PRACTICA 01

Alumno: Emiliano Martinez Torres

September 16, 2023

1 Introducción

En el presente reporte se llevará a cabo la demostración de los resultados obtenidos al realizar un código en el lenguaje de programación de Python de dos tipos de algoritmos diferentes el de ordenamiento por burbuja y ordenamiento por burbuja optimizada, comparando entre sí el tiempo de ejecución del algoritmo en distintos casos.

2 Desarrollo

2.1 Algoritmo de Burbuja:

El algoritmo de Burbuja es un algoritmo de ordenamiento simple que compara repetidamente elementos adyacentes en una lista y los intercambia si están en el orden incorrecto. El proceso se repite hasta que no se realizan más intercambios, lo que significa que la lista está ordenada.

A continuación, se explica el procedimiento paso a paso:

1. Inicio: Se inicia con una lista de elementos desordenados.
2. Comparación: El algoritmo compara el primer elemento con el siguiente (elemento actual y siguiente elemento).
3. Intercambio: Si el elemento actual es mayor que el siguiente, se intercambian.
4. Repetición: Se repiten los pasos 2 y 3 para todos los elementos de la lista.
5. Finalización: El algoritmo se repite desde el principio hasta que no se realicen más intercambios en un pase completo por la lista. Esto significa que la lista está ordenada.

El algoritmo de Burbuja puede ser ineficiente para listas grandes, ya que requiere un número significativo de comparaciones e intercambios.

2.2 Algoritmo de Burbuja Optimizado:

El algoritmo de Burbuja Optimizado es una mejora del algoritmo de Burbuja básico. La optimización se basa en el hecho de que después de cada paso a través de la lista, el elemento más grande ya está en su posición correcta. Por lo tanto, no es necesario volver a compararlo en las pasadas posteriores. Esta optimización reduce el número de comparaciones en comparación con el algoritmo de Burbuja básico.

A continuación, se explica el procedimiento paso a paso:

1. Inicio: Se inicia con una lista de elementos desordenados.
2. Comparación e Intercambio: Se comparan y se intercambian elementos adyacentes, al igual que en el algoritmo de Burbuja básico.
3. Marcar Cambio: Se mantiene un indicador (por ejemplo, una bandera llamada "swapped") para verificar si se realizó algún intercambio durante una pasada completa por la lista.
4. Repetición: Se repiten los pasos 2 y 3 para todos los elementos de la lista.
5. Finalización: El algoritmo se repite desde el principio hasta que no se realicen más intercambios en una pasada completa por la lista o hasta que la lista esté completamente ordenada.

La optimización del algoritmo de Burbuja Optimizada reduce la cantidad de comparaciones e intercambios necesarios, lo que lo hace más eficiente en comparación con el algoritmo de Burbuja básico.

Ambos algoritmos son algoritmos de ordenamiento simples y fáciles de implementar, pero la Burbuja Optimizada es más eficiente en términos de tiempo de ejecución, especialmente en listas grandes.

2.3 Analisis de casos

Algoritmo de Burbuja:

Mejor caso:

El mejor caso ocurre cuando la lista ya está completamente ordenada. En este escenario, el algoritmo de Burbuja solo realizará una pasada a través de la lista sin realizar ningún intercambio, ya que todos los elementos ya están en su posición correcta.

Ejemplo:

- Entrada: [1, 2, 3, 4, 5]
- Número de comparaciones: n - 1 = 4
- Número de intercambios: 0

Peor Caso:

El peor caso ocurre cuando la lista está ordenada en orden descendente. En este escenario, el algoritmo de Burbuja realizará el máximo número de comparaciones e intercambios.

Ejemplo:

- Entrada: [5, 4, 3, 2, 1]
- Número de comparaciones: (n-1) + (n-2) + ... + 1 = n(n-1)/2 = 10 comparaciones
- Número de intercambios: (n-1) + (n-2) + ... + 1 = n(n-1)/2 = 10 intercambios

Caso Promedio:

El caso promedio es más difícil de determinar, ya que depende de varios factores, incluidos los datos de entrada y la distribución de los elementos en la lista. En general, el caso promedio para el algoritmo de Burbuja es O(n2).

Algoritmo de Burbuja Optimizado:

Mejor caso:

El mejor caso para el algoritmo de Burbuja Optimizada ocurre cuando la lista ya está completamente ordenada. Similar al caso de Burbuja básico, en este escenario, solo se realizará una pasada a través de la lista sin realizar intercambios.

Ejemplo:

- Entrada: [1, 2, 3, 4, 5]
- Número de comparaciones: n - 1 = 4
- Número de intercambios: 0

Peor Caso:

El peor caso para el algoritmo de Burbuja Optimizada ocurre cuando la lista está ordenada en orden descendente. En este escenario, aunque se reduzca el número de comparaciones, aún se realizarán intercambios.

Ejemplo:

- Entrada: [5, 4, 3, 2, 1]
- Número de comparaciones: (n-1) + (n-2) + ... + 1 = n(n-1)/2 = 10 comparaciones
- Número de intercambios: (n-1) + (n-2) + ... + 1 = n(n-1)/2 = 10 intercambios

Caso Promedio:

Al igual que con el algoritmo de Burbuja, el caso promedio para el algoritmo de Burbuja Optimizada también es O(n2), aunque tiende a ser más rápido que el Burbuja básico debido a su optimización.

