PRIMER TALLER

Presenta: Manuel José Parra Malagón
Instructor: Oscar Alexander Mendez Aguirre
Fecha: 6 de Abril de 2024

I. Introducción

Este taller se centra en el desarrollo de habilidades de perfilado de código, fundamental para la evaluación de la complejidad computacional y temporal de los algoritmos. Siendo un indicador clave de la eficiencia de un algoritmo. A través de este taller, se desarrolló una comprensión profunda de cómo medir y comparar la eficiencia de diferentes algoritmos, utilizando conceptos como la notación Big O para representar la complejidad temporal. Conocimiento que resulta esencial para seleccionar el algoritmo más adecuado para resolver problemas específicos, especialmente en contextos donde se manejan grandes volúmenes de datos, donde las diferencias en la eficiencia pueden traducirse en tiempos de ejecución significativos.

II. Resultados de la práctica

Este taller se desarrolla en dos partes principales, cada una enfocada en aspectos fundamentales del perfilado de código y la medición de la complejidad computacional y temporal.

Puede consultar los resultados del taller en este enlace ***[consultar resultados generales]*** [8] [6] [7]

A partir de los resultados a través de los diferentes métodos se obtiene:

A. *Demostración de* ***Mutabilidad de Datos***

En esta sección, exploramos la mutabilidad de datos a través de varios lenguajes de programación, incluyendo Python, Go, Java y C++. Analizamos cómo cada lenguaje maneja la mutabilidad de los datos y cómo esto afecta la eficiencia y la complejidad de los algoritmos implementados en cada uno.

A partir de los ejercicios para cada lenguaje se obtiene la siguiente tabla:

Tipo de Dato	Lenguaje de Programación			
	Python	*Go Lang*	*Java*	*C++*
int	Mutable	Inmutable	Inmutable	Mutable
float	Mutable	Inmutable	Inmutable	Mutable
str	Inmutable	Inmutable	Inmutable	NA
bool	Mutable	Inmutable	Inmutable	Inmutable
complex	NA	Inmutable	NA	NA
char	NA	NA	Inmutable	Inmutable
double	NA	NA	NA	Inmutable

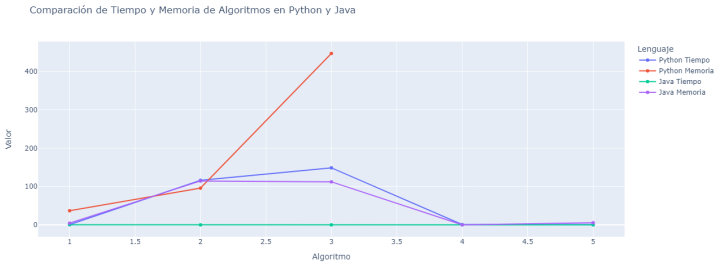
B. ***Profilling*** *en algoritmos en lenguajes de programación*

En esta sección, realizamos el perfilado de memoria y tiempo en cinco algoritmos diferentes utilizando la notación Big O para representar la complejidad temporal. Los lenguajes de programación utilizados para este perfilado incluyen Python y Java, destacando la importancia de la eficiencia en estos lenguajes para la resolución de problemas complejos.

A partir de los ejercicios para cada lenguaje se obtiene la siguiente tabla:

Algoritmo	Lenguaje de Programación	
	Python	*Java*
1	0.811 Sec / 36.8 MB	0.391 Sec / 4.57 MB
2	116.196 Sec / 95.8 MB	0.070438 Sec / 114.47 MB
3	148.746 Sec / 446.9 MB	0.030856 Sec / 112.24 MB
4	0.911 Sec / NA	0.497 Sec / 0.196 MB
5	0.810 Sec /NA	0.214 Sec / 5.65 MB

Usando los datos generados por el experimento se plantea el siguiente gráfico:



Donde se compara los rendimientos de los algoritmos en tiempo y espacio a través de los lenguajes de programación python y java. Es importante recalcar que:

