OBSERVACIONES DE LA PRACTICA

Daniel Felipe Vargas Cod 202123899

Santiago Martínez Cod 202124032

### Máquina 1 Máquina 2

|  |  |  |
| --- | --- | --- |
| **Procesadores** | Intel Celeron N3160 | 11th Gen Intel(R) Core(TM) i5-1135G7 |
| **Memoria RAM**  **(GB)** | 1,889 GB | 16,0 GB |
| **Sistema**  **Operativo** | Arch Linux x86\_64 | Microsoft Windows 10 Home Single Language |

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

# Maquina 1

## Resultados

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Porcentaje de la muestra [pct]** | **Tamaño de la muestra (ARRAY\_LIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 0.50% | 501 | 581.60 | 628.97 | 52.14 | 49.55 | 53.62 |
| 5.00% | 4391 | 45929.23 | 50200.80 | 759.91 | 453.08 | 468.43 |
| 10.00% | 8193 | 160861.00 | 174861.06 | 1581.12 | 934.94 | 994.54 |
| 20.00% | 15136 | +600000 | +600000 | 3355.22 | 2071.07 | 1808.17 |
| 30.00% | 21319 | +600000 | +600000 | 4951.66 | 2994.4 | 2656.29 |
| 50.00% | 32422 | +600000 | +600000 | 8559.63 | 5303.86 | 4206.06 |
| 80.00% | 47165 | +600000 | +600000 | 12664.83 | 9262.64 | 6467.17 |
| 100.00% | 56129 | +600000 | +600000 | 15466.12 | 13840.98 | 7881.8 |

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Porcentaje de la muestra [pct]** | **Tamaño de la muestra (SINGLE\_LINKED)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 0.50% | 501 | 19624.32 | 17846.38 | 1471.03 | 1314.54 | 231.67 |
| 5.00% | 4391 | +600000 | +600000 | 271055.94 | 168483.94 | 16120.69 |
| 10.00% | 8193 | +600000 | +600000 | +600000 | +600000 | +600000 |
| 20.00% | 15136 | +600000 | +600000 | +600000 | +600000 | +600000 |
| 30.00% | 21319 | +600000 | +600000 | +600000 | +600000 | +600000 |
| 50.00% | 32422 | +600000 | +600000 | +600000 | +600000 | +600000 |
| 80.00% | 47165 | +600000 | +600000 | +600000 | +600000 | +600000 |
| 100.00% | 56129 | +600000 | +600000 | +600000 | +600000 | +600000 |

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

### Algoritmo Arreglo (ARRAYLIST) Lista enlazada

**(LINKED\_LIST)**

|  |  |  |
| --- | --- | --- |
| **Merge sort** | X | X |

|  |  |  |
| --- | --- | --- |
| **Quick sort** |  |  |

*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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Porcentaje de la muestra [pct]** | **Tamaño de la muestra (ARRAY\_LIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 0.50% | 501 | 845.96 | 847.82 | 68.40 | 59.74 | 61.44 |
| 5.00% | 4391 | 65180.91 | 65741.79 | 1021.54 | 662.86 | 655.87 |
| 10.00% | 8193 | 231126.72 | 233613.80 | 2148.11 | 1420.45 | 1296.04 |
| 20.00% | 15136 | 807943.07 | 856497.83 | 4489.16 | 3089.70 | 2478.22 |
| 30.00% | 21319 | - | - | 6852.61 | 4419.34 | 3631.60 |
| 50.00% | 32422 | - | - | 11737.12 | 7788.25 | 5691.38 |
| 80.00% | 47165 | - | - | 16859.20 | 13016.29 | 8334.55 |
| 100.00% | 56129 | - | - | 20993.96 | 17966.48 | 10470.42 |

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Porcentaje de la muestra [pct]** | **Tamaño de la muestra (SINGLE\_LINKED)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 0.50% | 501 | 10803.28 | 9719.19 | 324.84 | 717.34 | 159.76 |
| 5.00% | 4391 | +600000 | +600000 | 113302.38 | 71854.44 | 7973.46 |
| 10.00% | 8193 | +600000 | +600000 | 454785.68 | 280618.19 | 27000.43 |
| 20.00% | 15136 | +600000 | +600000 | +600000 | 1143105.16 | 91978.42 |
| 30.00% | 21319 | +600000 | +600000 | +600000 | - | 181299.97 |
| 50.00% | 32422 | +600000 | +600000 | +600000 | - | 44162.33 |
| 80.00% | 47165 | +600000 | +600000 | +600000 | - | +600000 |
| 100.00% | 56129 | +600000 | +600000 | +600000 | - | +600000 |

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

### Algoritmo Arreglo (ARRAYLIST) Lista enlazada

**(LINKED\_LIST)**

|  |  |  |
| --- | --- | --- |
| **Merge sort** | X | X |
| **Quick sort** |  |  |

*Tabla 7. 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.**

**(Gráficas en el Archivo de Excel Adjunto)**

* + 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?

