PERBANDINGAN RUNNING TIME BUBBLE SORT DENGAN MERGE SORT

Daniel Septyadi (1301180009)
Muhammad Haidir Ali (1301180205)
Gilang Muhamad Rizky (1301180286)
Randika Dwi Maulana Rasyid (1301180361)

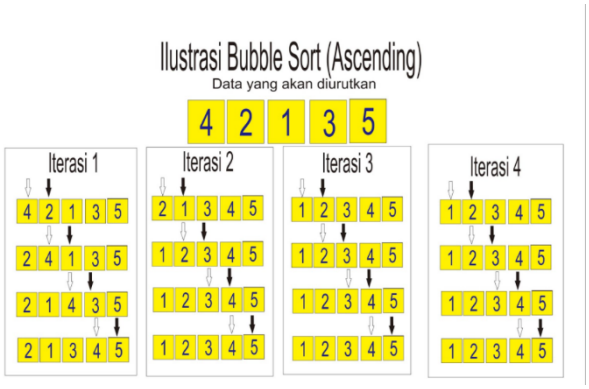
PENDAHULUAN

Pada pembuatan tugas besar ini, kami membuat dua algoritma sorting, yaitu Bubble Sort dan Merge Sort. Kemudian kedua algoritma tersebut dibandingkan berdasarkan running timenya dan mencari running time yang terbaik dari kedua algoritma tersebut. Perbandingan dilakukan dengan cara merunning kedua algoritma jika jumlah data yang dibandingkan sebanyak 10, 100, 1000, 10000 dan 100000. Kemudian hasil perbandingan dari kedua algoritma tersebut divisualisasikan oleh sebuah chart.
Kata kunci: Merge Sort, Bubble Sort, Running Time

1. PENJELASAN PROGRAM

1.1 Bubble Sort

Bubble sort merupakan algoritma yang berjalan secara berulang ulang dan dalam perulangannya melakukan perbandingan data dan menukar posisi dari data yang dibandingkan sampai tidak ada lagi data yang ditukar pada setiap iterasi.
Berikut adalah ilustrasi dari bubble sort :



Gambar 1.1 Ilutstrasi Bubble Sort

```
BubbleSort(array)
for i=length(array)-1 downto 0 do
        for j=1 to i do
                if array[j-1] > array[j] then
                swap(array[j-1],array[j])
                end if
        end for
end for
```

Bubble sort memiliki C(n), T(n) dan Kelas Efisiensi sebagai berikut :

$$C(n) = C_{op}n^2$$
$$T(n) \in O(n^2)$$

Kelas efisiensi Kuadratik

Algoritma Bubble sort memiliki beberapa kelebihan diantaranya algoritma ini mudah untuk diimplementasikan kedalam sebuah code dan algoritmanya relative simple. Namun hanya saja algoritma ini tidak cocok untuk digunakan jika data yang ingin diurutkan memiliki skala yang besar.

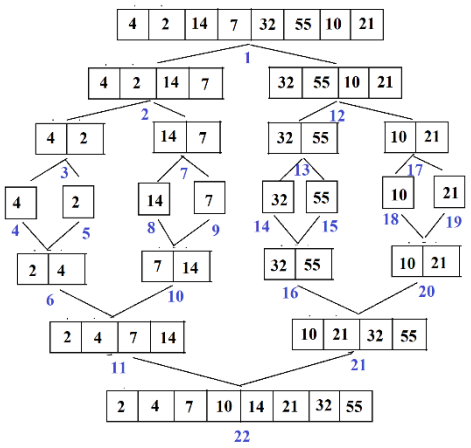
1.2 Merge Sort

Merge sort merupakan algoritma sorting yang bekerja dengan cara membagi sebuah struktur data menjadi lebih dari satu sub-bagian, lalu setiap anggota pada setiap sub-bagian akan dibandingkan dan posisinya akan ditukar jika memenuhi syarat. Berikut adalah pseudo code dari algoritma merge sort :

```
MergeSort(array,low,high)
        mid : integer
        if (low<high) then
                mid <- (low+high)/2
                MergeSort(array,low,mid)
                MergeSort(array,mid+1,high)
                Merge(array,low,high,mid)
        endif
```

Merge sort memiliki C(n), T(n) dan Kelas Efisiensi sebagai berikut :

$C(n) = C_{op} n \log n$
$T(n) \in O(n \log n)$
Kelas efisiensi logaritmik



Gambar 1.2 Ilustrasi Merge Sort

Algoritma Merge sort cenderung lebih efektif dibanding jenis algoritma sorting lainnya, hanya saja algoritma ini menggunakan memory yang lebih banyak dibandingkan dengan algoritma sorting lainnya.

