OBSERVACIONES DE LA PRACTICA

Estudiante 1: Juan José Sierra Alarcón Cod: 202013642

Estudiante 2: Eduardo José Herrera Alba Cod: 201912865

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz 1.60 GHZ | Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz 1.99 GHz |
| Memoria RAM (GB) | 4.00 GB | 12.00 GB |
| Sistema Operativo | Windows 10 Pro 64-bits | Windows 10 Pro 64-bits |

Tabla 1. Especificaciones de las máquinas para ejecutar las pruebas de rendimiento.

# **Maquina 1**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 3593.75 | 4296.87 | 156.25 | 250.00 | 234.375 |
| 2000 | 14796.77 | 17500.00 | 406.25 | 406.25 | 500.00 |
| 4000 | 63125.00 | 72500.00 | 1156.25 | 1343.75 | 1015.62 |
| 8000 | 247937.50 | 286609 | 2890.62 | 2906.25 | 2048.87 |
| 16000 | Se demora más de 15min | Se demora más de 15min | 6093.75 | 5312.5 | 5031.25 |
| 32000 | ‘’ | ‘’ | 14015.62 | 11236.75 | 10467.25 |
| 64000 | ‘’ | ‘’ | 34031.25 | 22375.00 | 23531.25 |
| 128000 | ‘’ | ‘’ | 80562.50 | 49406.25 | 46265.62 |
| 256000 | ‘’ | ‘’ | 220968.75 | 106000.00 | 100875.00 |
| 512000 | ‘’ | ‘’ | Se demora más de 15min | 225429.75 | 210713.25 |

Tabla 2. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación arreglo.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (LINKED\_LIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 247578.12 | 233609.37 | 12781.25 | 3875.00 | 1984.37 |
| 2000 | Se demora más de 15min | Se demora más de 15min | 55250.00 | 19734.37 | 7937.50 |
| 4000 | ‘’ | ‘’ | 260859.42 | 83062.50 | 27281.25 |
| 8000 | ‘’ | ‘’ | Se demora más de 15min | Se demora más de 15min | 118218.75 |
| 16000 | ‘’ | ‘’ | ‘’ | Se demora más de 15min | 461718.75 |
| 32000 | ‘’ | ‘’ | ‘’ | Se demora más de 15min | Se demora más de 15min |
| 64000 | ‘’ | ‘’ | ‘’ | Se demora más de 15min | Se demora más de 15min |
| 128000 | ‘’ | ‘’ | ‘’ | Se demora más de 15min | Se demora más de 15min |
| 256000 | ‘’ | ‘’ | ‘’ | Se demora más de 15min | Se demora más de 15min |
| 512000 | ‘’ | ‘’ | ‘’ | Se demora más de 15min | Se demora más de 15min |

Tabla 3. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación lista enlazada.

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Merge sort | Muy eficaz | Menos eficaz |
| Quick sort | Muy eficaz | Poco eficaz, pero mejor que Merge |

Tabla 4. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

## **Graficas**

* Cinco gráficas generadas por los resultados de las pruebas de rendimiento en la **Maquina 1.**
  + Comparación de rendimiento ARRAYLIST.
  + Comparación de rendimiento LINKED\_LIST.
  + Comparación de rendimiento para Insertion Sort.
  + Comparación de rendimiento para Selection Sort.
  + Comparación de rendimiento para Shell Sort.
  + Comparación de rendimiento para MergeSort.
  + Comparación de rendimiento para QuickSort.

# **Maquina 2**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 593.75 | 750 | 31.25 | 31.25 | 15.62 |
| 2000 | 2531.25 | 3109.37 | 93.75 | 62.50 | 62.50 |
| 4000 | 11359.37 | 13875.00 | 203.12 | 109.37 | 109.37 |
| 8000 | 48125.00 | 58671.87 | 484.37 | 281.25 | 234.37 |
| 16000 | 199093.75 | 245515.62 | 1093.75 | 515.62 | 468.75 |
| 32000 | 771515.62 | Se demora más de 15min | 2437.50 | 1109.37 | 1062.50 |
| 64000 | Se demora más de 15min | ‘’ | 5765.62 | 2375.00 | 2343.75 |
| 128000 | ‘’ | ‘’ | 14234.37 | 5234.37 | 5000.00 |
| 256000 | ‘’ | ‘’ | 38046.87 | 11718.75 | 10859.37 |
| 512000 | ‘’ | ‘’ | Se demora más de 15min | 245920.37 | 23947.25 |

Tabla 2. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación arreglo.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (LINKED\_LIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 41093.75 | 37328.12 | 2046.87 | 1343.75 | 171.87 |
| 2000 | 363593.75 | 333796.87 | 9734.37 | 6500.00 | 703.12 |
| 4000 | Se demora más de 15min | Se demora más de 15min | 44656.25 | 30187.50 | 2796.87 |
| 8000 | ‘’ | ‘’ | 243937.50 | 165046.875 | 11734.37 |
| 16000 | ‘’ | ‘’ | Se demora más de 15min | 604234.37 | 47671.87 |
| 32000 | ‘’ | ‘’ | ‘’ | Se demora más de 15min | 202218.75 |
| 64000 | ‘’ | ‘’ | ‘’ | ‘’ | Se demora más de 15min |
| 128000 | ‘’ | ‘’ | ‘’ | ‘’ | ‘’ |
| 256000 | ‘’ | ‘’ | ‘’ | ‘’ | ‘’ |
| 512000 | ‘’ | ‘’ | ‘’ | ‘’ | ‘’ |

Tabla 3. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación lista enlazada.

