**OBSERVACIONES DEL LA PRACTICA**

Lindsay Vanessa Pinto Morato Cod 202023138

Maicol Yojan Antonio Rincón Cod 202027329

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10GHZ | Intel Pentium Quad Core Processor N4200 1.10 GHz |
| Memoria RAM (GB) | 8,00GB (6.94 utilizable) | 4,00GB (3.84 Utilizable) |
| Sistema Operativo | Windows 10 Home Single languaje | Windows 10 Pro |

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 | 1911133,29 | 67851,53 |
| 0.50 | 1911133,18 | 67091,59 |
| 0.80 | 1911133,11 | 67083,80 |

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.

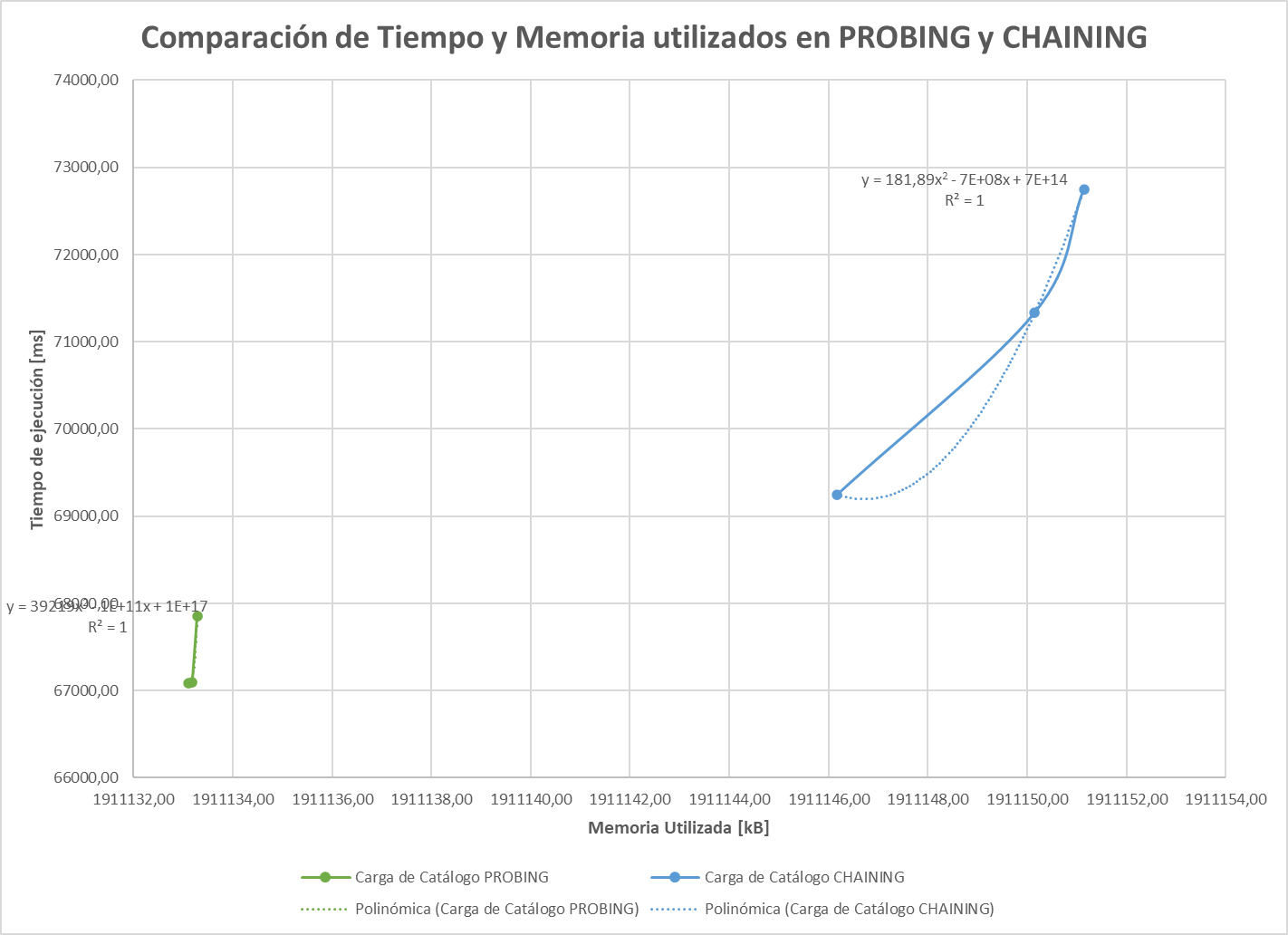
|  |  |  |
| --- | --- | --- |
| ***Carga de Catálogo CHAINING*** | | |
| **Factor de Carga (CHAINING)** | **Consumo de Datos [kB]** | **Tiempo de Ejecución [ms]** |
| 2.00 | 1911146,18 | 69249,65 |
| 4.00 | 1911150,15 | 71331,31 |
| 6.00 | 1911151,15 | 72751,78 |

