Metodos de ordenamiento en Python comparativa del metodo tradicional y con hilos

Cristhian Urrego Salazar - cristhianu92@gmail.com, Andres Felipe Farfan - anfefarfan@hotmail.com, and heriberto david yepes - HDYCOTECNOVA@gmail.com

I. RESUMEN

Buscando averiguar la eficacia en mtodos de ordenamiento experimentamos los resultados en los algoritmos al implementar hilos al cdigo usando python como lenguaje para subdividir los mtodos en tareas simultaneas con la intencin de averiguar si lograba mejorar su rendimiento.

II. PALABRAS CLAVES

Python, Ordenamiento, InsertionSort, MergeSort, HeapSort, QuickSort, CoutingSort, RadixSort, Github, Hilos

III. RESUMEN EN INGLES

Find the efficiency in ordering methods to experiment the results in the algorithms to implement threads in the code, using python as a language to subdivide the methods in simultaneous tasks with the intention of finding out if it is to improve their performance

IV. PALABRAS CLAVES EN INGLES

Python, Ordering, InsertionSort, MergeSort, Heap-Sort, QuickSort, CoutingSort, RadixSort, Github, Threads

V. INTRODUCCION

Buscando averiguar la eficacia en mtodos de ordenamiento experimentamos los resultados en los algoritmos al implementar hilos al cdigo usando python como lenguaje para subdividir los mtodos en tareas simultaneas con la intencin de averiguar si lograba mejorar su rendimiento.

VI. METODOLOGIA

Al ser un proyecto pequeo se uso programacin extrema como metodologia

VII. ALGORITMOS USADOS

InsertionSort, MergeSort, HeapSort, QuickSort, CoutingSort, RadixSort

VIII. LENGUAJE DE PROGRAMACIN USADO

Python es un lenguaje de programacin interpretado cuya filosofa hace hincapi en una sintaxis que favorezca un cdigo legible.

Se trata de un lenguaje de programacin multiparadigma, ya que soporta orientacin a objetos, programacin imperativa y, en menor medida, programacin funcional. Es un lenguaje interpretado, usa tipado dinmico y es multiplataforma.

Es administrado por la Python Software Foundation. Posee una licencia de cdigo abierto, denominada Python Software Foundation License,2 que es compatible con la Licencia pblica general de GNU a partir de la versin 2.1.1, e incompatible en ciertas versiones anteriores. (Extraido de wikipedia)

IX. CODIGO DE ALGORITMOS IMPLEMENTADO

Repositorio Github

https://github.com/afurourrego/ordenamientos_phyton/blob/master/menu.py

X. RESULTADOS ALCANZADOS

A	lgoritmo - I	InsertionSc	ort NO HILO	OS		Algoritmo	- Insertion	Sort HILOS	
	Comp	olejidad = C)(n^2)			Comp	olejidad = C)(n^2)	
Número de datos		Tiempo er	segundos		Número de datos		Tiempo en	segundos	
Millones	Sin imprimir	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo	Millones	Sin imprimir	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo
1	se corrio e	el algoritmo	durante 4	horas sin	1	no se rea	lizo el proc	eso con hil	os ya que
2	r	esultados	terminados	i.	2	se evide	ncia que to	ma mas ti	empo en
5					5	este tipo	de proceso	s de order	amiento.
10					10				
20					20				

https://raw.githubusercontent.com/afurourrego/ ordenamientos_phyton/master/a1_insertionsort. jpg

	Algoritm	o - Merge	NO HILOS			Algoriti	mo - Merg	HILOS	
	Compl	ejidad = O	n log n)			Compl	ejidad = O(n log n)	
Número de datos		Tiempo er	segundos		Número de datos		Tiempo er	segundos	
Millones	Sin Imprimir	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo	Millones	Sin	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo
1	15,46	16,04			1	18,93	19,81		
2	29,82	30,99			2	38,75	40,59		
5	67,01	69,68			5	108,78	113,3		
10	138,73	144,08			10	218,34	227,46		
20	201.63	302.68			20	445.01	487 84		

https://raw.githubusercontent.com/afurourrego/

ordenamientos_phyton/master/a2_merge.jpg

	Algoritmo	- heapSor	NO HILOS		1		Algoritm	no - heapSo	ort HILOS	
	Compl	ejidad = O(n log n)				Comple	ejidad = O(n log n)	
Número de datos		Tiempo er	segundos			Número de datos		Tiempo er	segundos	
Millones	Sin imprimir	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo		Millones	Sin imprimir	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo
1	13,29	13,94				1	21,47	21,99		
2	22,24	23,25				2	45,51	46,51		
5	59,58	62,21				5	110,55	113,15		
10	128,7	133,98				10	228,25	233,38		
20	278,08	289,27				20	480,89	491,45		

https://raw.githubusercontent.com/afurourrego/ordenamientos_phyton/master/a3_heapsort.jpg

