Análisis de Factores influyentes en el tiempo de ejecución en Dos Algoritmos de Ordenamiento

Cristhian Castillo, Sebastian Correa, Camilo Sepulveda, Jorge Lievano
Departamento de Tecnologías de Información y Comunicaciones
Universidad Icesi
Email: cristhian.castillo1@correo.icesi.edu.co, sebastian.correa1@correo.icesi.edu.co, johncamilo.sepulveda@hotmail.com

Abstract

En el siguiente documento se desarrollará un informe sobre un diseño de experimentos para estudiar, medir y cuantificar los efectos que tienen ciertos factores a nivel de hardware y elementos de software de un computador sobre el desempeño de dos algoritmos con complejidad temporal equivalente ($O(n^2)$). Así mismo se mostrarán los resultados obtenidos y los análisis realizados con herramientas de apoyo como lo son Visual Studio 2017, Minitab y Excel.

1. Introducción

En la actualidad, la sociedad está haciendo mayor uso de herramientas tecnológicas, tales como programas de escritorio o aplicaciones móviles, para resolver problemas con mayor eficiencia. Dentro de estos problemas, se pueden encontrar algunos que requieren, en algún punto de su solución, de un algoritmo de ordenamiento: ordenar los valores presentes en una hoja electrónica, ordenar los registros de personas por edad, ordenar los registros de trabajadores por antigüedad en la empresa, etc. Es por esto que los algoritmos de ordenamiento, aunque poco se hable de ellos, representan un papel de gran importancia para el continuo progreso de la sociedad, pero algo que es aún más importante, es saber elegir el algoritmo de ordenamiento ideal para cada situación.

Existen algoritmos de ordenamiento que funcionan mejor con entradas pequeñas, otros que funcionan mal para entradas pequeñas pero que eventualmente mejoran su tiempo de ejecución para entradas muy grandes, y otros que en definitiva no sirven para ninguna situación, pero que existen.

Aportes: los aportes en este trabajo son:

1. Se identifican cuáles son los factores que influyen en el tiempo de ejecución de los algoritmos estudiados.
2. Se define como los factores influyen en el tiempo de ejecución de los algoritmos.
3. Se presentan recomendaciones y advertencias sobre la utilización de los

algoritmos en casos particulares, dando también una visión global de ellos.

Contorno:

El resto del documento está estructurado de la siguiente manera:

En la sección 2 se describe el objeto de estudio y se proporcionan los objetivos del experimento. En la sección 3 se describe cómo se realiza el experimento y los resultados obtenidos tras dicho procedimiento. En la sección 4 se realiza un análisis empleando métodos estadísticos para examinar los resultados obtenidos. En la sección 5 se interpreta el resultado obtenido en los análisis de la sección anterior. En la sección 6 se presentan las conclusiones encontradas a partir de los resultados obtenidos. En la sección 7 se anexan todos los recursos que sean de utilidad o evidencia del experimento.

2. Planeación

Objetivos:

1. Determinar los factores que afectan el tiempo de respuesta de los algoritmos.
2. Identificar cómo los factores de estudio afectan en la complejidad de los algoritmos.
3. Comprobar la complejidad temporal y espacial de los algoritmos.
4. Dar recomendaciones acerca de la utilización de los algoritmos de Bubble Sort e Insertion Sort.

Unidad Experimental: El objeto de estudio para este experimento son los algoritmos de ordenamiento. Como existen demasiados algoritmos de este tipo se delimita el experimento a solamente dos de ellos, en este caso el algoritmo de ordenamiento por inserción (Insertion Sort) y el algoritmo de ordenamiento burbuja (Bubble Sort). El primero de ellos es un algoritmo de clasificación que ordena una matriz un elemento a la vez, es un algoritmo lento pero tiene la ventaja de que la forma en la que funciona es muy natural y sencillo de entender lo que lo hace comúnmente popular entre las personas que están iniciando en el mundo de la programación; El segundo es un algoritmo sencillo que revisa cada elemento de la lista con el siguiente e intercambiandolos de posición si están equivocados.

