OBSERVACIONES DE LA PRACTICA

Estudiante 1: José Vicenet Vargas Panesso Cod: 201815601

Estudiante 2: Daniel Reales Cod 201822265

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | Intel core i7-7700HQ | Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz, 1800 Mhz, 4 Core(s), 8 Logical Processor(s) |
| Memoria RAM (GB) | 16 | 8.00 |
| Sistema Operativo | Windows 10 | Windows 10 |

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

# **Maquina 1**

## **Resultados**

|  |  |  |  |
| --- | --- | --- | --- |
| Tamaño de la muestra (ARRAY\_LIST) | Insertion Sort (ms) | Selection Sort (ms) | Shell Sort (ms) |
| 1000 | 524.21005 | 634.3467500 | 27.5947 |
| 2000 | 2217.25375 | 2493.5002 | 70.29945 |
| 4000 | 9220.94995 | 9905.9792 | 157.99705 |
| 8000 | 36895.144 | 40755.8725 | 359.71225 |
| 16000 | 147906.401 | 170314.96485 | 888.40065 |
| 32000 | 624198.79 | 716931.5177 | 2195.2944 |
| 64000 | Más de 10 minutos. | Más de 10 minutos. | 4896.472 |
| 128000 | Más de 10 minutos. | Más de 10 minutos. | 10496.8632 |
| 256000 | Más de 10 minutos. | Más de 10 minutos. | 25239.33525 |
|  |  |  |  |

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) |
| 1000 | 39418.76665 | 37099.9892 | 1836.1011 |
| 2000 | 325156.76415 | 299347.4094 | 9294.735 |
| 4000 | Más de 10 minutos | Más de 10 minutos | 45590.3673 |
| 8000 | Más de 10 minutos | Más de 10 minutos | 209505.9377 |
| 16000 | Más de 10 minutos | Más de 10 minutos | 1033159.49495 |
| 32000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |
| 64000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |
| 128000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |
| 256000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |
| 512000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |

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

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Insertion sort | Más eficiente | Menos eficiente |
| Selection sort | Más eficiente | Menos eficiente |
| Shell sort | Más eficiente | Menos eficiente |

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.

# **Maquina 2**

## **Resultados**

|  |  |  |  |
| --- | --- | --- | --- |
| Tamaño de la muestra (ARRAY\_LIST) | Insertion Sort (ms) | Selection Sort (ms) | Shell Sort (ms) |
| 1000 | 520.6368 | 675.2205 | 25.44955 |
| 2000 | 2086.953 | 2479.4118 | 85.77 |
| 4000 | 9099.181 | 10598.205 | 160.0876 |
| 8000 | 40956.55175 | 42120.887 | 471.7547 |
| 16000 | 152000 | 180500 | 871.2021 |
| 32000 | 639500 | 777500 | 1957.798 |
| 64000 | Más de 10 minutos. | Más de 10 minutos. | 4877.956 |
| 128000 | Más de 10 minutos. | Más de 10 minutos. | 10593.71 |
| 256000 | Más de 10 minutos. | Más de 10 minutos. | 25283.89 |
| 512000 |  |  |  |

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) |
| 1000 | 47528.87 | 35438.24 | 2623.509 |
| 2000 | 326460.5 | 322686.5 | 9726.005 |
| 4000 | Más de 10 minutos | Más de 10 minutos | 44297.59 |
| 8000 | Más de 10 minutos | Más de 10 minutos | 207811.5 |
| 16000 | Más de 10 minutos | Más de 10 minutos | 1097035 |
| 32000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |
| 64000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |
| 128000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |
| 256000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |
| 512000 | Más de 10 minutos | Más de 10 minutos | Más de 10 minutos |

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

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Insertion sort | Más eficiente | Menos eficiente |
| Selection sort | Más eficiente | Menos eficiente |
| Shell sort | Más eficiente | Menos eficiente |

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.

# **Preguntas de análisis**

1. ¿El comportamiento de los algoritmos es acorde a lo enunciado teóricamente?

De forma general podemos ver que los algoritmos si se compartan de la manera esperada teóricamente. En el algoritmo Selection Sort, cuya complejidad temporal es se puede ver un aumento cuadrático en los tiempos de ejecución a medida que aumenta el tamaño de los datos, para ambos tipos de datos, tanto ARRAY\_LIST como LINKED\_LIST. Esto podemos verificarlo utilizando un ajuste cuadrático a la gráfica que compara el algoritmo y su tiempo de ejecución, como se muestra a continuación:

Como se aprecia en el gráfico, el modelo matemático describe con un el comportamiento temporal en milisegundos del algoritmo implementado con una estructura de datos tipo ARRAY\_LIST bajos las condiciones particulares de la Maquina 2 al momento de tomar los datos.

De manera similar, el algoritmo Insertion Sort, aunque mejor relativo al algoritmo Selection Sort, sigue presentando un comportamiento cuadrático. Realizando un ejercicio similar al anterior, vemos que una línea de tendencia polinómica de grado 2 ajusta los datos de una manera cercana bajo el modelo matemático

.

La mejora temporal relativa al algoritmo Selection Sort se puede apreciar también mediante el análisis del coeficiente que acompaña al término cuadrático. En el caso del algoritmo Insertion Sort, este coeficiente es menor.

Finalmente, para el algoritmo Shell Sort, se observa un comportamiento considerablemente mejor en comparación a los dos desempeños analizados anteriormente. Esto es de esperar desde la perspectiva teórica debido a que las complejidades en el peor caso, caso promedio y mejor caso son de , y respectivamente.

1. ¿Existe alguna diferencia entre los resultados obtenidos al ejecutar las pruebas en diferentes máquinas?

Si, existen diferencias menores en los tiempos de ejecución de los algoritmos para las distintas máquinas. En general, la Maquina 2 toma un mayor tiempo para realizar el ordenamiento de los datos independientemente del tamaño de la lista, el algoritmo utilizado o el tipo de Estructura de Datos seleccionada.

1. De existir diferencias, ¿A qué creen ustedes que se deben dichas diferencias?

Las diferencias entre los desempeños observados pueden ser atribuidad principalmente a dos factores. En primer lugar, el hardware que ambas maquinas utilizan para realizar los calculos y el amacenamiento es distinto. Esto lo podemos evidenciar debido a que la Máquina 1 posee un procesador Intel i7 mientras que la Máquina 2 tiene un procesador i5. Más aún, la capacidad de memoria principal (RAM), en la primera máquina es el doble que en la otra. El segundo conjunto de factores que pueden contribuir a las diferencias observadas en rendimiento son las condiciones particulares de ejecución. Es decir, incluso si las Máquinas fueran identicas en el hardware, si existen diferencias en la cantidad de software que estan ejecutando al momento de realizar los ordenamientos, entonces se observaran desempeños diferentes.

1. ¿Cuál Estructura de Datos es mejor utilizar si solo se tiene en cuenta los tiempos de ejecución de los algoritmos?

La mejor estructura para todos los algoritmos es la estructura ARRAY\_LIST. Esto se debe al funcionamiento interno de cada uno de los algoritmos. Ya que todos utilizan la función getElement(), disponible en la API del TAD Lista, la complejidad de esta instrucción cuando se utiliza un ARRAY \_LIST es O(1) mientras que para la SINGLE\_LINKED es O(n). Esto implica que al evaluar la complejidad del algoritmo no es necesario tener solo en cuenta su complejidad intrínseca, sino también la complejidad de esta función, la cual es claramente superior en una estructura de datos tipo ARRAY\_LIST.