Instituto Politécnico Nacional

Unidad Profesional Interdisciplinaria de Ingeniería

Campus Zacatecas.

Práctica 1:

"Análisis de casos".

Materia y Grupo:

Análisis y Diseño de Algoritmos. 3CM2.

Profesora a cargo:

Erika Paloma Sánchez-Femat.

Alumno:

Jorge Ulises Zapata Arteaga.

1 Introducción.

En esta práctica, tenemos como objetivo analizar a fondo un par de algoritmos de ordenamiento en listas. En este análisis tocaremos temas como lo son las complejidades, así como la comparación de casos (mejor, peor y promedio). Asímismo analizaremos el tiempo de ejecución de cada algoritmo tomando en cuenta cada caso estudiado.

2 Desarrollo de la práctica

a) Análisis de casos

- Calcular el mejor caso, peor caso y caso promedio para ambos algoritmos de ordenamiento.
  Proporcionar 3 ejemplos de datos de entrada para cada caso.

Algoritmo burbuja simple.
1.- Mejor caso: El mejor caso para el ordenamiento burbuja simple es cuando, el usuario, ingresa la lista ya ordenada ascendente. Esto ocasiona que el algoritmo no haga intercambios, simplemente recorre el arreglo comparando que estén ordenados.
Entrada: [4, 5, 6, 7, 8, 9, 10].

2.- Peor caso: El peor caso para el ordenamiento burbuja simple es cuando se ingresa la lista en orden descendente, es decir, orden invertido, en este caso el algoritmo tendrá

que hacer la mayor cantidad de comparaciones e intercambios.
Entrada: [10, 9, 8, 7, 6, 5, 4].

3.- Caso promedio: El caso promedio para el ordenamiento burbuja simple es cuando, el usuario ingresa la lista en orden aleatorio, en este caso la cantidad de intercambios y comparaciones dependerá exclusivamente de la posición de los datos de que el usuario ingrese en la entrada.
Entrada: [8, 4, 7, 10, 6, 5, 9].

Algoritmo burbuja mejorado.

1.- Mejor caso: Al igual que en el ordenamiento burbuja simple, el mejor caso para el ordenamiento mejorado es cuando la lista es ingresada ordenada ascendentemente, de este modo solo hará comparaciones, no intercambios.
Entrada: [1, 2, 3, 4, 5, 6, 7].

2.- Peor caso: El peor caso para el ordenamiento mejorado es cuando se ingresa la lista en orden descendente, en este caso también hará el máximo número de comparaciones e intercambios, pero puede omitir algunas comparaciones si no se realizan intercambios en una pasada.
Entrada: [7, 6, 5, 4, 3, 2, 1].

3.- Caso promedio: El caso promedio para el ordenamiento burbuja mejorado es cuando, el usuario ingresa la lista en orden aleatorio, la cantidad de intercambios y comparaciones también dependerá exclusivamente de la posición

de los datos de que el usuario ingrese en la entrada, pero se espera una mejora de rendimiento debido a las optimizaciones.
Entrada: [3, 6, 2, 4, 1, 5, 7].

- Encontrar y justificar a cuál clase de las siguientes pertenece cada caso de cada algoritmo:
  O(1), O(log n), O(n), O(n log n), O(n2), O(2n)

Algoritmo burbuja simple.
Mejor caso (Complejidad O(n)): cuando la lista ya está ordenada el algoritmo no debe hacer intercambios, solo debe recorrer el arreglo una vez para ver si no hacen falta más intercambios, lo que depende de la longitud n de la lista, por eso la complejidad es O(n).

Peor caso (Complejidad O(n2)): En el peor caso el algoritmo tiene que hacer el número máximo de intercambios, esto implica un número cuadrático de comparaciones.

Caso promedio (Complejidad O(n2)): El caso promedio se encuentra en algún punto intermedio entre el mejor y el peor caso. En promedio, el algoritmo realizará aproximadamente (n2)/2 comparaciones. Por lo tanto, el caso promedio es O(n2).

Algoritmo burbuja mejorado.

Mejor caso (Complejidad O(n)): cuando la lista ya está ordenada, el algoritmo no debe hacer intercambios, al igual que en burbuja simple solo recorre los n elementos.

Peor caso (Complejidad O(n2)): En el peor caso el algoritmo tiene que hacer el número máximo de intercambios, pero puede omitir algunas comparaciones. Aunque es más eficiente que el burbuja simple en este caso, todavía tiene una complejidad cuadrática.