Variables de respuesta: la variable que refleja los resultados del experimento es el tiempo de ejecución del algoritmo. Nótese que el tiempo de ejecución del algoritmo no es igual a la complejidad temporal de este, ya que el tiempo de ejecución está dado por múltiplos o submúltiplos

de la unidad temporal segundos (s) y la complejidad temporal está dada por una función matemática. Por estas razones, el resultado va a estar dado en unidades de tiempo, sin embargo, se puede utilizar la complejidad temporal como herramienta para hacer comparaciones y aproximaciones sobre el tiempo de ejecución.

Factores controlables:

A. **Tipo de Algoritmo:** Los algoritmos utilizados presentan grandes diferencias en la estructura que utilizan para realizar el respectivo ordenamiento, es decir, la manera que manejan los arreglos para realizar su función, por lo que puede afectar de manera directa el tiempo de ejecución para cada tratamiento.

B. **Tipo de arreglo:** La manera en que se encuentran los datos de los arreglos también pueden afectar significativamente en el tiempo de ejecución. Los datos en los arreglos están estructurados de tres formas distintas, especificando así los niveles para este factor: orden aleatorio, orden ascendente, orden descendente.

C. **Tamaño del arreglo:** Es importante determinar el impacto que recibe el tiempo de ejecución cuando los algoritmo tienen acceso a estructuras de datos con diferentes tamaños, por lo tanto, se evaluará el tiempo para distintos número de elementos en los arreglos. Los niveles para este factor son: 100, 1000, 100000 y 1000000 elementos.

D. **Procesadores:** Las plataformas físicas y las características técnicas de los procesadores sobre los cuales se ejecutan los algoritmos, también podrían ser factores determinantes que afecten el tiempo de ejecución de los algoritmos, por esto, se decide definir los niveles así:
    - Procesador intel core i7, 7th Gen.
    - Procesador intel core i5.
    - Procesador intel core i3 inside.
    - Procesador AMD Ryzen 3.

Factores no controlables:

Procesos ejecutándose al tiempo: los procesos que se ejecutan al mismo tiempo de ejecución del experimento parecen ser controlables, puesto que se pueden cerrar ventanas de navegadores, editores de texto o entornos de

desarrollo al momento de realizar el experimento. Sin embargo, este factor entra en la categoría de no controlables porque todo computador requiere un mínimo de procesos para su correcto funcionamiento, si los procesos se detienen forzosamente, pueden causar errores fatales que pueden llegar incluso a dañar el computador.

Factores estudiados: Los factores a estudiar son los cuatro definidos previamente en factores controlables, es decir: tipo de algoritmo, tipo de arreglo, tamaño de arreglo y procesadores. La razón de esta decisión es al ser controlables, se pueden hacer variaciones que mejoran la precisión del experimento, pues se pueden encontrar diversas combinaciones de factores, como ver qué pasa con los procesadores y cómo afectan si todos los demás niveles permanecen constantes, por poner un ejemplo. Además, tras hallar cómo influyen estos factores en el tiempo de ejecución, se pueden hacer recomendaciones para mejorarlo.

Procedimiento: Se ejecutan diversos casos de prueba (fijos) y se realizan pruebas que utilizan generadores de arreglos aleatorios. Cada integrante del grupo realiza las pruebas fijas y aleatorias en su computador llevando a cabo un registro en una hoja electrónica donde se evidencian tanto los factores que afectan al experimento, como los resultados obtenidos. El procedimiento será el mismo para cada tratamiento, lo único que varía son los factores definidos y posiblemente también los resultados.