Y como se muestra en la parte anterior los algoritmos de ordenamiento tienen 2 complejidades según sea el caso cuando es su mejor caso en ambos algoritmos su complejidad es de O(n) y para los otros dos casos ya sea para el peor o el promedio la complejidad de ambos algoritmos es de O(n2).

2.4 Comparación de resultados

Algoritmo de Burbuja:

- Mejor caso:

```
Seleccione un método de ordenamiento:
1. Burbuja
2. Burbuja Optimizada
3. Salir
Ingrese su elección: 1
Ingrese el tamaño de la lista: 5
Ingrese el elemento 1: 1
Ingrese el elemento 2: 2
Ingrese el elemento 3: 3
Ingrese el elemento 4: 4
Ingrese el elemento 5: 5
Lista ordenada usando Burbuja: [1, 2, 3, 4, 5]
Tiempo de ejecución: 0.0000064373 segundos
```

- Caso promedio:

```
Seleccione un método de ordenamiento:
1. Burbuja
2. Burbuja Optimizada
3. Salir
Ingrese su elección: 1
Ingrese el tamaño de la lista: 5
Ingrese el elemento 1: 3
Ingrese el elemento 2: 2
Ingrese el elemento 3: 4
Ingrese el elemento 4: 1
Ingrese el elemento 5: 5
Lista ordenada usando Burbuja: [1, 2, 3, 4, 5]
Tiempo de ejecución: 0.0000073910 segundos
```

- Peor caso:

```
Seleccione un método de ordenamiento:
1. Burbuja
2. Burbuja Optimizada
3. Salir
Ingrese su elección: 1
Ingrese el tamaño de la lista: 5
Ingrese el elemento 1: 5
Ingrese el elemento 2: 4
Ingrese el elemento 3: 3
Ingrese el elemento 4: 2
Ingrese el elemento 5: 1
Lista ordenada usando Burbuja: [1, 2, 3, 4, 5]
Tiempo de ejecución: 0.0000109673 segundos
```

Algoritmo de Burbuja Optimizado:

- Mejor caso:

```
Seleccione un método de ordenamiento:
1. Burbuja
2. Burbuja Optimizada
3. Salir
Ingrese su elección: 2
Ingrese el tamaño de la lista: 5
Ingrese el elemento 1: 1
Ingrese el elemento 2: 2
Ingrese el elemento 3: 3
Ingrese el elemento 4: 4
Ingrese el elemento 5: 5
Lista ordenada usando Burbuja Optimizada: [1, 2, 3, 4, 5]
Tiempo de ejecución: 0.0000059605 segundos
```

- Caso promedio:

```
Seleccione un método de ordenamiento:
1. Burbuja
2. Burbuja Optimizada
3. Salir
Ingrese su elección: 2
Ingrese el tamaño de la lista: 5
Ingrese el elemento 1: 3
Ingrese el elemento 2: 2
Ingrese el elemento 3: 4
Ingrese el elemento 4: 1
Ingrese el elemento 5: 5
Lista ordenada usando Burbuja Optimizada: [1, 2, 3, 4, 5]
Tiempo de ejecución: 0.0000085831 segundos
```

- Peor caso:

```
Seleccione un método de ordenamiento:
1. Burbuja
2. Burbuja Optimizada
3. Salir
Ingrese su elección: 2
Ingrese el tamaño de la lista: 5
Ingrese el elemento 1: 5
Ingrese el elemento 2: 4
Ingrese el elemento 3: 3
Ingrese el elemento 4: 2
Ingrese el elemento 5: 1
Lista ordenada usando Burbuja Optimizada: [1, 2, 3, 4, 5]
Tiempo de ejecución: 0.0000116825 segundos
```

Como conclusión de estos resultados a la hora de ejecutar los algoritmos obtengo que aunque sea un arreglo cortó el que se tiene que ordenar a la hora de comparar los tiempos realizados tanto por ordenamiento de burbuja como burbuja optimizada no es mucha la diferencia por ejemplo en el mejor caso la diferencia es solo de 0.0000004768 segundos siendo el ordenamiento por burbuja optimizado el más rápido, en el caso promedio la diferencia es solo de 0.0000011921 segundos siendo el ordenamiento por burbuja el más rápido, en el peor caso la diferencia es solo de 0.0000007152 segundos siendo el ordenamiento por burbuja el más rápido, teniendo estos resultados claros vemos como el ordenamiento por burbuja es más rápido en 2 de los casos planteados sin embargo esto se debe al tamaño del arreglo ya que en casos en donde el arreglo es mucho más grande el algoritmo de burbuja optimizada es mucho más eficiente.

3 Conclusiones

Como conclusión de esta práctica obtengo que estos algoritmos de ordenamiento sirven en distintas situaciones por ejemplo en el caso de ordenamiento por burbuja es muy eficiente en arreglos cortos o que no requiera muchos intercambios y comparaciones para ordenar, por otro lado el ordenamiento por burbuja optimizado es más eficiente en arreglos de una mayor longitud, pero en resumen ambos algoritmos son de utilidad a la hora de ordenar.