**OBSERVACIONES DEL LA PRACTICA**

Martín Rincón Cod 201914114

Mariana Ruiz Cod 202011140

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz 2.40 GHz | Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz 1.19 GHz |
| Memoria RAM (GB) | 16GB | 8GB |
| Sistema Operativo | Windows 10 64-bits | Windows 10 Pro 64 bits |

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]** |
| 0.30 | 1307441.18 | 19517.49 |
| 0.50 | 1307437.40 | 19270.42 |
| 0.80 | 1307436.92 | 19199.533 |

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 | 1307498.33 | 20878.18 |
| 4.00 | 1307459.03 | 19938.57 |
| 6.00 | 1307458.96 | 20230.97 |

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]** |
| 0.30 | 1644148,17 | 26909,97 |
| 0.50 | 1644137,55 | 26294,56 |
| 0.80 | 1644137,55 | 26594,36 |

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 | 1644164,39 | 27472,01 |
| 4.00 | 1644175,16 | 27148,54 |
| 6.00 | 1644162,44 | 27470,50 |

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 ves de la previamente conocida **time.process\_time()**?

**getTime()** utiliza **time.perf\_counter()** porque este es el reloj con la resolución más alta disponible para medir una corta duración e incluye el tiempo transcurrido mientras la máquina está en suspensión.

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

**start()** empieza a rastrear las asignaciones de memoria de Python, y stop() detiene el rastreo. Es importante utilizarlas porque sin ellas **take\_snapshot()** no funcionaría.

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

Para los datos de la primera máquina, con el esquema de colisiones Linear Probing, el tiempo de ejecución disminuye al aumentar el factor de carga máximo. Con el esquema de colisiones Separate Chaining, el tiempo de ejecución no sigue una tendencia clara al aumentar el factor de carga máximo.

Algo parecido sucede con los datos de la máquina dos, pues con ningún esquema de colisiones el tiempo sigue una tendencia clara.

El tiempo para linear probing siempre es superior cuando se usa factor de carga de 0.3, ya que este es el único caso en que va a ocurrir un rehash. Para los demás casos las variaciones se pueden justificar por las colisiones que puede arrojar la función de hash de cada ejecución, así como los subprocesos que realiza la máquina.

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

El esquema de colisiones Linear Probing, el consumo de memoria disminuía al aumentar el factor de carga máximo ya que para factores mayores a 0.3 no se ara rehash de modo que el programa no separara más instancias de memoria para su uso. Con el esquema de colisiones Separate Chaining, el consumo de memoria no sigue una tendencia clara; se supone que el consumo aumenta dependiendo de la función de hash, si esta genera más colisiones, entonces llenara más los buckets y esto se traduce en el consumo de más memoria.

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

Para ambas máquinas, el tiempo de ejecución era menor para el esquema de colisiones Linear Probing, incluso cuando se tiene que hacer rehash. Suponemos que las operaciones de insertar en listas, a pesar de ser de tiempo constante, son más costosas en tiempo. Como son tan pocas llaves, el rehash no consume un tiempo muy grande y no supera los tiempos del chaining.

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

Para ambas máquinas, el consumo de memoria era menor para el esquema de colisiones Linear Probing. Esto se justifica en que habiendo tan pocos elementos, las listas que se crean en cada espacio para Chaining pasan a tener cierto peso en comparación a probing, donde la mayoría de los espacios ocupados estarán vacíos.