Para llevar a cabo el procedimiento se determinan los siguientes niveles para cada factor:

Niveles para Algoritmo de Ordenamiento	
Nivel 1	Bubble Sort
Nivel 2	Insertion Sort

Tabla 1. Niveles para el Factor Algoritmo

Niveles para Procesador	
Nivel 1	Core i7
Nivel 2	Core i5
Nivel 3	Core i3
Nivel 4	AMD Ryzen 3

Tabla 2. Niveles para el Factor Procesador

Niveles para Tamaño de Arreglo	
Nivel 1	10^2
Nivel 2	10^3
Nivel 3	10^5
Nivel 4	10^6

Tabla 3. Niveles para el Factor Tamaño de Arreglo de Entrada

Niveles para Orden del Arreglo	
Nivel 1	Aleatorio
Nivel 2	Ascendente
Nivel 3	Descendente

Tabla 4. Niveles para el Factor Ordenamiento del Arreglo de Entrada

Tras definir los factores y niveles de cada uno, se obtienen los tratamientos de los cuales consta la experimentación. Estos se hallan combinando cada nivel de cada factor con los niveles de los demás factores, lo cual como resultado arroja 96 tratamientos.
ver **Anexo 1**.

Posteriormente se establece que para cada tratamiento se realizaran 10 repeticiones.

3. Realización

Se desarrolló un programa en el IDE Visual Studio de Microsoft, utilizando el lenguaje de programación C#. En este programa se crearon métodos para manipular los factores controlables con el fin de realizar diferentes tratamientos. Dichos métodos generan arreglos en sus diferentes niveles (aleatorio, ordenado ascendentemente y ordenado descendentemente) según el tamaño deseado, para posteriormente ordenarlos con ambos algoritmos. Se incluye además un método que genera un reporte en una hoja electrónica en formato .xlsx, todo esto automáticamente.
Una vez puesto a correr el programa, solamente hay que esperar a que cada computador de los integrantes del grupo genere su reporte, para su posterior análisis.
Durante el proceso de realización, se pudo notar que los computadores con mejor procesador avanzaban más rápido que los otros, y que los resultados que más tiempo tardaban en registrarse eran aquellos donde el tamaño del arreglo por ordenar era de 1’000,000 (10^6).
Se pueden evidenciar los resultados en el archivo .xlsx presente en la misma carpeta donde se encuentra este documento.

4. Análisis

Usando la herramienta minitab, se realizará un análisis de varianza ANOVA para validar o rechazar las hipótesis planteadas a continuación:

A. Hipótesis para factores individuales.

- Algoritmos:

H_0: Los tiempos para cada algoritmo son iguales

H_1: Al menos uno de los tiempos de los algoritmos es diferente.

- Tamaño del arreglo:

H_0: Los tiempos para cada tamaño son iguales

H_1: Al menos uno de los tiempos del tamaño del arreglo es diferente.

- Tipo de arreglo:

H_0: Los tiempos para cada tipo de arreglo son iguales

H_1: Al menos uno de los tiempos tipo de arreglo es diferente.

- Procesadores:

H_0: Los tiempos para cada procesador son iguales

H_1: Al menos uno de los tiempos para los procesador es diferente.

B. Hipótesis para combinaciones de factores.

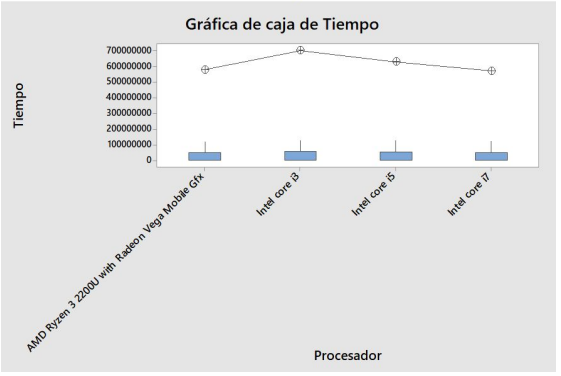
- combinación de factores:

