

UNIVERSIDAD AUTÓNOMA DE AGUASCALIENTES

CENTRO DE CIENCIAS BÁSICAS

DEPARTAMENTO DE SISTEMAS ELECTRÓNICOS

PROGRAMACIÓN I

**“COMPARATIVA ENTRE LOS DIFERENTES ALGORITMOS DE ORDENACIÓN”**

**EQUIPO**

**Alumno:** BELTRÁN ARROYO LUIS **ID:** 272674

**Alumno:** DÁVILA CAMPOS EDUARDO **ID:** 269314

**Alumno:** DELGADO LUNA VÍCTOR EMILIO **ID:** 227894

**Alumno:** GAYTÁN SÁNCHEZ BRAYAN DE JESÚS **ID:** 231028

**Alumno:** SÁNCHEZ ARIAS JOSÉ EMILIANO **ID:** 228488

**Carrera**: INGENIERÍA EN SISTEMAS COMPUTACIONALES

**Semestre**: 3°A

**Profesora**: Eduardo Serna Pérez

**Fecha**: 18 de diciembre de 2020

INVESTIGACIÓN

**“Comparativa entre los diferentes algoritmos de ordenación”**

**Comparativa entre los diferentes algoritmos de ordenación**

*Consideraciones:* Para la comprensión de la tabla que se muestra a continuación se hace uso de la notación Big O, Big Ω y Big ;

Es necesario, primeramente, definir estos conceptos:

*La complejidad del tiempo:* Es una función que describe la cantidad de tiempo que toma un algoritmo en términos de la cantidad de entrada al algoritmo.

*La complejidad espacial*: Es una función que describe la cantidad de memoria (espacio) que toma un algoritmo en términos de la cantidad de entrada al algoritmo.

Las operaciones de Big O reflejan el peor caso del algoritmo en cuestión, mientras que Big Ω y Big reflejan el mejor caso y el caso promedio respectivamente.

La variable N es utilizada para simbolizar la cantidad de objetos a ordenar, por tanto, el tiempo de ejecución obviamente varia conforme varíe la cantidad.

Para obtener un tiempo aproximado solo es necesario realizar las operaciones que se encuentran entre paréntesis sustituyendo la N por el valor adecuado. El caso de O (1) muestra que no importa la variación de N, el algoritmo siempre se ejecutará en una sola operación.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algoritmo de ordenación | Complejidad de tiempo | | | Complejidad espacial |
| Mejor caso | Caso promedio | Peor caso | Peor caso |
| Inserción | Ω(N) | (N2) | O(N2) | O (1) |
| Selección | Ω(N2) | (N2) | O(N2) | O (1) |
| Burbuja | Ω(N) | (N2) | O(N2) | O (1) |
| Mergesort | Ω (N log N) | (N log N) | O (N log N) | O(N) |
| Shellsort | Ω (N log N) | (N log N) | O(N) | O (1) |
| Quicksort | Ω (N log N) | (N log N) | O(N2) | O (N log N) |
| Heapsort | Ω (N log N) | (N log N) | O (N log N) | O (1) |

**¿Como se mide el tiempo de ejecución de un algoritmo?**

Tradicionalmente, la medición del tiempo de ejecución de un algoritmo como se toma como el tiempo de CPU. Para ello se almacena el tiempo actual del sistema de un ordenador en una variable v, se ejecuta el algoritmo A a evaluar y después de que el algoritmo termine, se almacena el tiempo actual del sistema en otra variable w. C(A)=w-v es entonces el tiempo consumido por una ejecución de A.

La mayoría de los lenguajes de programación ofrecen funciones para acceder a la hora actual del sistema que pueden utilizarse para realizar tales mediciones.

**¿De qué factores depende el tiempo de ejecución de un algoritmo? Describe cada factor.**

Dos áreas son importantes para el desempeño:

* *La eficiencia espacial***:** La memoria requerida, también llamada, complejidad espacial
* *Eficiencia en el tiempo:* El tiempo requerido, también llamado complejidad del tiempo

Hay algunas circunstancias en las que el espacio/memoria utilizado debe ser analizado. Por ejemplo, para grandes cantidades de datos o para la programación de sistemas empotrados.

Componentes del uso del espacio/memoria:

1. *Espacio de instrucción.* Afectado por: el compilador, las opciones del compilador, la computadora de destino (CPU).
2. *Espacio de datos.* Afectado por: el tamaño de los datos/memoria asignada dinámicamente, variables estáticas del programa.
3. *Espacio de pila en tiempo de ejecución.* Afectado por: el compilador, llamadas a funciones en tiempo de ejecución y recursividad, variables locales, parámetros.

**¿Qué es una medida asintótica?**

El análisis asintótico de un algoritmo se refiere a la definición de la base/encuadre matemático de su rendimiento en tiempo de ejecución. Utilizando el análisis asintótico, podemos muy bien concluir el mejor caso, el caso medio y el peor caso de un algoritmo.

El análisis asintótico está limitado por la entrada, es decir, si no hay entrada en el algoritmo, se concluye que funciona en un tiempo constante. Aparte de la "entrada", todos los demás factores se consideran constantes.