- Para medir los algoritmos en python se utilizó las herramientas de **PyInstrument,** $memory\_profiler$ **y matplotlib para los gráficos generados**, puede consultar el producto de esta labor ***[en este enlace]***
- Para medir los algoritmos en java se utilizó la herramienta de **Profiler** que se encuentra en el **IDE de intellij**, puede consultar el producto de esta labor ***[en este enlace]***

Aclarados estos conceptos, continuaremos a las conclusiones

III. Conclusiones

- Los lenguajes de **Go Lang** y **Java** presentan datos **inmutables**, esto se debe a que estos lenguajes

adoptan un enfoque de **seguridad** y **eficiencia** en la gestión de memoria. [2]

- La **inmutabilidad** de los datos significa que una vez que un objeto es creado, su estado no puede ser modificado. Esto implica que cualquier cambio en el objeto requiere la creación de una nueva instancia, lo cual puede ser beneficioso en términos de **seguridad**, ya que evita efectos secundarios no deseados y facilita el razonamiento sobre el código. [9] [1]
- Además, este enfoque puede mejorar la **eficiencia** en ciertos contextos, como en la programación **concurrente**, donde los objetos **inmutables** pueden ser compartidos entre múltiples hilos sin la necesidad de sincronización adicional. [2]
- **Java** es un lenguaje de programación más **rápido** y con mejor gestión de la **memoria** para ejecutar algoritmos, esto se debe a que incorpora la recogida automática de basura y la gestión de **memoria**, lo que ayuda a mejorar el **rendimiento** de los programas. [3] [4]
- Además, **Java** es **robusto** porque incorpora la gestión de **excepciones** y la recolección de basura, lo que permite detectar y gestionar cualquier **error** inesperado que pueda producirse durante la ejecución del programa, mientras que la recolección de basura libera automáticamente la **memoria** que ya no se necesita. [3] [5]
- Esto, junto con su **arquitectura** diseñada para reducir el gasto de **memoria** durante el tiempo de ejecución, contribuye a su alto **rendimiento** y **eficiencia** en la ejecución de algoritmos. [5]

IV. Referencias

[1] Anon. *Mutable and Immutable Data Types with Golang*. https://golang.ch/mutable-and-immutable-data-types-with-golang/. Accessed on April 6, 2024. 2022.

[2] S. Chakraborty. *Interfaces in Golang - Golang Docs*. https://golangdocs.com/interfaces-in-golang. Accessed on April 6, 2024. 2020.

[3] E. C. Fernández. *¿Para qué sirve la programación Java? — Tokio*. https://www.tokioschool.com/noticias/para-que-sirve-programacion-java/. Accessed on April 6, 2024. 2021.

[4] *Golang vs Java: 6 Key Comparisons*. https://blog.boot.dev/golang/golang-vs-java-go/. Accessed on April 6, 2024. 2021.

[5] *Inmutabilidad: La clave para crear un buen diseño en Java*. https://www.bbvanexttechnologies.com/blogs/inmutabilidad-la-clave-para-crear-un-buen-diseno-en-java/. Accessed on April 6, 2024. 2024.

[6] J. Rickerby. *joerick/pyinstrument*. https://github.com/joerick/pyinstrument. Accessed on April 6, 2024. Apr. 2024.

[7] R. Singh. *Performance Profiling Tools in 2024*. https://www.devopsschool.com/blog/performance-profiling-tools-in-2024/. Accessed on April 6, 2024. Feb. 2024.

[8] The IntelliJ IDEA Blog. *Get Started With Java Profiling in IntelliJ IDEA*. https://blog.jetbrains.com/idea/2021/05/get-started-with-profiling-in-intellij-idea/. Accessed on April 6, 2024. Dec. 2023.

[9] *Which types are mutable and immutable in the Google Go Language?* https://stackoverflow.com/questions/8018081/which-types-are-mutable-and-immutable-in-the-google-go-language. Accessed on April 6, 2024.