Facultad de Ingeniería
Departamento de Ingeniería de Sistemas
ISIS 1225 - Estructuras de datos y algoritmos
Reto 2

Docente: Dario Ernesto Del Carmen Correal Torres

Estudiantes:

- María Alejandra Vargas Torres 201123148 - ma.vargas73@uniandes.edu.co
  - Requerimiento 2
- Andrés Felipe Molina Mahecha 201923434 - a.molinam@uniandes.edu.co
  - Requerimiento 3
- Juan Sebastián Alegría Zúñiga 202011282 - j.alegria@uniandes.edu.co
  - Requerimiento 4

Requerimientos del reto:

1. **Requerimiento uno:**
   a. Orden de crecimiento temporal: O(n)
2. **Requerimiento dos:**
   a. Encargado: Maria Alejandra Vargas Torres.
   b. Orden de crecimiento temporal: O(n)
3. **Requerimiento tres:**
   a. Encargado: Andrés Felipe Molina Mahecha.
   b. Orden de crecimiento temporal: --
4. **Requerimiento cuatro:**
   a. Encargado: Juan Sebastián Alegría Zúñiga.
   b. Orden de crecimiento temporal: O(n)
5. **Requerimiento cinco:**
   a. Orden de crecimiento temporal: O(n)

Link al repositorio: https://github.com/EDA-SEC-02-EQUIPO-04/Reto2_202020

Preguntas de análisis:

1. ¿Qué estructura debería tener el catálogo?

   R// El catálogo debería ser una estructura conformada principalmente por mapas y tablas de hash, ya que esto nos permite optimizar aún más las operaciones que ofrece nuestro programa, a que se usarán listas encadenadas o arreglo. De este modo, se obtienen búsquedas por directores, productoras, y promedios de una manera mucho más veloz (orden O(1)) al estar siempre toda la información actualizada recién se agregue o cargue un nuevo set de datos a la estructura del catálogo.

2. ¿Qué hacer al procesar cada línea de cada uno de los archivos?

   R// Al procesar cada línea de los archivos se debe agrupar y aplicar todas las operaciones necesarias para guardar en mapas funcionales específicos de las necesidades del programa. Además, se obtienen datos como el promedio de las películas para ir calculando y que a futuro la consulta de esa información sea mucho más rápida para el usuario.

3. ¿Cómo se debería ordenar la información?, ¿Por qué concepto (fecha, actor, director)?

   R// La información se debería ordenar por cualquier concepto, este ordenamiento es propio de la necesidad del programa y la información más relevante para quien se esté diseñando el programa.

4. ¿Cómo usar la menor cantidad de memoria posible?

   R// Si lo que se quiere es usar la menor cantidad de memoria posible, tendríamos que retornar a las estructuras de datos más sencillas, listas como arreglos o listas

Link al repositorio: https://github.com/EDA-SEC-02-EQUIPO-04/Reto2_202020

encadenadas, pero esta decisión podría llegar a incrementar bastante los tiempos de consulta e interacción con los datos.

5. ¿Cómo se pueden utilizar las estructuras de datos vistas en clase?

   R// Para utilizar las estructuras de datos vistas en clase, se podría aplicar la función hash al id de las películas y en base a esto ir agrupando los datos en otros mapas, de acuerdo a las necesidades y requerimientos del reto.

6. ¿Cuáles son los campos ideales para definir la llave de un Map?

   R// La llave de un mapa ideal es aquella que produzca la menor cantidad de colisiones, esto podría ser un ID único de cada película. Pero igualmente, nos conviene usar otros campos como el nombre del director, el género de películas o la productora para de este modo agrupar las películas y mostrarlas de manera más eficaz al hacer una consulta.

Las siguientes tablas muestran que en cuestión de tiempo es mucho más rápido la implementación de este reto que utilizando maps y tablas de hash.

Tamaño tabla	Requerimiento 2	Requerimiento 3	Requerimiento 4
50000	0,0015625	-	0.008705
100000	0,0046875	-	0.010298
150000	0	-	0.01197
200000	0.0015625	-	0.0105
2500000	0.0015625	-	0.01119

Tabla 1. Comparación de requerimientos individuales respecto a implementación anterior[Probing]

Nota: Teniendo en cuenta el tamaño de la tabla, ya que el factor de carga (FC) es menor a 0.1, se tomó como referencia un FC de 0.4 para el requerimiento 2 y a su vez, al utilizar un computador

con las siguientes características : Procesador Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz, 1190 Mhz, 4 procesadores principales, 8 procesadores lógicos, se identificó que los tiempos son mucho menores.

.

Tamaño tabla	Requerimiento 2	Requerimiento 3	Requerimiento 4
50000	0	-	0.011253
100000	0	-	0.010125
150000	0	-	0.009494
200000	0,0046875	-	0.009658
2500000	0.0015625	-	0.00878

Tabla 1. Comparación de requerimientos individuales respecto a implementación anterior[Chaining]

Fuente de Datos	Arraylist [ms]	Singlelinkedlist [ms]
load_csv_file	62,5	46,875
count_director_movies	15,625	15,625
get_director_movies_id	0,015625	15,625
get_actor_name_id	0,015625	15,625
get_actor_director	0,015625	0,078125
find_good_movies	48,875	34796,875
short_movies	Tienen el mismo tiempo del ordenamiento que se seleccione en order_movies	Tienen el mismo tiempo del ordenamiento que se seleccione en order_movies
rank_movies		
understand_genre	93,75	203,125
know_director	48,875	78,125
know_actor	31,25	46,875

Link al repositorio: https://github.com/EDA-SEC-02-EQUIPO-04/Reto2_202020

order_movies		
Insertion sort	11250	437,5
Shell sort	46,875	1750
Selection sort	6156,25	281,25
rank_movies_on_genres		
Insertion sort	593,75	609,375
Shell sort	46,875	46,875
Selection sort	1171,875	1109,375

Tabla 3. Tiempo de operaciones en implementación anterior

Link al repositorio: https://github.com/EDA-SEC-02-EQUIPO-04/Reto2_202020