

**Instituto Tecnológico y de Estudios
Superiores de Monterrey
Campus Puebla**

Implementación de métodos computacionales (Gpo 1)

Profesor:

Luciano García Bañuelos



**Actividad Integradora 5.3 Resaltador de sintaxis paralelo (evidencia
de competencia)**

Salvador Alejandro Gaytán Ibáñez A01730311

9 de junio del 2021

Para la actividad 5.3 me fue requerido tomar como base el código del resaltador sintáctico de la actividad anterior para poder no solo medir los tiempos de ejecución de este, sino también hacer las modificaciones pertinentes para poder analizar mas de 250 archivos de Python tanto de manera secuencial (la manera normal en la cual lo hacía en la actividad anterior) y de manera concurrente con la finalidad de comparar ambos para poder observar sus diferencias en uso de recursos de la pc (memoria, núcleos, ram, etc) y sus tiempos de ejecución.



Una vez realizadas las modificaciones pertinentes obtuve los siguientes resultados utilizando la dependencia Benchee:

Name	ips	average	deviation	median	99th %
concurrente	0.84	1.18 s	 23.71%	1.16 s	1.64 s
secuencial	0.32	3.16 s	 60.95%	2.14 s	5.37 s

Comparison:

concurrente	0.84
secuencial	0.32 - 2.66x slower +1.97 s

Memory usage statistics:

Name	average	deviation	median	99th %
concurrente	0.30 MB	 0.09%	0.30 MB	0.30 MB
secuencial	269.52 MB	 0.00%	269.52 MB	269.52 MB

Comparison:

concurrente	0.30 MB
secuencial	269.52 MB - 892.64x memory usage +269.22 MB

De los resultados se puede apreciar que, como era lo esperado, la manera concurrente (la cual hace uso de Taskasync como manejador de concurrencia) es no solo más rápida en sus

tiempos de ejecución que la manera secuencial 2.66 (veces para ser exactos) sino también reduce enormemente el uso de memoria (usando 892.64 veces más para la manera secuencial que para la concurrente) lo cual me ayuda a concluir que si se aumentaran el número de archivos Python que mi programa tuviese que analizar se vería una diferencia aun mayor entre los tiempos de ejecución y el uso de memoria entre el método secuencial tradicional y el método concurrente por lo cual llegaría un punto donde mi computadora (Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz y 7.86 GB de ram) se saturaría con la manera secuencial pero estaría sobrado con una ejecución de manera concurrente. Ahora bien, después de un análisis de mi código puedo concluir que la complejidad del mismo es lineal, es decir $O(n)$, sin embargo, que tanto la manera concurrente como la secuencial sean de orden lineal no quiere decir que tengan tiempos de ejecución iguales (como se puede apreciar en las comparaciones anteriores), esto se debe a que la complejidad de la secuencial es de $O(2n)$ y el método concurrente tiene una complejidad de $O(1/4 n)$ por lo que aunque ambas sean lineales los tiempos de ejecución varían drásticamente y se ven resultados mas pronunciados a mayor sea la cantidad de datos a procesar.

Si bien las implicaciones éticas de este trabajo no se ven a simple vista, el simple hecho de comprender la funcionalidad de los programas con ejecución secuencial y concurrente abre las posibilidades para escribir códigos más eficientes que pueden ser utilizados tanto para ayudar al desarrollo tecnológico de la sociedad o para perjudicar el trabajo de colegas informáticos mediante ataques o el famoso “hacking” por lo que depende de cada uno de nosotros el que hacer con dicho conocimiento y, por mi parte, haré todo lo posible para usarlo a favor del beneficio común.