II. STRATEGI ALGORITMA

Untuk hasil running time dari kedua sort yaitu merge sort dan bubble sort dengan n seperti dibawah ini, hasil ditunjukan dalam bentuk tabel.

n=10; n=100; n=1000; n=10000; n=100000.

Tabel 2.1 Perbandingan Running Time Merge Sort dan Bubble Sort

Jumlah data (n)	Running Time (microsecond)	
	Bubble Sort	Merge Sort
10	23.0	29.0
100	1499.0	585.0
1000	114619.0	4527.0
10000	11529591.0	62962.01
100000	1315844195.0	823086.0

III. FUNGSIONALITAS PROGRAM

```
#Bubble sort Function
def bubbleSort(list):
    temp = []
    for i in range(len(list)-1, 0, -1):
        for j in range(i):
            if list[j] < list[j+1]:
                temp = list[j]
                list[j] = list[j+1]
                list[j+1] = temp
    return list
```

Gambar 3.1 Function Bubble Sort

Function pertama yaitu function bubbleSort, yang berfungsi untuk mengurutkan sebuah data dengan metode bubble Sort.

```python
#Merge sort Function
def mergeSort(_list):
    if len(_list) < 2:
        return _list
    else:
        get_center=len(_list)//2
        left=_list[:get_center]
        right=_list[get_center:]
        mergeSort(left)
        mergeSort(right)
        i=0;j=0;k=0
        while i < len (left) and j < len (right):
            if left[i]>right[j]:
                _list[k]=left[i]
                i=i+1
            else:
                _list[k]=right[j]
                j=j+1
            k=k+1
        while i < len (left):
            _list[k]=left[i]
            i=i+1
            k=k+1

        while j < len (right):
            _list[k]=right[j]
            j=j+1
            k=k+1
```

Gambar 3.2 Function Merge Sort

Kemudian ada function merge Sort yang berfungsi untuk mengurutkan data dengan metode Merge Sort.

```python
def cariAngka(N):
    # seed random number generator
    seed(1)
    angka = []
    for i in range(N):
        value = randint(0, N)
        angka.append(value) #tangkap random number dalam list
    return angka
    #print("Angka acak adalah sebagai berikut : \n",angka)
```

Gambar 3.3 Function CariAngka

Function cari angka merupakan sebuah method yang mana method ini akan menggenerate sebuah angka sebanyak N.

```python
def getIteration():
    iteration = int(input("Jumlah Iteration: "))
    iteration_list = []
    for i in range(iteration):
        iteration_list.append(int(input("Input Batas Iteration ke "+str(i+1)+" : ")))

    return iteration_list
```

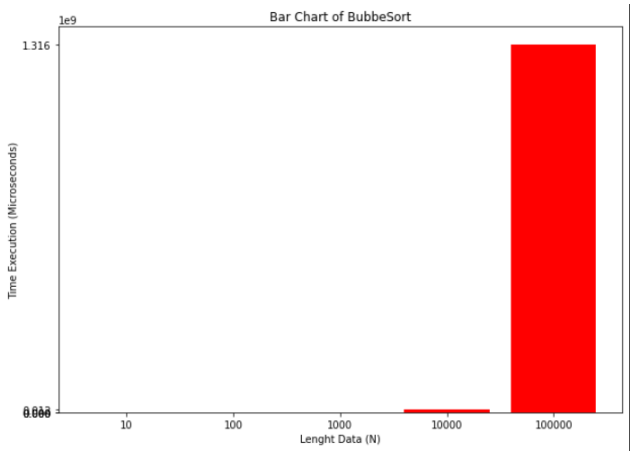
Gambar 3.4 Function getIteration

Method ini digunakan pada main program untuk menerima input dari user dalam menentukan jumlah iterasi yang akan dilakukan.

