Luis Enrique Gutierrez Mendioroz

A00837481

Programacion de Estructuras de Datos y Algoritmos Fundamentales (Gpo 850)

Profesor: Eduardo Rodriguez Tello

Reflexión Algoritmos de Ordenamiento y Búsqueda

Reflexión Actividad Integradora 1

En el ámbito de crear algoritmos computacionales es importante tomar en cuenta y de manera muy rigurosa la eficiencia y complejidad computacional de un algoritmo. Dentro del campo de la tecnología y desarrollo de algoritmos existen una serie de códigos que nos permiten realizar diferentes acciones como lo es buscar y ordenar información. Para ejemplificar dicho caso, en este proyecto se utilizó a modo de comparación los algoritmos QuickSort BubbleSort y Sort (en el caso de búsqueda). En este escrito reflexivo se hará una comparación de dichos algoritmos en el cual veremos cómo se ve afectada la eficiencia de cada uno al utilizar diferentes metodologías.

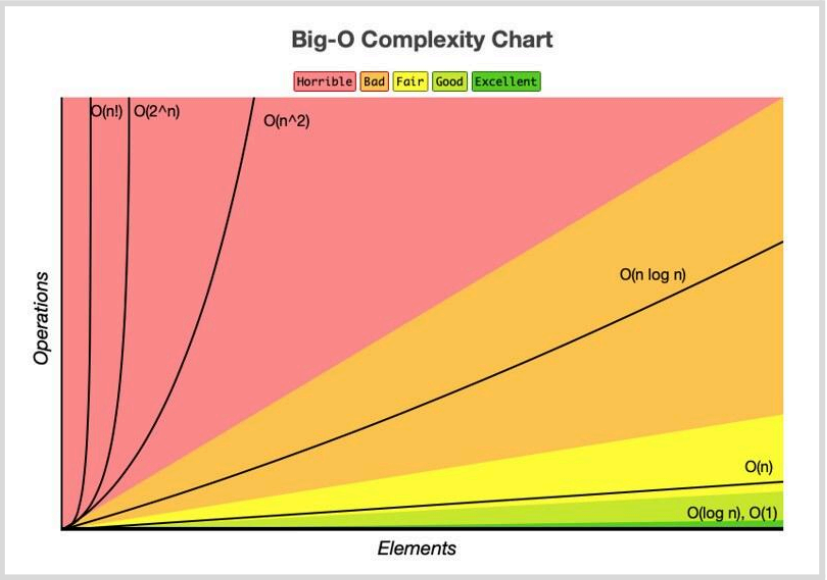
Se hará una breve explicación para poder contextualizar el funcionamiento de cada algoritmo.

El Bubble Sort realiza comparaciones lineales de una lista haciendo una serie de swaps de izquierda a derecha. Significado que hace un cambio de índice entre dos valores colindantes. Cómo consiguiente y dada la complejidad del proyecto presentado este algoritmo tiene complejidad de o(n^2), dado que el tipo de información a ordenar es extensa y no se encuentra ordenada. Es importante recalcar que no es el algoritmo más óptimo para ordenar series de información muy grandes gracias a su tipo de complejidad computacional por lo que podemos concluir que este algoritmo va a tomar más tiempo en realizar el ordenamiento.

Por otro lado, en cuanto al algoritmo de ordenamiento Quick Sort realiza una partición de la lista y declara un pivote (cualquier valor). Efectúa una serie de comparaciones y swaps ubicando los valores menores a la izquierda y los valores mayores a la derecha. A diferencia del BubbleSort, QuickSort tiene una complejidad computacional promedio de o(log n) por ende es de los algoritmos más rápidos y más conveniente de utilizar para realizar ordenamiento de información.

El entender el concepto de complejidad computacional, nos puede dar una idea coherente de cuál algoritmo tardará más. Es decir que, el algoritmo que tenga la complejidad o(n) será el más rápido ya que crece de una forma más lenta a medida que se le aumenta la carga de la información. La siguiente gráfica muestra la relación que existe entre la cantidad

de elementos y operaciones en un algoritmo y lo categorizan en base a los resultados.



Gráfica 1.

