Universidad Simón Bolívar
Reporte de laboratorio semana 5

Análisis de algoritmos de ordenamiento

Autor:

Miguel Salomon
19-10274

Profesor:

Guillermo Palma

Caracas, 11 de junio 2023

Reporte

El siguiente estudio experimental consiste en correr un conjunto de algoritmos de ordenamiento sobre cuatro secuencias de diferentes tamaños (1.000.000, 2.000.000, 3.000.000, 4.000.000 de números enteros), midiendo sus tiempos de ejecución.

Los siguientes son los detalles de la máquina y el entorno donde se ejecutaron los algoritmos:

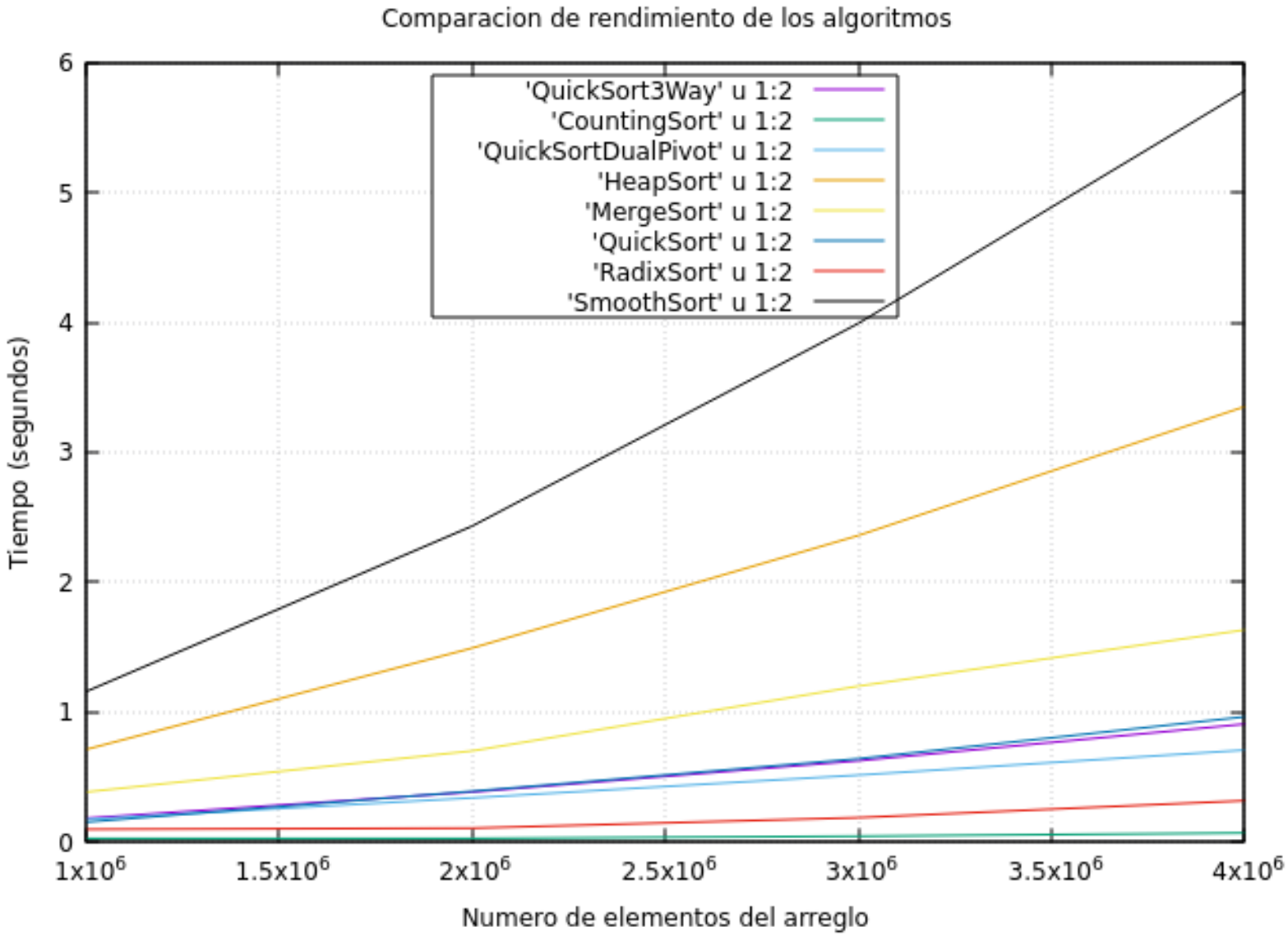
-Sistema operativo: Ubuntu 22.04.
-Procesador: AMD Ryzen 5 5600H 3.30GHz.
-Memoria RAM: 16 GB
-Compilador: kotlinc-jvm 1.8.21 (JRE 17.0.6+10).
-Entorno de ejecución: OpenJDK 64-bit Server VM Temurin-17.0.6+10

