OBSERVACIONES DEL LA PRACTICA

Santiago Tenjo Venegas Cod 202113965
Juan Esteban Jiménez Benavides Cod 201922487

	Máquina 1	Máquina 2
Procesadores	Intel® Core™ i5-4210U CPU @ 1.70GHz × 4	3.3 GHz 6-Core Intel Core i5
Memoria RAM (GB)	7.7 GiB	8 GB 2667 MHz DDR4
Sistema Operativo	Solus 4.3 Fortitude	MacOS Monterrey 12.2.1

Tabla 1. Especificaciones de las máquinas para ejecutar las pruebas de rendimiento.

Maquina 1

Resultados

Carga de Catálogo PROBING

Factor de Carga (PROBING)	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
0.10	67049.55	3481.1
0.50	67049.12	3449.61
0.90	67049.30	3447.31

Tabla 2. Comparación de consumo de datos y tiempo de ejecución para carga de catálogo con el índice por categorías utilizando PROBING en la Maquina 1.

Carga de Catálogo CHAINING

Factor de Carga (CHAINING)	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
2.00	68124.32	3465.37
4.00	68124.35	3484.06
8.00	68124.40	3642.17

Tabla 3. Comparación de consumo de datos y tiempo de ejecución para carga de catálogo con el índice por categorías utilizando CHAINING en la Maquina 1.

Graficas

La gráfica generada por los resultados de las pruebas de rendimiento en la **Maquina 1.**

- Comparación de memoria y tiempo de ejecución para PROBING y CHAINING

Maquina 2

Resultados

Carga de Catálogo PROBING

Factor de Carga (PROBING)	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
0.10	66980.13	2430.92
0.50	67003.21	2434.97
0.90	67002.92	2426.02

Tabla 4. Comparación de consumo de datos y tiempo de ejecución para carga de catálogo con el índice por categorías utilizando PROBING en la Maquina 2.

Carga de Catálogo CHAINING

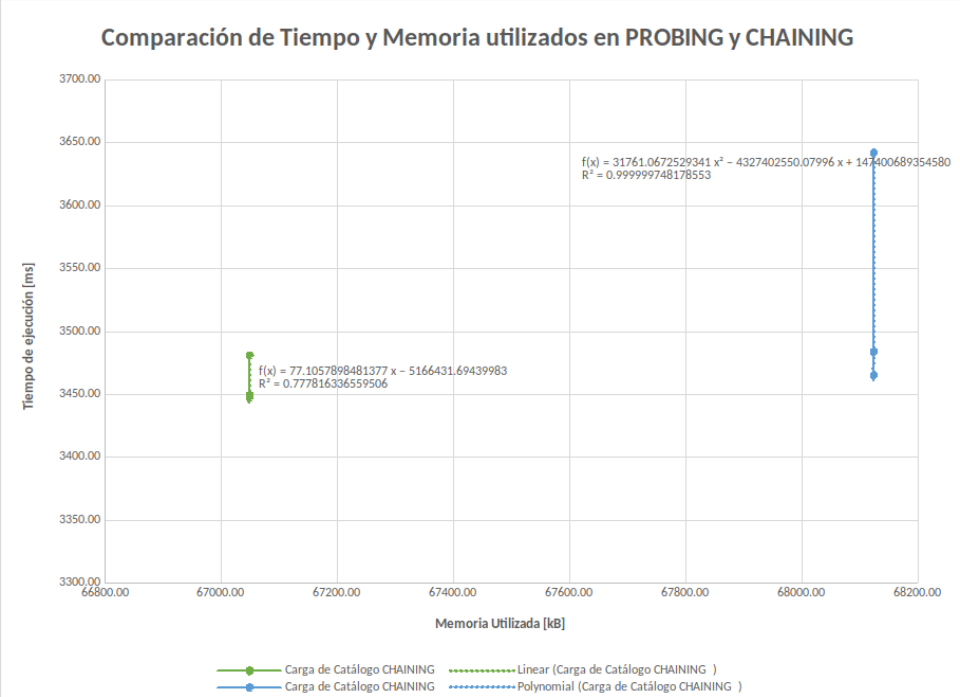
Factor de Carga (CHAINING)	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
2.00	68102.83	2473.34
4.00	68102.84	2474.99
8.00	68101.84	2504.55

Tabla 5. Comparación de consumo de datos y tiempo de ejecución para carga de catálogo con el índice por categorías utilizando CHAINING en la Maquina 2.