H_0: Los tiempos para cada combinación de factores son iguales

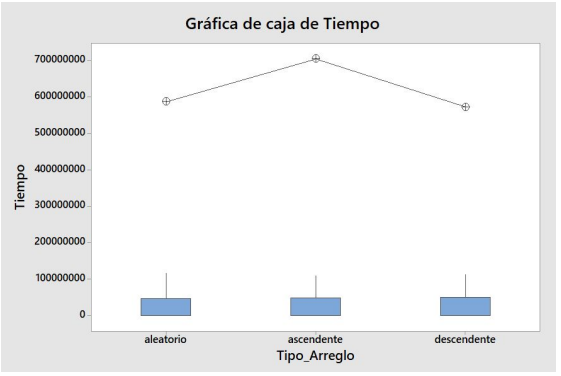
H_1: Al menos uno de los tiempos para cada combinación de factores es diferente

C. Reporte de resultados.

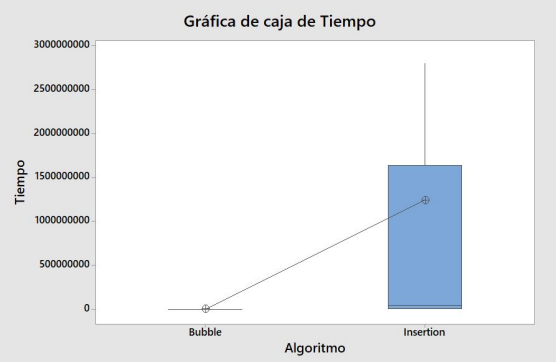
- **Figura 1. Comparaciones por parejas de Fisher respecto al procesador.**



- **Figura 2. Comparaciones por parejas de Fisher respecto al tipo del arreglo.**



- **Figura 3. Comparaciones por parejas de Fisher respecto al algoritmo.**



- **ANOVA por cada factor**

ANOVA: Tiempo vs. Procesador; Tamaño; Algoritmo; Tipo_Arreglo

Información del factor

Factor	Tipo	Niveles	Valores
Procesador	Fijo	4	AMD Ryzen 3 2200U with Radeon Vega Mobile Gfx; Intel core i3; Intel core i5; Intel core i7
Tamaño	Fijo	4	100; 1000; 100000; 1000000
Algoritmo	Fijo	2	Bubble; Insertion
Tipo_Arreglo	Fijo	3	aleatorio; ascendente; descendente

Análisis de varianza de Tiempo

Fuente	GL	SC	MC	F	P
Procesador	3	2,58593E+18	8,61977E+17	0,14	0,939
Tamaño	3	3,94674E+20	1,31558E+20	20,73	0,000
Algoritmo	1	3,69982E+20	3,69982E+20	58,29	0,000
Tipo_Arreglo	2	3,34879E+18	1,67439E+18	0,26	0,768
Error	950	6,02966E+21	6,34701E+18		
Total	959	6,80025E+21			

Resumen del modelo

S	R-cuad.	R-cuad. (ajustado)
2519326821	11,33%	10,49%

5. Interpretación

- Con respecto al procesador, en la figura 1 se observa que para los diferentes procesadores existe un análisis estadístico diferente, aunque realmente similares exceptuando por la configuración Intel Core i5 que se evidencia la demora en ordenar los datos.
- Independiente de los demás factores, se observa que el algoritmo más eficiente es el BubbleSort, por el contrario InsertionSort es el más ineficiente entre estos dos.

6. Conclusiones

Tras analizar los datos obtenidos como resultado de la experimentación, se llega a la conclusión de que todos los factores seleccionados influyen en el tiempo de ejecución del algoritmo de ordenamiento. De la conclusión anterior, desprenden las conclusiones individuales (conclusiones por factor) que se detallan a continuación:

Algoritmo de ordenamiento: como en este caso se seleccionaron dos algoritmos cuya complejidad temporal es cuadrática, se llega a la conclusión de que algoritmos con dicha complejidad son extremadamente ineficientes para cantidades masivas de datos, es decir, arreglos que excedan de 10^5 elementos. Esto se debe a que como la complejidad temporal está dada por una función matemática, su valor puede ir cambiando. En el caso de la función cuadrática, su aumento no es muy notable en números pequeños, pero al mirar una gráfica (curva de crecimiento) de la función, se puede evidenciar que para números arbitrariamente grandes crece muchísimo más.