Los resultados fueron los siguientes:

Algoritmo	n = 1.000.000	n = 2.000.000	n = 3.000.000	n = 4.000.000
mergeSortInsertion	0.38815774 (± 0.0273251)	0.70364824 (± 0.10653108)	1.2026161033 (± 0.1195118488)	1.634160485 (± 0.092004621)
heapSort	0.715100028 (± 0.10679624)	1.49560390 (± 0.03012478)	2.3599770356 (± 0.0741010008)	3.3509659336 (± 0.061335053)
smoothSort	1.160643568 (± 0.100863968)	2.43393728 (± 0.03757576)	3.9916255753 (± 0.0421331035)	5.7775047433 (± 0.101559891)
quicksortClasico	0.156375704 (± 0.005519378)	0.39355501 (± 0.03235337)	0.6428253296 (± 0.0421331035)	0.9665774176 (± 0.041264208)
quicksortThreeWay	0.185481770 (± 0.026678258)	0.38762797 (± 0.04390637)	0.627443292 (± 0.0352223127)	0.9110659753 (± 0.021836690)
quickSortDualPivot	0.177948212 (± 0.039912723)	0.34256326 (± 0.04558624)	0.517388899 (± 0.0352355473)	0.7096703566 (± 0.048912084)
countingSort	0.031754329 (± 0.003359204)	0.03084891 (± 0.00234593)	0.0487172986 (± 0.0017527372)	0.0731501833 (± 0.013840230)
radixSort	0.101109075 (± 0.049944080)	0.10937471 (± 0.00562493)	0.1900180533 (± 0.0411548337)	0.3215622926 (± 0.084269454)

Todos estos resultados están en segundos.

Comparación de rendimiento:



Como se puede observar, se obtienen de estos algoritmos el comportamiento esperado según la teoría, la cual indica una diferencia notoria en la práctica entre los órdenes de crecimiento de los algoritmos involucrados en la gráfica. El más lento de estos es SmoothSort seguido de HeapSort.

Asimismo se observa, un orden de crecimiento por debajo, los algoritmos de Counting Sort y Radix Sort, en semejanza con la teoría, dado que tienen un tiempo de ejecución en el peor de los casos de O(n + k) y O(d(n + k)) respectivamente, donde k es el mayor número presente en la secuencia y d es la cantidad de dígitos del número más grande de la secuencia.

El algoritmo más rápido es Counting Sort y esto se debe a que, a pesar de tener que crear un arreglo de tamaño aproximadamente igual a n para ejecutar el algoritmo, lo hace solo una vez, y en cambio Radix Sort ordena la secuencia de 5 a 6 veces, lo que hace que los factores constantes involucrados sean más elevados y obtenga un tiempo en este caso ligeramente mayor.

Podemos concluir que los algoritmos de Counting Sort y Radix Sort (y en general, los algoritmos de ordenamiento en tiempo lineal) son notablemente más eficientes para ordenar secuencias de enteros no negativos ya que no se basan en comparaciones.