FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

TP2 - Algueiza

Algoritmos y Programación II
Cátedra Buchwald - 1C 2023

Integrantes:

Alexia Kimberly Aroa Curico - 110014
Marcela Jazmin Cruz - 110066

Ayudante:

Juan Martín de la Cruz

FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Introducción

A continuación se realizará una descripción acerca del sistemas de consultas de vuelos desarrollado, en el cual si bien se utilizaron TDAs desarrollados a lo largo de la cursada, también se crearon nuevos para facilitar el desarrollo.

TDAs desarrollados para el Trabajo Práctico

Aeropuerto

Aeropuerto es un TDA creado para contener y manejar la información sobre el conjunto de vuelos ingresados al sistema, cumpliendo los tiempos de complejidad pedidos a la hora de ejecutar las diferentes funcionalidades. Se definen los siguientes tipos de campos:

- Hash de Vuelo
- ABB's de Listas de vuelos
- Hash de Conexiones

Vuelo

Creado con la funcionalidad de almacenar toda la información recibida sobre cada vuelo a la hora de utilizar el ***comando "****agregar_archivo****"***. Además, se pueden realizar consultas sobre los diferentes vuelos, así como también, se pueden actualizar los datos de estos.

Fecha

Este TDA almacena información sobre las fechas de los vuelos. Se pueden realizar comparaciones entre dos fechas y saber si estas son válidas para realizar las funcionalidades del sistema.

Conexión

Reúne toda data relacionada a las conexiones entre diferentes vuelos, almacenadas en un ABB, para obtener el siguiente vuelo que cumpla con lo especificado por el usuario.

Desarrollo del sistema con los TDAs implementados

Se desarrollaron las siguientes funcionalidades:

1. **Agregar_Archivo:** Se crean distintos vuelos con los datos obtenidos de los archivos, creando sus conexiones con el origen y destino. Luego, se guardan en el Aeropuerto.

2. **Ver_Tablero:** Se hacen uso de los ABBs para mostrar los vuelos en un rango determinado por las fechas, de forma ascendente o descendente. Siendo v la cantidad de vuelos, se cumple la complejidad de ser *O(v)* en el peor caso (en el que se tenga que mostrar todos los vuelos del sistema) ó *O(log v)* en un caso promedio (en el que no se pidan mostrar demasiados vuelos).

3. **Info_vuelo:** Para cumplir con la complejidad pedida, se utiliza un Hash que contiene los vuelos del sistema, sabiendo que obtener un vuelo es *O(1)* y que dicho Vuelo puede mostrar su información en *O(1)* también.

4. **Prioridad_Vuelos:** Como se requería que sea *O(n + K log n)*, siendo *K* la cantidad de vuelos a mostrar y *n* la cantidad de vuelos en el sistema, se decidió iterar el Hash de Vuelos (teniendo complejidad *O(n)*) para cargarlos en un Heap de ***vuelos prioritarios*** el cual ordena los vuelos según la prioridad de cada uno (teniendo complejidad *O(log n)*). Una vez terminado esto, se desencolan *K* cantidad de vuelos pedidos por el usuario. Como desencolar del Heap es *O(log n)* y se realiza por *K* vuelos, se cumple la complejidad pedida.

5. **Siguiente_vuelo:** Cuando el usuario ingrese los datos para obtener el siguiente vuelo, se chequea la conexión origen-destino en el Hash de conexiones (*O(1)*) del Aeropuerto. Se implementó el TDA Conexión debido a que por cada conexión existente podemos acceder un ABB que tenga ordenado por fechas los vuelos que hacen dicha conexión, teniendo en cuenta que no se realiza una iteración completa, sino solo para buscar la fecha siguiente a la recibida y obtener el vuelo en cuestión.

6. **Borrar:** Para cumplir con la complejidad *O(K log n)*, siendo *K* la cantidad de vuelos que hay en el rango de fechas ingresado y *n* la cantidad de vuelos en todo el sistema, se planteó iterar un ABB entre las fechas recibidas (incluídas) para obtener tanto los vuelos como todas las fechas a borrar, teniendo en cuenta que la cantidad vuelos es mayor que la cantidad de fechas (*O(K)*). Una vez obtenidas las fechas, se borran de los vuelos de los ABBs del Aeropuerto (*O(log n)*), del Hash de vuelos (*O(1)*) y Hash de Conexiones (*O(1)*).