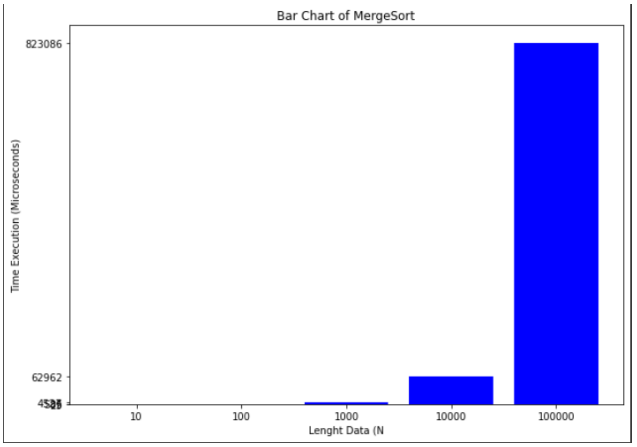
IV. SCREEN SHOOT OUTPUT PROGRAM

Output dari program sudah tertulis seperti yang ada pada **Tabel 2.1**. Berdasarkan **Tabel 2.1** dapat terlihat bahwa performa Merge Sort lebih baik dibandingkan dengan bubble sort. Dan performansi tersebut dapat terlihat dengan jelas saat n = 100.000 untuk bubble sort memakan waktu (Running Time) selama kurang lebih 21 menit, sedangkan Merge sort memiliki running time selama kurang dari 1 menit. Berdasarkan hasil tersebut membuktikan kekurangan dari bubble sort yaitu algoritma bubble sort tidak cocok digunakan untuk jumlah data yang besar.

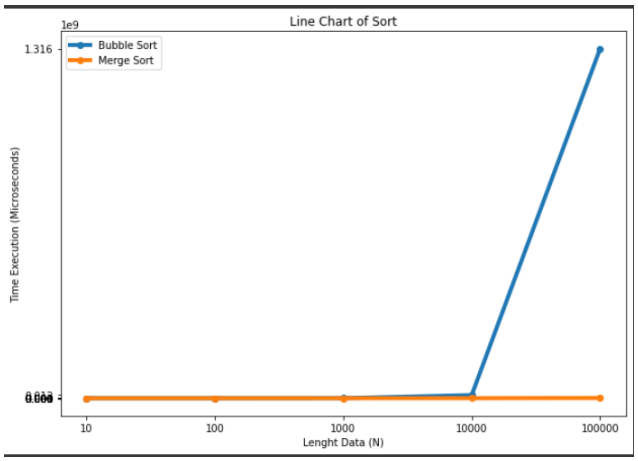
Berikut merupakan visualisasi dari hasil running kedua algoritma dan visualisasi dari perbandingan kedua algoritma dalam bentuk chart :



Gambar 4.1 Running Time Algoritma Bubble Sort



Gambar 4.2 Running Time Algoritma Merge Sort



Gambar 4.3 Perbandingan Bubble Sort dan Merge sort

Berdasarkan line chart pada gambar 4.3 dapat terlihat perbedaan running time yang sangat signifikan saat N (jumlah data) yang ingin diurutkan sebanyak 100.000 data.

V. LAMPIRAN

Link Code :
https://colab.research.google.com/drive/1J_CKIBg05EyCUjY1Ifx26KmAyoEzVu5w?usp=sharing

Link Video : https://youtu.be/C_BBhVCFNHw

VI. Kontribusi

Daniel Septyadi - Coding Program
Gilang Muhamad Rizky - Laporan
Muhammad Haidir Ali - PPT
Randika Dwi Maulana Rasyid - PPT

VI. REFERENSI

[1] Astrachan, O. (2003). Bubble sort: an archaeological algorithmic analysis. *ACM Sigcse Bulletin*, *35*(1), 1-5.
[2] Min, W. (2010, July). Analysis on bubble sort algorithm optimization. In *2010 International forum on information technology and applications* (Vol. 1, pp. 208-211). IEEE.
[3] Abidin. T, Zamanhuri. I. Struktur Data dan Algoritma Metode Pengurutan Mege Sort.