En general, puede verse que los algoritmos recursivos analizados siguen el comportamiento linearítmico que predice la teoría. Sin embargo, cabe destacar que el algoritmo de Quicksort fue más lento que Merge en todas nuestras pruebas, a pesar de que la teoría sustente que Merge debe ser un poco más lento por tener un coeficiente de 6 frente al coeficiente de 1.3 de Quicksort. En términos de los algoritmos iterativos, encontramos que estos se acogen bastante bien a la teoría, pues Insertion y Selection tienen una complejidad cercana a la cuadrada, mientras que Shell tiene una complejidad cercana a N^3/2. Pudimos encontrar esto al hacer una regresión sobre las curvas generadas por el tiempo que tardó cada ordenamiento en terminar, con lo cual pudimos ver que una función cuadrática se ajusta mejor a la gráfica de complejidad del insertion y selection sort (aunque no tuvimos la oportunidad de verificarlo también para la estructura SINGLE\_LINKED, ya que solo obtuvimos un dato con estos ordenamientos). En el caso del ordenamiento de tipo Shell, encontramos que probablemente no encontramos suficientes datos para mostrar la naturaleza entre linearítmica y N^3/2 que posee la complejidad de este ordenamiento, lo cual resultó en Excel frecuentemente prefiriendo una regresión lineal o polinomial (de grado 2) para ajustarse mejor a la gráfica de complejidad del Shell sort. Sin embargo, a pesar de esta situación, todavía es evidente como la complejidad de este ordenamiento no es lineal, y al realizar un ajuste del tipo x log(x) (para copiar la complejidad teórica del Shell sort de n log (n)), encontramos que esta también se ajusta casi perfectamente a la gráfica.

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

Si bien ejecutar las pruebas en una mejor maquina con más capacidad de procesamiento nos permite obtener más datos y por lo tanto realizar mejores aproximaciones, la complejidad de los algoritmos se mantiene en diferentes maquinas como es de esperar, independiente de su capacidad de procesamiento. Es aparente que ejecutar las pruebas en máquinas diferentes resultará en tiempos diferentes, pero lo importante es la relación entre los tiempos y el tamaño de muestra que nos permite determinar la complejidad de un algoritmo independientemente de la maquina donde se probó.

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

Como se señaló previamente, dichas diferencias se deben a las diferencias en capacidad de procesamiento que tienen diferentes maquinas. Principalmente, encontramos que lo que más impacta a la duración de un ordenamiento es la velocidad del reloj de la CPU con la cual se está ejecutando el algoritmo. Esto tiene sentido, ya que un a CPU que pueda realizas más ciclos por segundo podrá simplemente realizar más tareas en una misma cantidad de tiempo, en este caso realizar más comparaciones cada segundo, lo cual lógicamente reducirá el tiempo que tarda en ordenarse la totalidad de los datos.

Adicionalmente, la cantidad de memoria RAM disponible en cada computador tuvo un efecto visible, ya que en computadores con memora RAM más limitada en ocasiones era imposible completar ciertos ordenamientos, lo cual no era un problema en máquinas con más memoria RAM disponible que podían completar el ordenamiento sin quedarse sin memoria.

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

Si sólo se tiene en cuenta el tiempo de ejecución, es evidente que, la mejor estructura de datos a utilizar es ARRAY\_LIST. Podemos sustentar esta respuesta con nuestros resultados experimentales, ya que como se registró en las tablas 2 y 3, la estructura ARRAY\_LIST tuvo un mejor tiempo de ejecución a SINGLE\_LINKED para todos los algoritmos cuando se utilizó un tamaño de muestra idéntico.

1. Para el caso analizado de ordenamiento de los artistas, teniendo en cuenta los resultados de tiempo reportados por todos los algoritmos de ordenamiento (iterativos y recursivos), proponga un ranking de los algoritmos de ordenamiento (de mayor eficiencia a menor (en tiempos de ejecución) para ordenar la mayor cantidad de artistas.

Tanto para ARRAY\_LIST como en SINGLE\_LINKED el siguiente ranking es válido para la máquina 1 y la 2:

1. Merge sort
2. Quicksort
3. Shell sort
4. Selection sort
5. Insertion sort