El mejor algoritmo de entre los dos seleccionados, es el algoritmo de ordenamiento burbuja, o bubble sort como normalmente se le conoce.

Para mejorar el tiempo de ordenamiento de un arreglo, se

recomienda utilizar algoritmos de ordenamiento con complejidad temporal cuadrática para entradas pequeñas (entre 10^1 y 10^3). Para cantidades más grandes, es recomendable usar algoritmos cuya complejidad temporal sea más eficiente que una cuadrática, como los algoritmos de ordenamiento con complejidad temporal logarítmica (merge sort, quick sort y heap sort).

Procesador: en cuanto a procesador, se puede concluir que los procesadores de gama alta ejecutan más rápido los algoritmos que procesadores de gama media. Nótese además, que los procesadores no solo vienen identificados por gama sino también por generación. Esto se debe a que los procesadores de gama alta traen mejoras dentro de sus propios factores controlables, lo que les permite realizar operaciones y ejecutar instrucciones con mayor facilidad.

No hay muchas recomendaciones para disminuir la influencia negativa (o aumentar la positiva) de un procesador sobre el tiempo de ejecución del algoritmo, ya que el procesador puede ser un factor de control costoso. Esto debido a que nadie cambia un procesador solamente para ordenar un arreglo, por ello se aconseja elegir bien al momento de comprar un computador, el procesador importa mucho. Además, por muy bueno que sea el procesador si la complejidad temporal tiende a crecer demasiado, el computador puede llegar a tardar horas en ejecutar el algoritmo al encontrarse con arreglos muy grandes, como sucedió en esta experimentación.

7. Anexos

Diagrama de clases

Diagrama de objetos

Datos obtenidos

References

Sedgewick, R. (1989). *Algorithms*. Addison-Wesley.

Anexo_1. Tratamientos del Experimento.

Tratamiento	Algoritmo	Procesador	Tamaño	Tipo Arreglo
1	Bubble	Intel core i7	100	aleatorio
2	Bubble	Intel core i7	100	ascendente
3	Bubble	Intel core i7	100	descendente
4	Bubble	Intel core i7	1000	aleatorio
5	Bubble	Intel core i7	1000	ascendente
6	Bubble	Intel core i7	1000	descendente
7	Bubble	Intel core i7	100000	aleatorio
8	Bubble	Intel core i7	100000	ascendente
9	Bubble	Intel core i7	100000	descendente
10	Bubble	Intel core i7	1000000	aleatorio
11	Bubble	Intel core i7	1000000	ascendente
12	Bubble	Intel core i7	1000000	descendente
13	Insertion	Intel core i7	100	aleatorio
14	Insertion	Intel core i7	100	ascendente
15	Insertion	Intel core i7	100	descendente
16	Insertion	Intel core i7	1000	aleatorio
17	Insertion	Intel core i7	1000	ascendente
18	Insertion	Intel core i7	1000	descendente
19	Insertion	Intel core i7	100000	aleatorio
20	Insertion	Intel core i7	100000	ascendente
21	Insertion	Intel core i7	100000	descendente
22	Insertion	Intel core i7	1000000	aleatorio
23	Insertion	Intel core i7	1000000	ascendente
24	Insertion	Intel core i7	1000000	descendente

