Sorting/Pengurutan Rata-Rata Nilai Matematika Siswa

Analisis Kompleksitas Algoritma

Nama Kelompok:

Andi Alfian Wahyudi	103012300190
Tristan Satria Perdana Hermawan	103012300253

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

2024

Daftar Isi

BAB 1 Pendahuluan

1.1 Latar Belakang Masalah

Di era digital yang semakin berkembang, manajemen data akademik menjadi bagian penting dari pendidikan. Bagaimana mengolah dan menganalisis data dengan cepat dan efisien adalah salah satu tantangan yang sering dihadapi, terutama ketika berurusan dengan jumlah data yang besar. Pengurutan data secara sistematis dalam hal nilai matematika siswa dapat membantu mengidentifikasi siswa dengan nilai terbaik dan siswa yang membutuhkan bantuan lebih lanjut.

1.2 Rumusan Masalah

1. Bagaimana algoritma sorting bisa di implementasikan terhadap data nilai siswa?
2. Algoritma sorting mana yang optimal untuk mengurutkan data nilai siswa?

1.3 Maksud dan Tujuan

Laporan ini bertujuan untuk menganalisis dan mengelola data nilai matematika siswa melalui penerapan algoritma sorting. Studi ini tidak hanya bertujuan untuk mengoptimalkan waktu dan efisiensi pengurutan, tetapi juga untuk memberikan rekomendasi penggunaan algoritma terbaik berdasarkan hasil analisis perbandingan.

BAB 2 Pembahasan

2.1 Algoritma

Algoritma adalah serangkaian langkah logis dan sistematis yang dirancang untuk menyelesaikan suatu masalah atau mencapai tujuan tertentu. Langkah-langkah ini harus terdefinisi dengan jelas, terstruktur, dan dapat diimplementasikan. Algoritma yang digunakan pada studi kasus ini yaitu algoritma sorting.

Algoritma Sorting, adalah metode untuk mengatur elemen-elemen dalam sebuah struktur data agar berada dalam urutan tertentu, seperti ascending (menaik) atau descending (menurun). Algoritma sorting yang akan di implementasikan untuk studi kasus ini yaitu *Quick Sort*, *Bubble Sort*, dan *Insertion Sort*.

Quick Sort, algoritma ini mengurutkan berdasarkan Divide and Conquer yang memilih elemen sebagai pivot dan membagi array yang diberikan di sekitar pivot yang dipilih dengan menempatkan pivot tersebut pada posisi yang benar dalam array yang diurutkan.

Bubble Sort, algoritma ini menyusun array atau string elemen ke dalam urutan menaik atau menurun. Algoritma ini mengurutkan elemen dengan membandingkan elemen yang berdekatan dari kiri ke kanan dalam array, dan menukar elemen jika tidak berurutan.

Insertion Sort, algoritma ini mengurutkan dengan memasukkan setiap elemen dari daftar yang tidak diurutkan ke posisi yang benar secara berulang di bagian daftar yang diurutkan.

Satu hal lagi yang perlu diperhatikan, yaitu Bahasa pemograman. Bahasa pemograman adalah bahasa yang digunakan untuk memberikan instruksi kepada komputer agar dapat melakukan tugas tertentu. Bahasa pemograman yang digunakan dalam studi kasus ini yaitu bahasa Python dengan alasan lebih mudah menerapkan studi kasus ini menjadi sebuah web application.