Caso promedio (Complejidad O(n2)): El caso promedio también se encuentra en algún punto intermedio entre el mejor y el peor caso, realizando aproximadamente (n2)/2 comparaciones. Por lo tanto, el caso promedio es O(n2).

b) Comparación de resultados

- Comparar el tiempo de ejecución de cada algoritmo de ordenamiento para el mejor, peor y caso promedio. ¿Qué conclusiones puedes sacar de estos resultados?

Algoritmo burbuja simple.
Mejor caso con entrada [4, 5, 6, 7, 8, 9, 10]: 5.56 seg.
Peor caso con entrada [10, 9, 8, 7, 6, 5, 4]: 8.05 seg.
Caso promedio con entrada [8, 4, 7, 10, 6, 5, 9]: 16 seg.

Algoritmo burbuja mejorado.
Mejor caso con entrada [1, 2, 3, 4, 5, 6, 7]: 4.49 seg.
Peor caso con entrada [7, 6, 5, 4, 3, 2, 1]: 7.33 seg.
Caso promedio con entrada [3, 6, 2, 4, 1, 5, 7]: 18.40 seg.

De lo anterior podemos concluir que hay una clara diferencia en términos de eficiencia entre ambos algoritmos, en el ordenamiento burbuja mejorado podemos observar una diferencia de aproximadamente 3 segundos con respecto al ordenamiento simple en el mejor y peor caso; sin embargo, en el caso promedio observamos un aumento de 2 segundos en el tiempo de ejecución, pero lo podemos atribuir al ordenamiento de las entradas. En términos generales, podemos decir que, al menos en esta práctica, el método mejorado de ordenamiento cumple su función en su mayoría.

c) Análisis paso a paso de cada algoritmo.
En ese apartado se expondrán los pasos lógicos que se siguieron para la creación de cada uno de los algoritmos, así como también se definirá que complejidad tienen los algoritmos.

Algoritmo de ordenamiento burbuja simple.

Paso 1: Definimos una variable en la que el usuario defina el tamaño de la lista.

Paso 2: Declaramos un arreglo 'lista' para en él almacenar todos los números que ingrese el usuario (lista=[]).

Paso 3: Creamos un bucle para pedir al usuario ingresar los números que él desee poner dentro de la lista y los guardamos (lista.append(n)) .

Paso 4: Imprimimos la lista con un for antes de ordenarla para tener mejor visión del proceso.

Paso 5: Suponemos el arreglo [5, 3, 1]Empezamos a hacer comparaciones: ¿el número en la posición i=0(lista[0]=5) es mayor al de la posición i+1 (lista[1]=3)?. Sí, entonces movemos el número 5 a la siguiente posición y el 3 pasa a ocupar la posición anterior, es decir [3, 5, 1], para esto utilizamos una variable aux que nos sirve como un mediador para almacenar los números en lo que hacemos el intercambio.

Paso 6: ¿Es lista[1] >lista[2]? Sí, 5>1, entonces 5 pasa a la posición del 1, quedando: [3, 1, 5].

Paso 7: Ahora hacemos otras comparaciones para verificar si se necesita intercambiar otro número:

¿Es lista[0] >lista[1]? Sí, 3>1, entonces 3 pasa a la posición de 1 y el número 1 se regresa una posición, quedando [1, 3, 5]

Paso 8: ¿Es lista[1] >lista[2]? No, entonces ahí se queda. Quedando ordenado en un total de 4 comparaciones.

Algoritmo de ordenamiento burbuja mejorado.
Paso 1: Definimos una variable en la que el usuario defina el tamaño de la lista.
Paso 2: Declaramos un arreglo 'lista' para en él almacenar todos los números que ingrese el usuario (lista=[]).
Paso 3: Creamos un bucle para pedir al usuario ingresar los números que él desee poner dentro de la lista y los guardamos (lista.append(n)) .
Paso 4: Imprimimos la lista con un for antes de ordenarla para tener mejor visión del proceso.
Paso 5: Declaramos una variable que sirve como bandera para verificar si hubo cambios realizados, para así no hacer comparaciones innecesarias.
Paso 6: Hacemos el mismo procedimiento de comparaciones al igual que en el ordenamiento burbuja simple.

3 Conclusiones.

En conclusión, hemos analizado a profundidad los algoritmos de ordenamiento burbuja simple y optimizado. Hemos visto la complejidad de ambos algortimos, así como sus complejidades individuales en los mejores, peores y casos promedios. Además tenemos una comparación en tiempos de ejecución donde podemos observar la eficiencia de cada caso. Concluyendo así que, aunque es posible optimizar el ordenamiento burbuja, en realidad no es eficiente en la práctica y uso de listas con gran cantidad de datos.

4 Referencias bibliográficas.

https://juncotic.com/ordenamiento-de-burbuja-algoritmos-de-ordenamiento/
https://www.youtube.com/watch?v=7fgE0ZFtwPk