Obtenida de europeanvalley.es - la complejidad de los algoritmos

Analizando la complejidad computacional de ambos algoritmos podemos y basándonos en la información anteriormente presentada podemos inferir a simple vista que el algoritmo de QuickSort será el más efectivo.

En el caso particular del proyecto se realizaron pruebas con ambos algoritmos y se guardaron en archivos de texto, donde se muestra la información para llegar a una conclusión.

comparacion_Quick.txt

```
1  Ingrese el metodo de ordenamiento a utilizar: 1 =
   función Sort de C++, 2 = bubbleSort, 3 = quicksort. 1
2  Ingrese el caso de prueba a utilizar: 1, 2, 3. 3
3
4  Oct 20 23:45:19 31.197.29.145:2896  Failed password
   for illegal user guest
5  Oct 21 00:20:41 211.179.220.223:7881  Failed
   password for illegal user guest
6  Oct 21 00:29:14 208.57.27.44:1137  Failed password
   for illegal user root
7  Numero de salidas de la busqueda: 3
8  Tiempo de ejecución en ms: 77
9  Comparaciones realizadas: 293346
10
11 Jun 1 00:22:36 49.121.182.153:6021  Failed password
   for illegal user guest
12 Jun 1 00:34:43 254.243.231.221:7416  Failed password
   for illegal user guest
13 Jun 1 00:49:31 15.113.211.66:1795  Failed password
   for illegal user root
```

Imagen 1. Comparación de QuickSort

comparacion_Bubble.txt

```
   for illegal user guest
474 Oct 30 23:22:47 90.96.74.140:4913  Failed password
   for illegal user guest
475 Oct 30 23:36:16 177.229.139.75:5410  Failed password
   for illegal user root
476 Oct 30 23:39:19 17.210.90.53:8022  Illegal user
477 Oct 30 23:48:41 182.153.115.244:4855  Illegal user
478 Numero de salidas de la busqueda: 474
479 Tiempo de ejecución en ms: 21001
480 Comparaciones realizadas: 16806
481 Intercambios realizados: 70166913
482
483 Jun 1 00:22:36 49.121.182.153:6021  Failed password
   for illegal user guest
484 Jun 1 00:34:43 254.243.231.221:7416  Failed password
   for illegal user guest
485 Jun 1 00:49:31 15.113.211.66:1795  Failed password
   for illegal user root
486 Jun 1 00:59:02 159.72.70.232:99  Failed password for
   illegal user guest
```

Imagen 2. Comparación de BubbleSort

Dado los resultados obtenidos anteriormente podemos observar que el algoritmo de QuickSort es sumamente más efectivo en cuanto a comparaciones, número de salidas y tiempo de ejecución. Gracias a los datos obtenidos en las pruebas podemos confirmar que un algoritmo que tenga complejidad computacional de o(log n) siempre será el más efectivo. Sin embargo, la idea que QuickSort sea más rápido no significa que BubbleSort no sea efectivo sino que si lo es pero con mayor demora.

En definitiva, el objetivo principal va muy de la mano con la complejidad computacional de los algoritmos. Al crear algoritmos que sean simples, mantenibles y sobre todo eficaces para poder llevar a cabo tareas complejas nos dará el resultado esperado en el menor tiempo posible. Finalmente, como se ha podido comprobar nuestro programa de QuickSort ha demostrado ser el algoritmo más rápido y efectivo en cuanto a ordenar información se refiere.

Referencias

GfG. (2023, November 21). *Bubble sort - data structure and algorithm tutorials*. GeeksforGeeks. https://www.geeksforgeeks.org/bubble-sort/

GfG. (2024, March 22). *Quicksort - data structure and algorithm tutorials*. GeeksforGeeks. https://www.geeksforgeeks.org/quick-sort/

Jorge Calvo Profesor y Responsable de Tecnología e Innovación en Colegio Europeo de Madrid -- Asesor y Formador Tecnología Educativa -- CoFundador EuropeanValley Institución, & Facebook Twitter, Instagram . (n.d.). *La Complejidad de los Algoritmos*. Blog Europeanvalley. https://www.europeanvalley.es/noticias/la-complejidad-de-los-algoritmos/#:~:text=La%20complejidad%20de%20un%20algoritmo,el%20contexto%20de%20los%20algoritmos.