Analisis de Algoritmos

==

\* Profesor: Mauricio Rojas

\* Alumno: Jeffrey Camareno

\* Tarea 1

Hipotesis para el algoritmo de Bubble Sort

==

Este algoritmo de ordenamiento basicamente lo que hace es ir revisando cada elemento de la lista, vector o arreglo que va a ser ordenada con el siguiente, intercambiándolos de posición si están en el orden equivocado. Es necesario revisar varias veces toda la lista hasta que no se necesiten más intercambios.

Por esta razon me parece que el algoritmo pierde eficiencia dado que al tener que revisar y analizar cada elemento de la lista mas de una vez, hace que el consumo de recurso a nivel de tiempo sea bastante alto

Hipotesis para el algoritmo selection Sort

==

El ordenamiento de burbuja y su funcionamiento basicamente es ir comparando en parejas los elementos de un vector, arreglo, o lista.

La idea principal de este algoritmo es comparar si el segundo elemento de la pareja es mayor o menor al primer elemento de la pareja, y asi sucesivamente va tomando el primer elemento con forme se vaya encontrando el elemento menor de la lista.

En otras palabras su funcionamiento es:

\* Buscar el mínimo elemento de la lista

\* Intercambiarlo con el primero

\* Buscar el siguiente mínimo en el resto de la lista

\* Intercambiarlo con el segundo

Creo que el resultado de este algoritmo es mejor, o sea mas eficiente a la hora de realizar el ordenamiento tomando en cuenta el recurso del tiempo de ejecucion.

Determinacion de parametros y resultados obtenidos para bubble sort Tomando una lista de 100 mil elementos:

==

\* Mejor Caso: Que la lista esté ordenada, su tiempo de ejecucion fue de 112,60 segundos

\* Peor Caso: Que la lista esté ordenada en sentido inverso, su tiempo de ejecucion fue de 562,21 segundos

\* Caso Promedio (medio): Que la lista esté desordenado aleatoriamente, su tiempo de ejecucion fue de 344,51 segundos

Determinacion de parametros y resultados obtenidos para selecction sort Tomando una lista de 100 mil elementos:

==

\* Mejor Caso: Que la lista esté ordenada, su tiempo de ejecucion fue de 8,53 segundos

\* Peor Caso: Que la lista esté ordenada en sentido inverso, su tiempo de ejecucion fue de 97,46 segundos

\* Caso Promedio (medio): Que la lista esté desordenado aleatoriamente, su tiempo de ejecucion fue de 19,20 segundos

Conclusiones:

==

Finalmente logro concluir por definitiva que es mucho mas eficiente el algoritmo de seleccion sort comparado al bubble sort, esto tomando en cuenta el recuerdo del tiempo de ejecucion.

Las pruebas fueron realizadas tomando en cuenta las siguientes especificaciones:

\* Sistema Operativo: Windows 8.1

\* Memoria Ram: 4 Gb

\* Procesador: Intel Core i5 (2.53 GHz)