Graficas

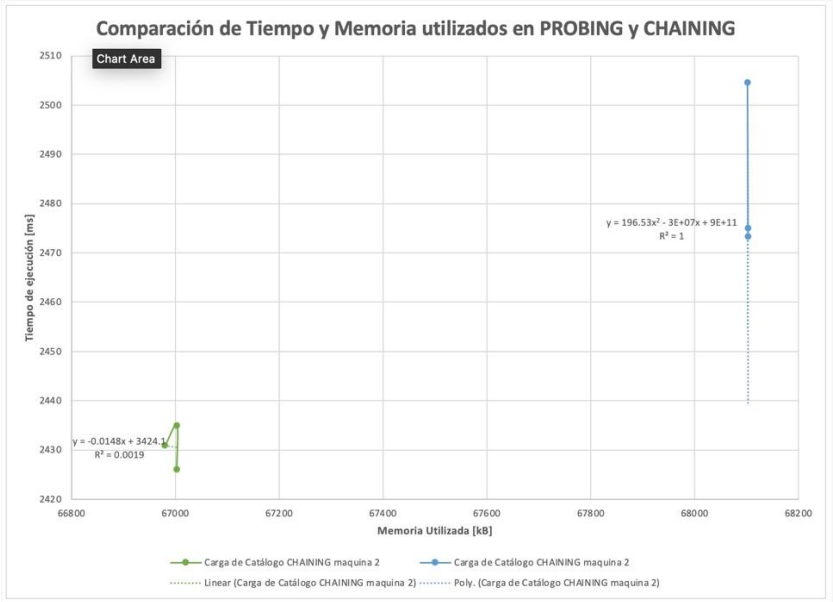
La gráfica generada por los resultados de las pruebas de rendimiento en la **Maquina 1.**

- Comparación de memoria y tiempo de ejecución para PROBING y CHAINING

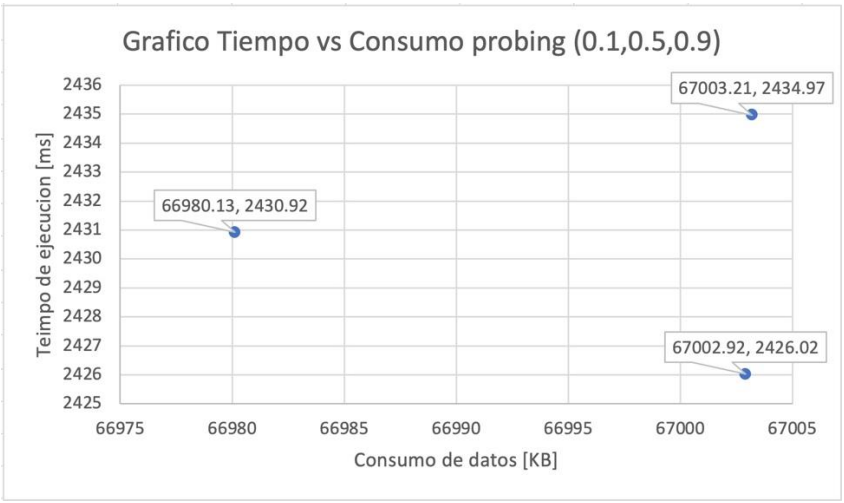


La gráfica generada por los resultados de las pruebas de rendimiento en la **Maquina 2.**

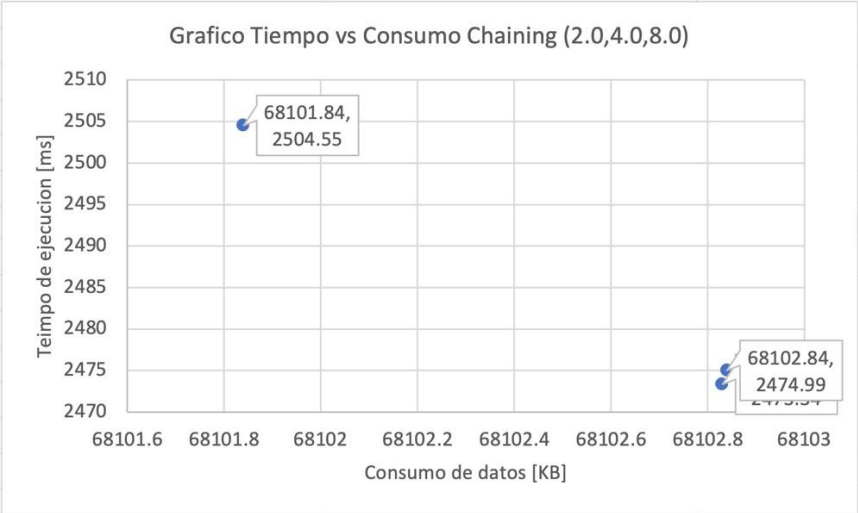
- Comparación de memoria y tiempo de ejecución para PROBING y CHAINING



Grafica de Comparación de memoria y tiempo de ejecución para PROBING y CHAINING para carga de catálogo con el índice por categorías utilizando CHAINING en la Maquina 2.