**¿Qué es la cota superior de un algoritmo y con qué símbolo se representa?**

La notación Big O describe la cota superior (un límite superior). En otras palabras, la notación Big O establece una afirmación acerca de la mayor cantidad de algún recurso (normalmente tiempo) que requiere un algoritmo para alguna clase de entradas de tamaño n (típicamente la peor de esas entradas, el promedio de todas las entradas posibles, o la mejor de esas entradas).

**¿Qué es la cota inferior de un algoritmo y con qué símbolo se representa?**

La función Big Omega(Ω) se utiliza en la informática para describir el rendimiento o la complejidad de un algoritmo.

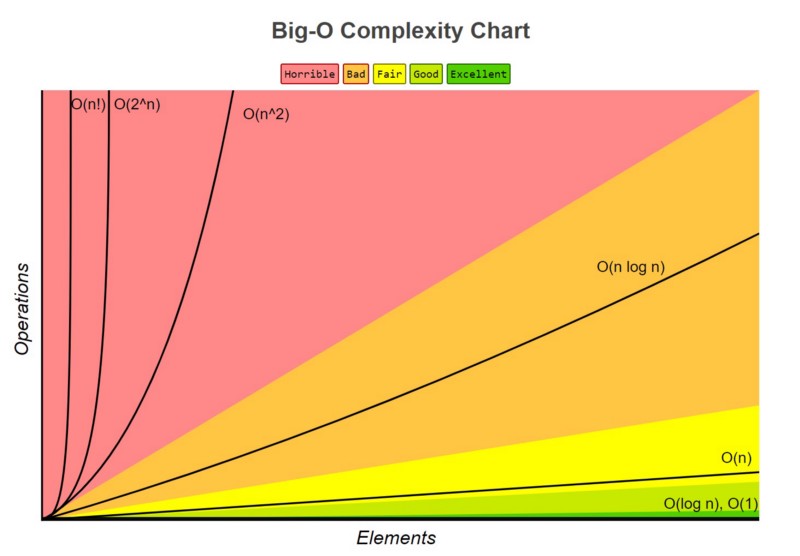
Se dice que el tiempo de ejecución es "Big-Ω de f(n)". se usa la notación Big-Ω para la cota inferior (los límites inferiores asintóticos), ya que limita el crecimiento del tiempo de funcionamiento desde abajo para tamaños de entrada lo suficientemente grandes.

**¿Qué es la notación O grande?**

La notación Big O es una de las herramientas fundamentales para que los informáticos analicen el coste de un algoritmo.

La notación Big O es una notación matemática que describe el comportamiento limitante de una función cuando el argumento tiende hacia un valor particular o infinito. Es un miembro de una familia de notaciones inventadas por Paul Bachmann, Edmund Landau y otros, llamadas colectivamente notación Bachmann-Landau o notación asintótica.

En palabras simples, la notación Big O describe la complejidad de un código usando términos algebraicos.



**A qué se refiere analizar un algoritmo en el mejor, peor y caso promedio.**

Por lo general, el tiempo requerido por un algoritmo se divide en tres tipos -

* *En el mejor de los casos*: el tiempo mínimo requerido para la ejecución del programa.
* *Caso promedio:* Tiempo promedio requerido para la ejecución del programa.
* *Peor caso:* Tiempo máximo requerido para la ejecución del programa.

**Referencias bibliográficas**

Anónimo. (30 de abril de 2019). Análisis de las diferentes técnicas de ordenamiento. <https://www.geeksforgeeks.org/analysis-of-different-sorting-techniques/#:~:text=A%20sorting%20technique%20is%20stable,merge%20sort%20are%20stable%20techniques.&text=Similarly%2C%20quick%20sort%20and%20heap%20sort%20are%20also%20unstable>.

Anónimo. (07 de septiembre de 2020). Diferencia entre Big O, Big Omega y Big Theta. <https://www.geeksforgeeks.org/difference-between-big-oh-big-omega-and-big-theta/>

Anónimo. (2020). Estructuras de datos. Análisis Asintótico. <https://www.tutorialspoint.com/data_structures_algorithms/asymptotic_analysis.htm#:~:text=It%20measures%20the%20worst%20case,f(n)%20%E2%89%A4%20c>.

Institute of Applied Optimization. (2020). Medición del tiempo de ejecución de los algoritmos (de optimización). <http://iao.hfuu.edu.cn/blogs/science-blog/29-measuring-the-runtime-of-optimization-algorithms>

Anónimo. (2018). Eficiencia de los Algoritmos. <http://www.cs.kent.edu/~durand/CS2/Notes/03_Algs/ds_alg_efficiency.html>

Huang, S. (16 de enero de 2020). ¿Qué es la explicación de la notación de "Big O"? Complejidad de espacio y tiempo. <https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/>

Anónimo. (06 de julio de 2020). Cotas inferiores y la notación . <https://opendsa-server.cs.vt.edu/ODSA/Books/CS3/html/AnalLower.html>