	Algoritmo						no - quickSi							Quick	Sort		
	Comple	ejidad = O(n log n)			Compl	ejidad = O(n log n)						Carca	3010		
Número de datos		Tiempo er	segundos		Número de datos		Tiempo er	segundos		SO ON		goritmo -	quickSor	t NO HILOS	—A	goritmo - quick	Sort HILO
Millones	Sin Imprimir	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo	Millones	Sin Imprimir	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo	NUDEN SEGUN	10						
1	7,97	8,42			1	Se corrio	con 1 millo	on de datos	y genero	∯ 20	00 —					_/	
2	17,37	18,45			2	el sigu	iente error	: Recursion	nError:	E .	0	_					
5	55,61	58,22			5	maximur	n recursion	depth exi	eeded in		-	1		2	3	4	5
10	147,87	153,26			10		comp	arison					MIL	ONES DE	DATOS OR	DENADOS	
20	601,77	612,55			20												

https://raw.githubusercontent.com/afurourrego/ordenamientos_phyton/master/a4_quicksort.jpg

- 1	Algoritmo -			s			o - couting		
	Comp	olejidad = ()(n+k)			Com	plejidad = (O(n+k)	
Número de datos		Tiempo er	segundos		Número de datos		Tiempo er	n segundos	
Millones	Sin Imprimir	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo	Millone	Sin	Imprimie ndo Datos	Tiempo esperado	Delta Tiempo
1	0,83	1,34			1	46,84	47,73		
2	1,58	2,59			2	90,89	92,76		
5	3,97	6,61			5	226,51	231,1		
10	7,94	13,41			10	454,35	463,26		
20	16.07	27.21			20	908.45	926.79		

https://raw.githubusercontent.com/afurourrego/ordenamientos_phyton/master/a5_coutingsort.jpg

	Algoritmo	- radixSor	NO HILOS			Algoritn	10 - radixSe	ort HILOS	
	Com	plejidad =	O(nk)			Com	plejidad = I	O(nk)	
Número de datos		Tiempo er	n segundos		Número de datos		Tiempo er	n segundos	
de datos			_		de datos			_	_
Millones	Sin	Imprimie ndo	Tlempo	Delta	Millones	Sin	Imprimie ndo	Tlempo	Delta
Millottes	imprimir	Datos	esperado	Tiempo	Millones	imprimir	Datos	esperado	Tiempo
1	2,74	3,22			1	12,23	11,08		
2	5,61	6,51			2	20,53	22,27		
5	14,57	16,96			5	51,41	55,76		
10	28,76	33,49			10	102,37	110,91		
20	59.82	69.51			20	204.83	221.86		

https://raw.githubusercontent.com/afurourrego/ordenamientos_phyton/master/a6_radixsort.jpg

ordenamientos_phyton/master/a0_comparativa.jpg

XII. CONCLUSIONES

- El mtodo de insertion Sort es obsoleto para ordenamientos en millones de datos.
- El Mejor m
todo m
s eficaz en este tipo de ordenamientos es Couting Sort.
 $\,$
- La aplicacin de Threads para este caso no hace el trabajo ms eficiente.

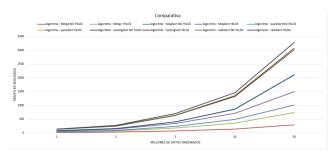
XIII. RECOMENDACIONES

-Para ordenamiento de n
meros se recomienda usar Couting Sort al ser el mas eficiente.

XIV. COMPUTADOR USADO

- -Macbook Pro 2014
- -Procesador 2.6 GHz Intel Core i5
- -Memoria $8~\mathrm{GB}~1600~\mathrm{MHz}~\mathrm{DDR3}$
- -Graficos Intel Iris 1536 MB

XI. EVALUACION DE LOS RESULTADOS DE LOS ALGORITMOS IMPLEMENTADOS



https://raw.githubusercontent.com/afurourrego/

XV. REFERENCIAS BIBLIOGRAFICAS

Python, 2019, https://es.wikipedia.org/wiki/Python