OBSERVACIONES DEL LA PRÁCTICA

Andrés Vargas 202013817
María Alméciga 202023369

	Máquina 1	**Máquina 2**
Procesadores	AMD A10-8700P Radeon R6	AMD Ryzen 5 3500U
Memoria RAM (GB)	16 GB	8 GB
Sistema Operativo	Windows 8.1 Pro	Windows 10

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	263904.536	146841.192
0.50	263904.489	84202.591
0.80	263904.384	58920.24

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

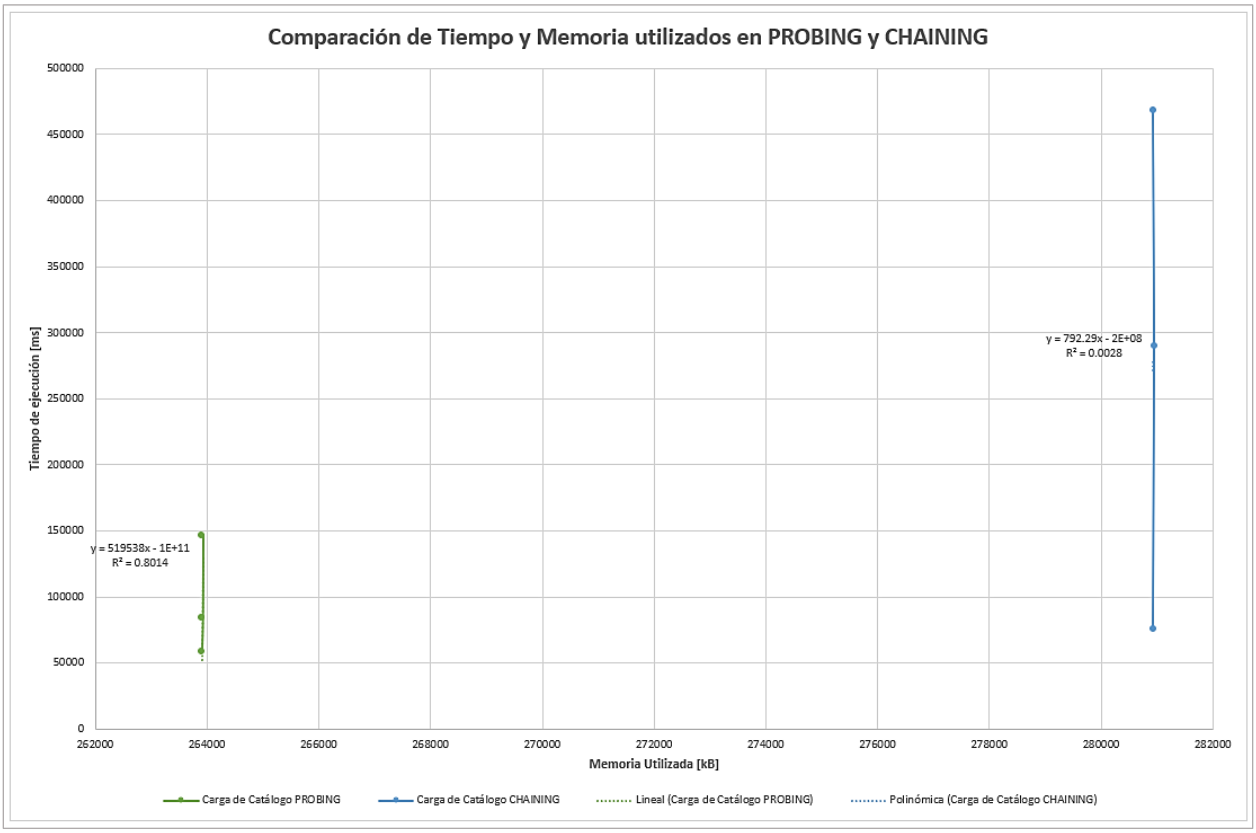
Factor de Carga (CHAINING)	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
2.00	280925.313	468201.791
4.00	280948.248	290073.076
6.00	280925.313	75601.933

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	280947.921	200945.582
0.50	280947.921	122262.808
0.80	280947.921	71536.459