25	Bubble	Intel core i5	100	aleatorio
26	Bubble	Intel core i5	100	ascendente
27	Bubble	Intel core i5	100	descendente
28	Bubble	Intel core i5	1000	aleatorio
29	Bubble	Intel core i5	1000	ascendente
30	Bubble	Intel core i5	1000	descendente
31	Bubble	Intel core i5	100000	aleatorio
32	Bubble	Intel core i5	100000	ascendente
33	Bubble	Intel core i5	100000	descendente
34	Bubble	Intel core i5	1000000	aleatorio
35	Bubble	Intel core i5	1000000	ascendente
36	Bubble	Intel core i5	1000000	descendente
37	Insertion	Intel core i5	100	aleatorio
38	Insertion	Intel core i5	100	ascendente
39	Insertion	Intel core i5	100	descendente
40	Insertion	Intel core i5	1000	aleatorio
41	Insertion	Intel core i5	1000	ascendente
42	Insertion	Intel core i5	1000	descendente
43	Insertion	Intel core i5	100000	aleatorio
44	Insertion	Intel core i5	100000	ascendente
45	Insertion	Intel core i5	100000	descendente
46	Insertion	Intel core i5	1000000	aleatorio
47	Insertion	Intel core i5	1000000	ascendente
48	Insertion	Intel core i5	1000000	descendente
49	Bubble	Intel core i3	100	aleatorio

50	Bubble	Intel core i3	100	ascendente
51	Bubble	Intel core i3	100	descendente
52	Bubble	Intel core i3	1000	aleatorio
53	Bubble	Intel core i3	1000	ascendente
54	Bubble	Intel core i3	1000	descendente
55	Bubble	Intel core i3	100000	aleatorio
56	Bubble	Intel core i3	100000	ascendente
57	Bubble	Intel core i3	100000	descendente
58	Bubble	Intel core i3	1000000	aleatorio
59	Bubble	Intel core i3	1000000	ascendente
60	Bubble	Intel core i3	1000000	descendente
61	Insertion	Intel core i3	100	aleatorio
62	Insertion	Intel core i3	100	ascendente
63	Insertion	Intel core i3	100	descendente
64	Insertion	Intel core i3	1000	aleatorio
65	Insertion	Intel core i3	1000	ascendente
66	Insertion	Intel core i3	1000	descendente
67	Insertion	Intel core i3	100000	aleatorio
68	Insertion	Intel core i3	100000	ascendente
69	Insertion	Intel core i3	100000	descendente
70	Insertion	Intel core i3	1000000	aleatorio
71	Insertion	Intel core i3	1000000	ascendente
72	Insertion	Intel core i3	1000000	descendente
73	Bubble	AMD Ryzen 3	100	aleatorio
74	Bubble	AMD Ryzen 3	100	ascendente

75	Bubble	AMD Ryzen 3	100	descendente
76	Bubble	AMD Ryzen 3	1000	aleatorio
77	Bubble	AMD Ryzen 3	1000	ascendente
78	Bubble	AMD Ryzen 3	1000	descendente
79	Bubble	AMD Ryzen 3	100000	aleatorio
80	Bubble	AMD Ryzen 3	100000	ascendente
81	Bubble	AMD Ryzen 3	100000	descendente
82	Bubble	AMD Ryzen 3	1000000	aleatorio
83	Bubble	AMD Ryzen 3	1000000	ascendente
84	Bubble	AMD Ryzen 3	1000000	descendente
85	Insertion	AMD Ryzen 3	100	aleatorio
86	Insertion	AMD Ryzen 3	100	ascendente
87	Insertion	AMD Ryzen 3	100	descendente
88	Insertion	AMD Ryzen 3	1000	aleatorio
89	Insertion	AMD Ryzen 3	1000	ascendente
90	Insertion	AMD Ryzen 3	1000	descendente
91	Insertion	AMD Ryzen 3	100000	aleatorio
92	Insertion	AMD Ryzen 3	100000	ascendente
93	Insertion	AMD Ryzen 3	100000	descendente
94	Insertion	AMD Ryzen 3	1000000	aleatorio
95	Insertion	AMD Ryzen 3	1000000	ascendente
96	Insertion	AMD Ryzen 3	1000000	descendente