OBSERVACIONES DEL LA PRACTICA

Juanita Ramírez Cod 202121634
Sergio Villamarin Cod XXXX
Estudiante 3 Cod XXXX

	Máquina 1	Máquina 2	Máquina 3
Procesadores	1,4 GHz Intel Core i5 de cuatro núcleos		
Memoria RAM (GB)	8 GB		
Sistema Operativo	Version 13.4.1		

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 Real @LP [ms]
0.1	885.973	16.3
0.5	887.234	16.6
0.7	890.972	18.2
0.9	899.321	23.4

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

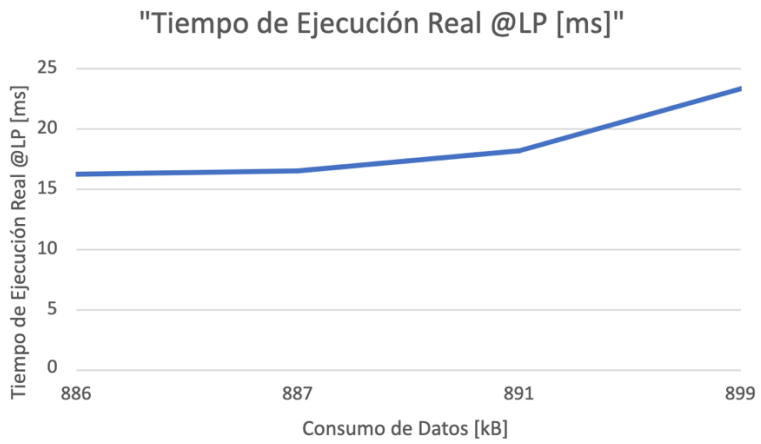
Factor de Carga (CHAINING)	Consumo de Datos [kB]	Tiempo de Ejecución Real @SC [ms]
2.00	885,120	35,705
4.00	888,220	36,101
6.00	887,134	36,100
8.00	888,267	35,705

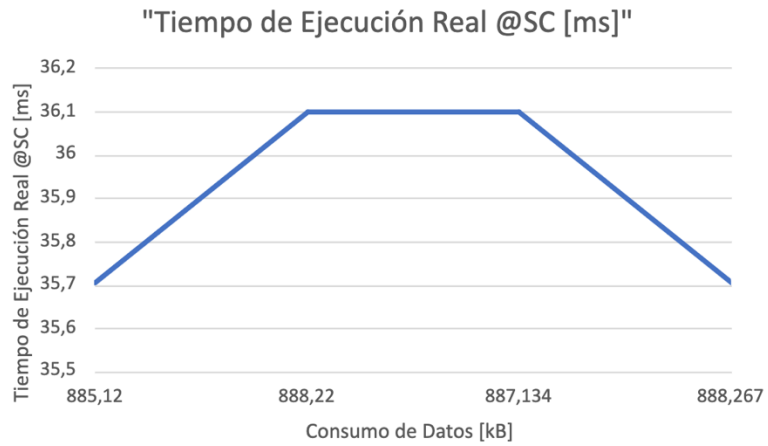
Tabla 3. Comparación de consumo de datos y tiempo de ejecución para carga de catálogo con el índice por categorías utilizando CHAINING en la Maquina 1.

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 Real @LP [ms]
0.1		
0.5		
0.7		
0.9		

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 Real @SC [ms]
2.00		
4.00		

	6.00
	8.00

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 2.**

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

Preguntas de análisis

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

En la función getTime() se utiliza time.perf_counter() en vez de otras funciones como time.process_time(), debido a que esta no solo devuelve el valor flotante del tiempo en segundos sino que tiene en cuenta el tiempo durante el sueño y abarca todo el sistema. En otras palabras, tiene en cuenta el tiempo que hay en el momento en que un hilo se desocupe y entre el tiempo que se le asigna otro trabajo. Además, tiene en cuenta el tiempo de espera de un hilo, es decir, el tiempo cuando un hilo se detiene y tiene que esperar a que otro hilo cumpla con una labor para que este pueda seguir con su trabajo.

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

Las funciones start() y stop() de la librería tracemalloc son importantes debido a que permiten iniciar y finalizar el proceso para medir memoria. Si estos no se utilizaran no se podría establecer cuando comenzar o cuando se deberñia detener esta medición, por lo que seguiría por un tiempo indefinido.

3. ¿Por qué no se puede medir paralelamente el **uso de memoria** y el **tiempo de ejecución** de las operaciones?

No se puede medir paralelamente el uso de memoria y el tiempo de ejecución de las operaciones debido a que se afectan mutuamente, es decir, la medición de la memoria afecta el tiempo de ejecución de las operaciones y viceversa. Lo que causa conflictos en que los resultados sean óptimos.

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

Por el momento, solo se creó un índice. Sin embargo, es muy llamativo crear más indices debido al acceso y busqueda en estos. En el reto crearíamos uno que contenga las fechas, torneos y jugadores.

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

En los tres casos mencionados usaríamos linear probing con el fin de guardar una única pareja, llave-valor, en una sola posición, se usa cuando la carga de la tabla es baja, es eficiente y ocupa menos espacio en memoria. Por otro lado, separate chaining se utiliza cuando la carga de la tabla es alta y tambien es eficiente en la busqueda, aunque se guarda una lista con varios elementos para cada posicion.

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

El factor de carga (fc) se denomina como N/M. Donde N son la cantidad de elementos, y M es el tamaño de la lista que está en cada posición, para separate chaining y M son los espacios que se debe mover para encontrar uno vacío en el caso de linear probing. Para separate chaining el factor de carga esperado es de 4 y para linear probing 0.5.

7. ¿Qué cambios percibe en el **tiempo de ejecución** al modificar el factor de carga máximo para cargar el catálogo de contenido FIFA?

La relacion es proporcional. Entre mayor el factor de carga, mayor es el tiempo necesario para que la carga termine.

8. ¿Qué cambios percibe en el **consumo de memoria** al modificar el factor de carga máximo para cargar el catálogo de contenido FIFA?

El factor de carga máximo influye en la cantidad de memoria utilizada en una estructura de datos, para un mayor factor de carga mas espacio ocupa cargar el catalogo

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

La modificación del esquema de colisiones en una estructura de datos, puede tener un impacto significativo en el tiempo de ejecución, entre menos probabilidad de las colisiones, menor es el tiempo de busqueda. Separate Chaining tiene menor probabilidades de colisiones y Menor tiempo de búsqueda en promedio que linear probing.

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

La modificación del esquema de colisiones en una estructura de datos puede afectar el consumo de memoria, Linear Probing tiende a utilizar menos memoria en comparación con Separate Chaining, ya que no se necesitan estructuras de datos adicionales.

11. ¿Qué configuración de ideal ADT Map escogería para el **índice jugadores FIFA ("scorer")**?, especifique el mecanismo de colisión, el factor de carga y el numero inicial de elementos.

Para Scorer se escogio probing, factor de carga de 0.5 y 44764 elementos.