Grafica de Comparación de memoria y tiempo de ejecución para PROBING para carga de catálogo con el índice por categorías utilizando CHAINING en la Maquina 2.



Grafica de Comparación de memoria y tiempo de ejecución para CHAINING para carga de catálogo con el índice por categorías utilizando CHAINING en la Maquina 2.

Preguntas de análisis

1) ¿Por qué en la función **getTime()** se utiliza **time.perf_counter()** en vez de otras funciones como **time.process_time()**?

Esta función permite obtener la medición más precisa posible al momento de contar el tiempo de cada algoritmo, esto debido a que usa un contador de rendimiento distinto.

Cabe resaltar que la función perf_counter() siempre devuelve el valor flotante del tiempo en segundos. Devuelve el valor (en fracciones de segundo) de un contador de rendimiento, es decir, un reloj con la resolución más alta disponible para medir una duración corta.

2) ¿Por qué son importantes las funciones **start()** y **stop()** de la librería **tracemalloc**?

Estas funciones sirven para indicar a la librería Tracemalloc, cuando debe de empezar a tomar huellas o datos de memoria, y cuando debe de detenerse.

Start se encarga de inicializar el proceso para medir memoria y del mismo modo Stop finaliza el proceso para medir memoria o en el caso del tiempo se encargan respectivamente de iniciar toma de tiempo y memoria al inicio del proceso y finalizar la toma de tiempo y memoria al final del proceso

3) Teniendo en cuenta cada uno de los requerimientos del reto ¿Cuántos índices implementaría en el Reto? y ¿Por qué

Se deben de realizar un indice por año, popularidad, discografia y disrtribucion, ya que estos son los paremetros que se debn usar para clasificar la iinformacion en los Maps.

4) Según los índices propuestos ¿en qué caso usaría **Linear Probing** o **Separate Chaining** en estos índices? y ¿Por qué?

Linear Probing, para los indices que requieren almacenar conjuntos en una misma llave, como por ejemplo año y discografia. Mientras que, para los indices como lo son distribucion y popularidad, sera mejor el Separate Chaining.

5) Dado el número de elementos de los archivos Spotify (large), ¿Cuál sería el factor de carga para estos índices según su mecanismo de colisión?

Para Lienar Probing, el factor de carga es de 0.10 y para el Separate Chaining el factor de carga debe de ser de 4.00.

6) ¿Qué cambios percibe en el **tiempo de ejecución** al modificar el factor de carga máximo para cargar el catálogo de Spotify?

   - Al aumentar el factor de carga tanto en CHAINING como en PROBING se produce un ligero aumento en el tiempo de ejecucion requerido respectivamente.

7) ¿Qué cambios percibe en el **consumo de memoria** al modificar el factor de carga máximo para cargar el catálogo de Spotify?

   - Al aumentar el factor de carga tanto en CHAINING como en PROBING se produce un ligero aumento el la cantidad de consumo de memoria respectivamente.

8) ¿Qué cambios percibe en el **tiempo de ejecución** al modificar el esquema de colisiones?, si los percibe, describa las diferencias y argumente su respuesta.

   - Tomando como evidencia los resultados de las pruebas que adjuntamos en las tablas y graficos anteriores hay un ligero aumento respecto al consumo de TIEMPO DE EJECUCION entre PROBING y CHAINING siendo PROBING el que requiere menor cantidad tiempo en todos sus casos.

9) ¿Qué cambios percibe en el **consumo de memoria** al modificar el esquema de colisiones?, si los percibe, describa las diferencias y argumente su respuesta.

    - Tomando como evidencia los resultados de las pruebas que adjuntamos en las tablas y graficos anteriores hay un ligero aumento respecto al consumo de CONSUMO DE MEMORIA entre PROBING y CHAINING siendo PROBING el que requiere menor cantidad memoria en todos sus casos.

10) ¿Qué configuración ideal de ADT Map escogería para el **índice géneros musicales**?, especifique el mecanismo de colisión, el factor de carga y el numero inicial de elementos.

    - En este caso utiilzamos la siguiente configuracion :

```
catalog['generos'] = mp.newMap(50000, maptype='PROBING', loadfactor=0.9, comparefunction=compare_generes)
```

    - Mecanismo de colisión: Probing (demuestra un mejor desempeno en cuanto a tiempo de ejecucion y uso de memoria).
    - Factor de carga : Inicialmente utilizamos 0.9, sin embargo un factor de carga mejor como lo es 0.10 demuestra un desempeno ligeramente mejor en cuanto a consumo de memoria y tiempo de ejecucion, ademas que brinda una menor probabilidad de que se genere una colision.
    - Numero inicial de elementos: Inicialmente usamos 50000, pero nos dimos cuenta con base al enunciado que el archivo spotify-artists-utf8-large.csv contiene la totalidad de los datos con 56129 registros, por lo que un numero de 60000 seria mejor.
    -