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Merge sort | Muy eficaz | Menos eficaz |
| Quick sort | Muy eficaz | Poco eficaz, pero mejor que Merge |

Tabla 4. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

## **Graficas**

* Cinco gráficas generadas por los resultados de las pruebas de rendimiento en la **Maquina 2.**
  + Comparación de rendimiento ARRAYLIST.
  + Comparación de rendimiento LINKED\_LIST.
  + Comparación de rendimiento para Insertion Sort.
  + Comparación de rendimiento para Selection Sort.
  + Comparación de rendimiento para Shell Sort.
  + Comparación de rendimiento para MergeSort.
  + Comparación de rendimiento para QuickSort.

# **Preguntas de análisis**

1. ¿El comportamiento de los algoritmos es acorde a lo enunciado teóricamente?
2. ¿Existe alguna diferencia entre los resultados obtenidos al ejecutar las pruebas en diferentes máquinas?
3. De existir diferencias, ¿A qué creen ustedes que se deben dichas diferencias?
4. ¿Cuál Estructura de Datos es mejor utilizar si solo se tiene en cuenta los tiempos de ejecución de los algoritmos?
5. Para el caso analizado de ordenamiento de los videos, teniendo en cuenta los resultados de tiempo reportados por todos los algoritmos de ordenamiento estudiados (iterativos y recursivos), proponga un ranking de los algoritmos de ordenamiento (de mayor eficiencia a menor eficiencia en tiempo) para ordenar la mayor cantidad de videos.

# **Gráficas Máquina 1**

En el documento xcx adjunto. No se graficó correctamente la información de la tabla. Para los datos de Insertion, Selection, Shell, Quick de la segunda tabla, no hubiera sido posible hacer ninguna aproximación, ya que los datos son insuficientes (en dos casos solo hay un dato, y en otros dos casos solo hay tres datos). Esto es debido a la mayor complejidad de estos ordenamientos. En su contraparte, sí hubiera sido posible graficar Merge Sort, debido a su menor complejidad (O(n log n))

# **Gráficas Máquina 2**

En el documento xcx adjunto. No se graficó correctamente la información de la tabla. Para los datos de Insertion y Selection de la segunda tabla, no hubiera sido posible hacer ninguna aproximación, ya que los datos son insuficientes (solo hay dos datos). Esto es debido a la mayor complejidad (O(n2)) de estos ordenamientos. En este caso, hubiera sido posible graficar Shell, Quick y Merge Sort, debido a su menor complejidad (O(n log n)). Como se explica más adelante, la diferencia con los resultados del Estudiante 1 se debe a que la Máquina 2 es más potente y al momento de realizar las pruebas estaba ejecutando menos operaciones al mismo tiempo.

# **Respuestas a Preguntas de análisis**

1) Sí: las funciones de ordenamiento reflejan los tiempos establecidos teóricamente. Esto es, Shell Sort es la función más rápido organizando los datos (su complejidad es O(n1.5)), seguida de Insertion Sort (O(n2)) y por último Selection Sort (O(n2)). Por otro lado, estas estas funciones son radicalmente más eficientes en los Arreglos que en las Listas Enlazadas, ya que fueron concebidas para esta Estructura de Datos. Por otro lado, la funciones Quick Sort O(n2) y Merge Sort es O(n log(n)) son ligeramente más efectivas que Shell Sort en el Arreglo, pero notoriamente mejores que Shell Sort en la Lista Enlazada.

2) Sí, la diferencia es abismalmente notoria, en cuanto una máquina es mucho más rápida que la otra a la hora de cargar y ordenar los datos.

3) Esta diferencia radica en diferentes factores: otras acciones o programas que se estén ejecutando al tiempo, o los componentes de cada máquina (tales como memoria RAM, modelo del procesador y frecuencia del procesador) que tienen diferente magnitud, lo que provoca un desequilibrio a la hora de cargar y ordenar los datos.

4) Teniendo en cuenta las funciones de ordenamiento testeados (Shell, Insertion, Selection, Merge y Quick), la mejor Estructura de Datos a utilizar entre Arreglo y Lista Enlazada a utilizar es el Arreglo.

5)

1. Merge Sort
2. Quick Sort
3. Shell Sort
4. Insertion Sort
5. Selection Sort