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**

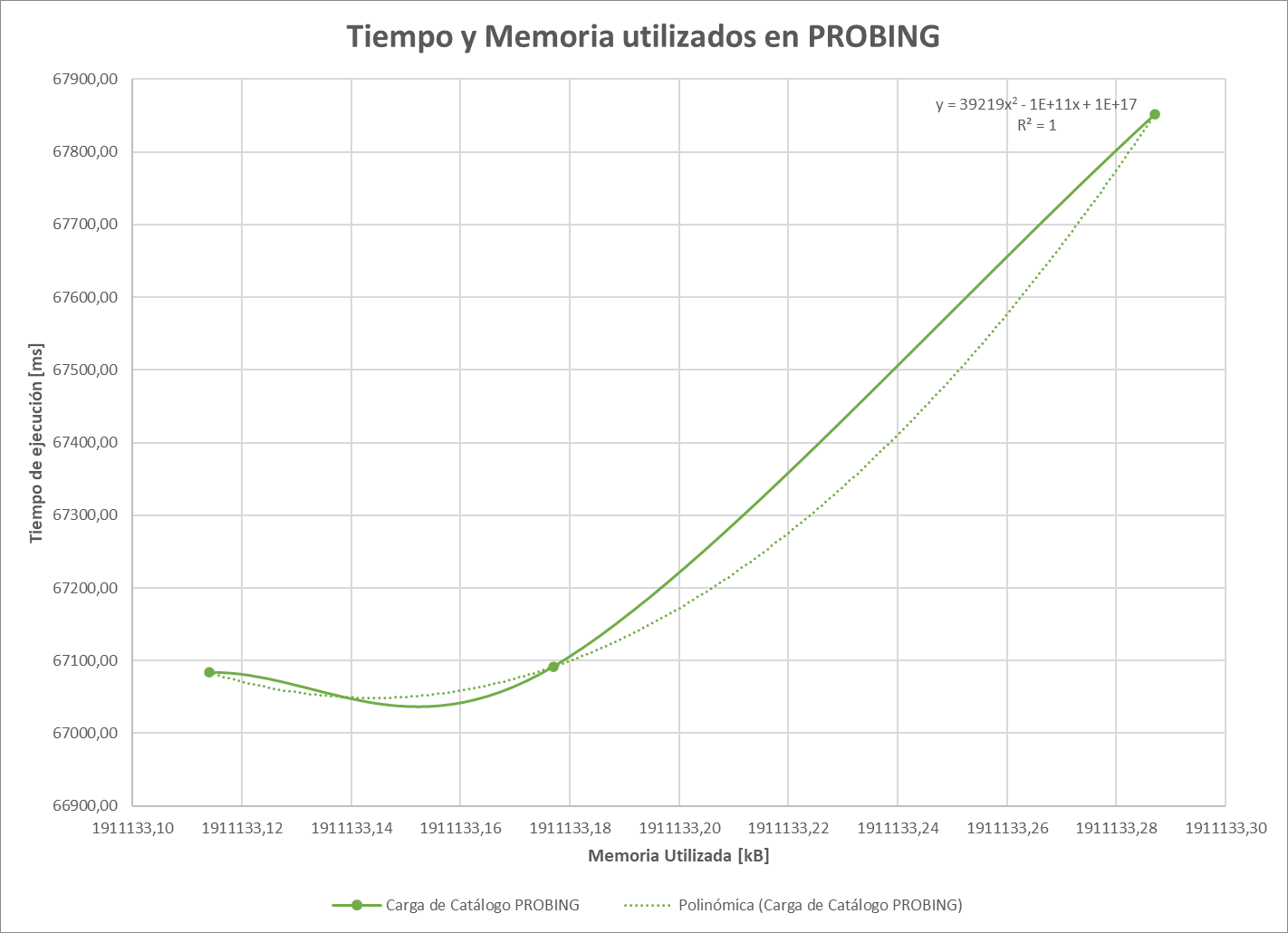
La gráfica generada por los resultados de las pruebas de rendimiento en la **Maquina 1.**

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



La gráfica anterior muestra que para la carga de datos de tipo CHANING tuvo variaciones ligeramente más marcadas tanto en uso de memoria como en tiempo las cuales eran esperadas por los recorridos adicionales que deben hacerse dentro de cada bucket a medida que aumenta el factor de carga.

