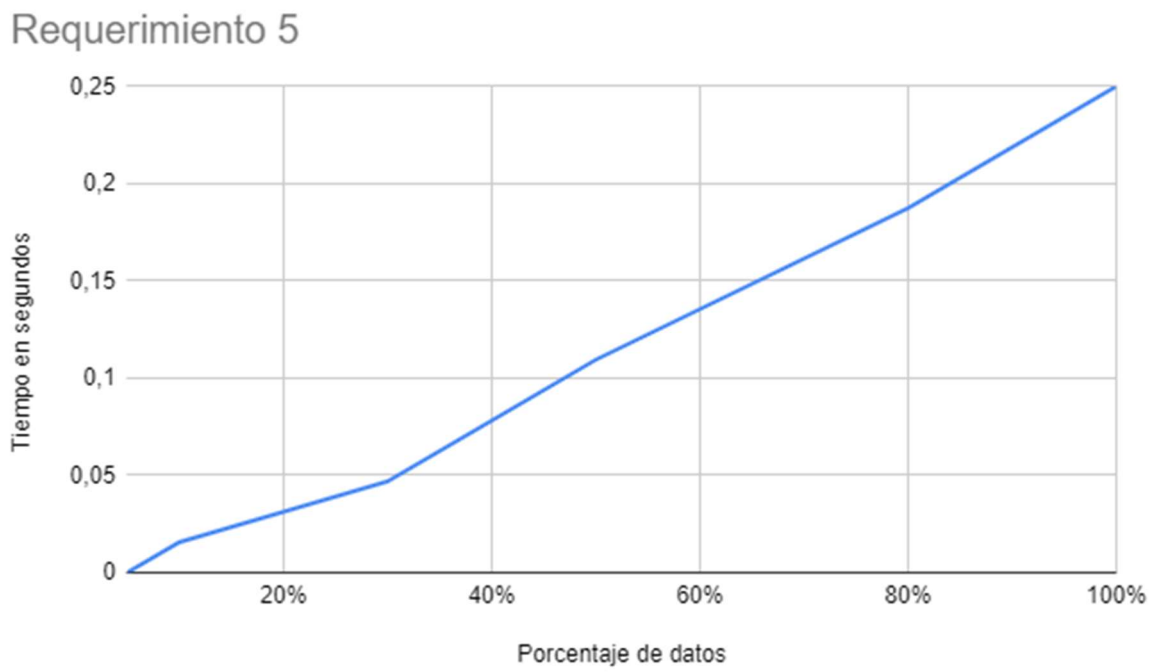
Análisis del requerimiento 5:

Problema. Como administrador de tráfico aéreo deseo conocer el impacto que tendría que un aeropuerto específico saliera de funcionamiento. Para tal fin se requiere conocer la lista de aeropuertos que podrían verse afectados.

Complejidad Temporal

En este requerimiento la complejidad temporal es de O (E + (v-1)) aproximadamente, debido a que en este requerimiento se recorren los arcos adyacentes al vértice que se "elimina", en el peor de los casos dicho vértice está conectado con todos los demás, por lo que tendría que pasar por todos los arcos cada vez.

El siguiente grafico muestra el comportamiento con el 5%, 10%, 20%, 30%, 50%, 80% y 100% de los datos:



Análisis del requerimiento 6:

Problema. Como administrador aéreo se desea encontrar la ruta mínima en distancia para viajar entre dos ciudades. Teniendo en cuenta que en cada ciudad se debe determinar el aeropuerto más cercano y relevante por su flujo de vuelos (ej.: El Aeropuerto Internacional el Dorado es más importante que el aeropuerto de Guaymaral), utilice el API Airport Nearest Relevant para identificar los aeropuertos tanto de origen como de destino y compararlos con los resultados del requerimiento 3