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

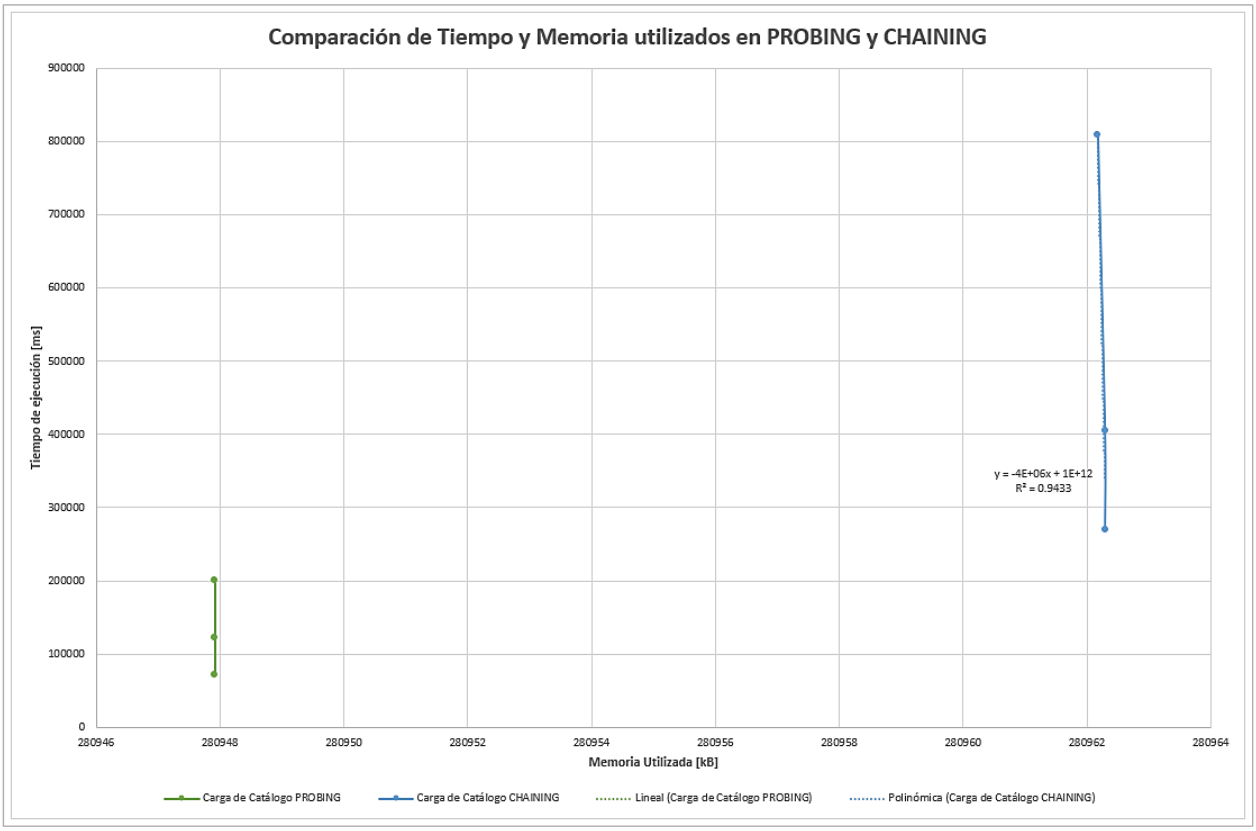
Factor de Carga (CHAINING)	**Consumo de Datos [kB]**	**Tiempo de Ejecución [ms]**
2.00	280962.171	808560.674
4.00	280962.288	404509.364
6.00	280962.288	269578.885

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()**?

Después de investigar, se encuentra que time.perf_counter y time.process_time pueden ayudar a lograr el mismo objetivo de cronometrar un intervalo de tiempo. Sin embargo, se encuentra que la primera función es significativamente mejor que la segunda en aspectos como dar resultados más precisos.

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

La función start() permite que se comience todo el proceso al empezar a rastrear las asignaciones de memoria de Python, mientras que stop() va a permitir finalizar el proceso, al "tomar una captura" de los rastreos antes de ser borrados, y terminar de rastrear las asignaciones de memoria de Python.

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

Se evidencia que el tiempo de ejecución disminuye con el aumento del factor de carga máximo para el catálogo.

Para las pruebas se seleccionó el archivo de videos videos-20pct.csv, y no el archivo large o cualquier otro más grande, pues en principio se evidenció en el medidor de consumo de recursos que el uso la memoria podía acercarse bastante a su capacidad máxima con frecuencia, y en ocasiones el procesador se sobrecargaba por largos periodos de tiempo, con lo que las máquinas se recalentaban bastante con el procesamiento de un mayor volumen de datos. Adicionalmente fue mucho más útil tener todas las pruebas completadas para sacar conclusiones más precisas, en vez de quedar inutilizadas al haber excedido el tiempo límite en las mismas.

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

Se evidencia que el consumo de memoria permanece prácticamente idéntico en todas las pruebas.

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

Se evidencia que el uso del esquema de colisiones Probing disminuye considerablemente el tiempo de ejecución, en comparación a las pruebas con el esquema Chaining. Si bien los factores de carga en las pruebas son distintos, es evidente que con el uso del tipo Probing se obtuvieron resultados mucho más rápidamente.

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

Se evidencia que el consumo de memoria es ligeramente más pequeño en las pruebas con el uso del esquema Probing (por ejemplo, 280947.921 kB para la máquina 2), en comparación a las pruebas en las que se usa el esquema Chaining (280962.249 kB en promedio para la máquina 2). Sin embargo, podría considerarse que no es una diferencia tan significativa en realidad (solamente de 14.328 kB).

Referencias

[1] N. Dunn, «Python Clocks Explained,» *Webducator,* 2015.

[2] Python.org, «tracemalloc — Rastrea la asignación de memoria,» *Python.org Documentation.*