A continuación, presentamos las gráficas de cada uno por separado para mejor ilustración:



# **Maquina 2**

## **Resultados**

|  |  |  |
| --- | --- | --- |
| ***Carga de Catálogo PROBING*** | | |
| **Factor de Carga** (PROBING) | **Consumo de Datos [kB]** | **Tiempo de Ejecución [ms]** |
| 0.30 | 274023.46 | 139030.59 |
| 0.50 | 274027.10 | 148412.38 |
| 0.80 | 274039.68 | 207820.66 |

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 | 274020.21 | 155861.75 |
| 4.00 | 274021.43 | 174365.69 |
| 6.00 | 274021.39 | 192689.99 |

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

Y las gráficas de Chaining y Probing por separado:

# **Preguntas de análisis**

1. ¿Por qué en la función **getTime()** se utiliza **time.perf\_counter()** en vez de la previamente conocida **time.process\_time()**?

La función process\_time solamente tiene en cuenta el tiempo de ejecución del proceso, mientras que perf\_counter también mide el tiempo utilizado en otras operaciones independientes del proceso. En este sentido el perf\_counter proporciona al usuario una medición más real de acuerdo con el uso que se le dará al código en situaciones normales.

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

La librería tracemalloc es una herramienta diseñada para rastrear los espacios de memoria asignados por Python. Las funciones start() y stop() de este módulo son importantes porque marcan el inicio y el final del análisis del proceso de asignación de memoria realizado por Python durante la ejecución del código.

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

El tiempo de ejecución para el tipo PROBING en el equipo 1 disminuyó a medida que el factor de carga se hizo más grande. Para este caso se realizaron varias pruebas teniendo como resultado datos similares, sin embargo, no consideramos que este resultado se ajuste con la teoría pues con el factor de carga de 0.8 habría necesidad de realizar un Re-Hash lo cual debería traducirse en mayor tiempo de ejecución que no se ve reflejado en los datos tomados de la práctica.

Así mismo, En el equipo 1 para el CHAINING se observa que, a mayor factor de carga, los tiempos se vuelven mayores lo cual si corresponde con el resultado esperado, pues al tener que recorrer no solo las posiciones de la tabla de hash sino que las posiciones de cada uno de los buckets para insertar un nuevo elemento, se vuelve más costosa en tiempo.

Por el contrario, en la Maquina 2 el tiempo de ejecución para el tipo PROBING aumentaba a medida que el factor de carga se hacía mayor, esto sucede igual en el CHAINING, los tiempos de ejecución de este aumentan con forme se aumenta su factor de carga.

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

Para la carga de tipo PROBING se pudo observar en la máquina 1, que la memoria utilizada no tuvo variación significativa pues su cambio fue de máximo 11 décimas lo cual no puede verse como significativo. Esta medición tiene sentido pues siempre se están cargando los mismos datos en un algoritmo cuya mayor modificación es en el aumento de factor de carga.

Sin embargo, en la misma máquina y para la carga de tipo CHAINING, el uso de memoria tuvo una mayor variación la cual se debe a que a medida que aumenta el factor de carga, los bucket dentro de cada una de las posiciones de la tabla de hash aumentan su tamaño y por tanto, el algoritmo debe hacer mayor recorrido traduciéndose en aumento en el uso de memoria.

En la maquina 2 tanto para el PROBING como para el CHAINING no se evidencio un cambio significativo en el consumo de memoria, en algunas de las pruebas el consumo de memoria disminuía y en otros aumentaba para el mismo factor de carga, pero la diferencia era mínima de una a la otra.

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.

Los tiempos de ejecución en general, tanto en la maquina 1 como en la maquina 2, fueron significativamente menores para el esquema de tipo linear PROBING que para el separate CHANING. Esto se debe a que el algoritmo en el esquema PROBING recorre solo las posiciones de la tabla de Hash mientras que en el CHANING debe recorrer adicional a esto los bucket que ya se encuentren creados en cada una de las posiciones de la tabla de hash para insertar un nuevo elemento haciéndolo más costoso en tiempo.

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.

Como se dijo anteriormente, en la máquina 1, se percibió cambio en el uso de memoria al utilizar el esquema de tipo CHAINING pues aumentó a medida que el factor de carga lo hizo. Esto se debe a que el recorrido para insertar un nuevo elemento no solo se hace sobre la tabla de hash buscando la posición de hash determinada, sino que también debe hacerse recorrido sobre cada una de las posiciones del bucket para insertar un nuevo elemento, haciendo esto más costoso en tiempo.

Por su parte, el linear PROBING al realizar solamente comparaciones sobre las posiciones de la tabla de hash, presenta datos de consumo de memoria más estables y con variaciones que no son significativas para el análisis.

Sin embargo, en la maquina 2 en el tipo CHAINING se consumió menos memoria que en el PROBING, y a pesar de cambiar el factor de carga este consumo se mantuvo en el CHAINING pero no en el PROBING