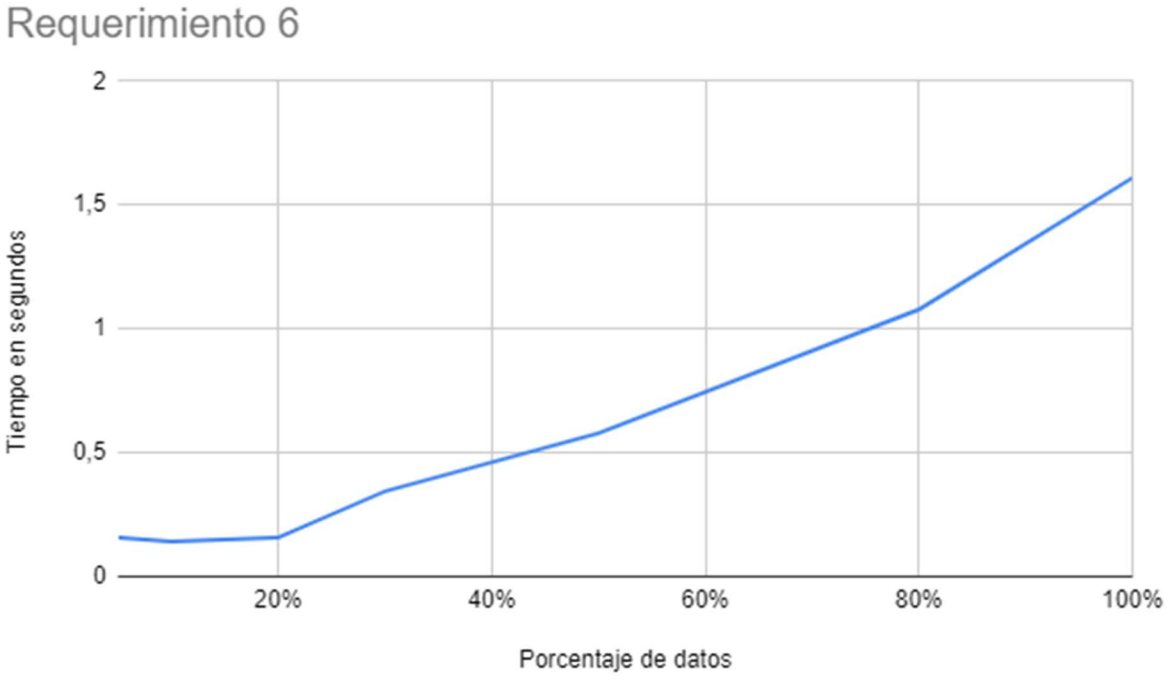
Complejidad Temporal

Para este requerimiento la complejidad temporal es igual a la complejidad del requerimiento 3, sin embargo, al realizar la búsqueda dos veces (una usando el API web y otra sin usarla) la complejidad se multiplica por dos dando como resultado lo siguiente:

$$O(2\left(2H + 2\left(\left(\frac{D}{10} + 1\right) - \left(\frac{D}{10}\%1\right) + N\right) + ElogV\right))$$

Donde H es el número de ciudades Homónimas, D la distancia en kilómetros del aeropuerto más cercano a una ciudad, N la cantidad aeropuertos en el rango, E la cantidad de arcos (rutas) y V la cantidad de vértices (aeropuertos).

El siguiente grafico muestra el comportamiento con el 5%, 10%, 20%, 30%, 50%, 80% y 100% de los datos:



Análisis del requerimiento 7:

Problema. Grafiquen un mapa con los resultados de cada uno de los requerimientos obligatorios del enunciado (del primero al quinto).

Complejidad Temporal

Para este requerimiento la complejidad temporal es la complejidad temporal del requerimiento que se esté ejecutando ya que se implementó un menú que permite ver gráficamente el requerimiento que el usuario indique. Depende del requerimiento se deben sumar las siguientes complejidades:

Requerimiento 1:

$$O(N)$$

Siendo N la cantidad de aeropuertos que funcionan como punto de interconexión aérea.

Requerimiento 2:

$$O(2)$$

Aquí es O (2) debido a que se insertan los 2 aeropuertos que se ingresaron como parámetro.

Requerimiento 3:

$$O(E)$$

Siendo E la cantidad de rutas necesarias para llegar del aeropuerto A al aeropuerto B

Requerimiento 4:

$$O(E)$$

Siendo E la cantidad de rutas realizadas por el viajero

Requerimiento 5:

$$O(E + N)$$

Siendo E la cantidad de rutas que dejarían de funcionar si el aeropuerto se cierra y N el número de aeropuertos afectados