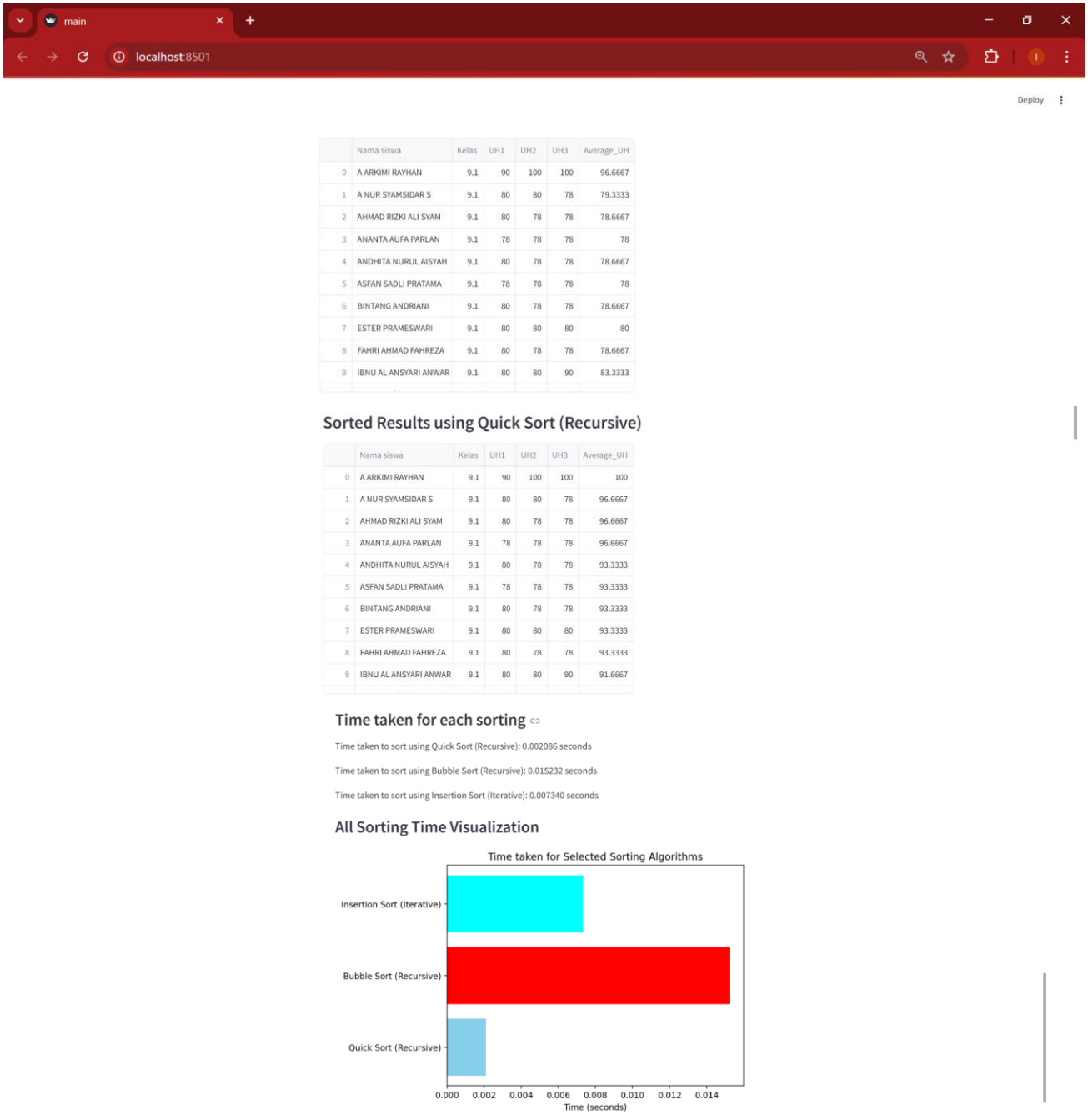
2.2 Implementasi

Implementasi algoritma studi kasus ini dilakukan sehingga dapat menganalisis dan mengelola data nilai matematika siswa melalui penerapan algoritma sorting. Pada studi kasus ini kami terapkan 3 algoritma sorting yang berbeda yaitu quick sort, bubble sort, dan insertion sort.

File Code dapat diakses pada link GitHub tersebut :
https://github.com/ZenZu-fian/tubes/blob/main/main.py

Visualisasi dari pengimplementasian algoritma :

Data yang digunakan untuk visualisasi implementasi algoritma dapat diakses pada link GitHub tersebut:
https://github.com/ZenZu-fian/tubes/blob/main/Data%20UH%20.xlsx



	Nama siswa	Kelas	UH1	UH2	UH3	Average_UH
0	A ARKIMI RAYHAN	9.1	90	100	100	96.6667
1	A NUR SYAMSIDAR S	9.1	80	80	78	79.3333
2	AHMAD RIZKI ALI SYAM	9.1	80	78	78	78.6667
3	ANANTA AUFA PARLAN	9.1	78	78	78	78
4	ANDHITA NURUL AISYAH	9.1	80	78	78	78.6667
5	ASFAN SADLI PRATAMA	9.1	78	78	78	78
6	BINTANG ANDRIANI	9.1	80	78	78	78.6667
7	ESTER PRAMESWARI	9.1	80	80	80	80
8	FAHRI AHMAD FAHREZA	9.1	80	78	78	78.6667
9	IBNU AL ANSYARI ANWAR	9.1	80	80	90	83.3333

Sorted Results using Quick Sort (Recursive)

	Nama siswa	Kelas	UH1	UH2	UH3	Average_UH
0	A ARKIMI RAYHAN	9.1	90	100	100	100
1	A NUR SYAMSIDAR S	9.1	80	80	78	96.6667
2	AHMAD RIZKI ALI SYAM	9.1	80	78	78	96.6667
3	ANANTA AUFA PARLAN	9.1	78	78	78	96.6667
4	ANDHITA NURUL AISYAH	9.1	80	78	78	93.3333
5	ASFAN SADLI PRATAMA	9.1	78	78	78	93.3333
6	BINTANG ANDRIANI	9.1	80	78	78	93.3333
7	ESTER PRAMESWARI	9.1	80	80	80	93.3333
8	FAHRI AHMAD FAHREZA	9.1	80	78	78	93.3333
9	IBNU AL ANSYARI ANWAR	9.1	80	80	90	91.6667

Time taken for each sorting

Time taken to sort using Quick Sort (Recursive): 0.002086 seconds

Time taken to sort using Bubble Sort (Recursive): 0.015232 seconds

Time taken to sort using Insertion Sort (Iterative): 0.007340 seconds

All Sorting Time Visualization

Kompleksitas Waktu

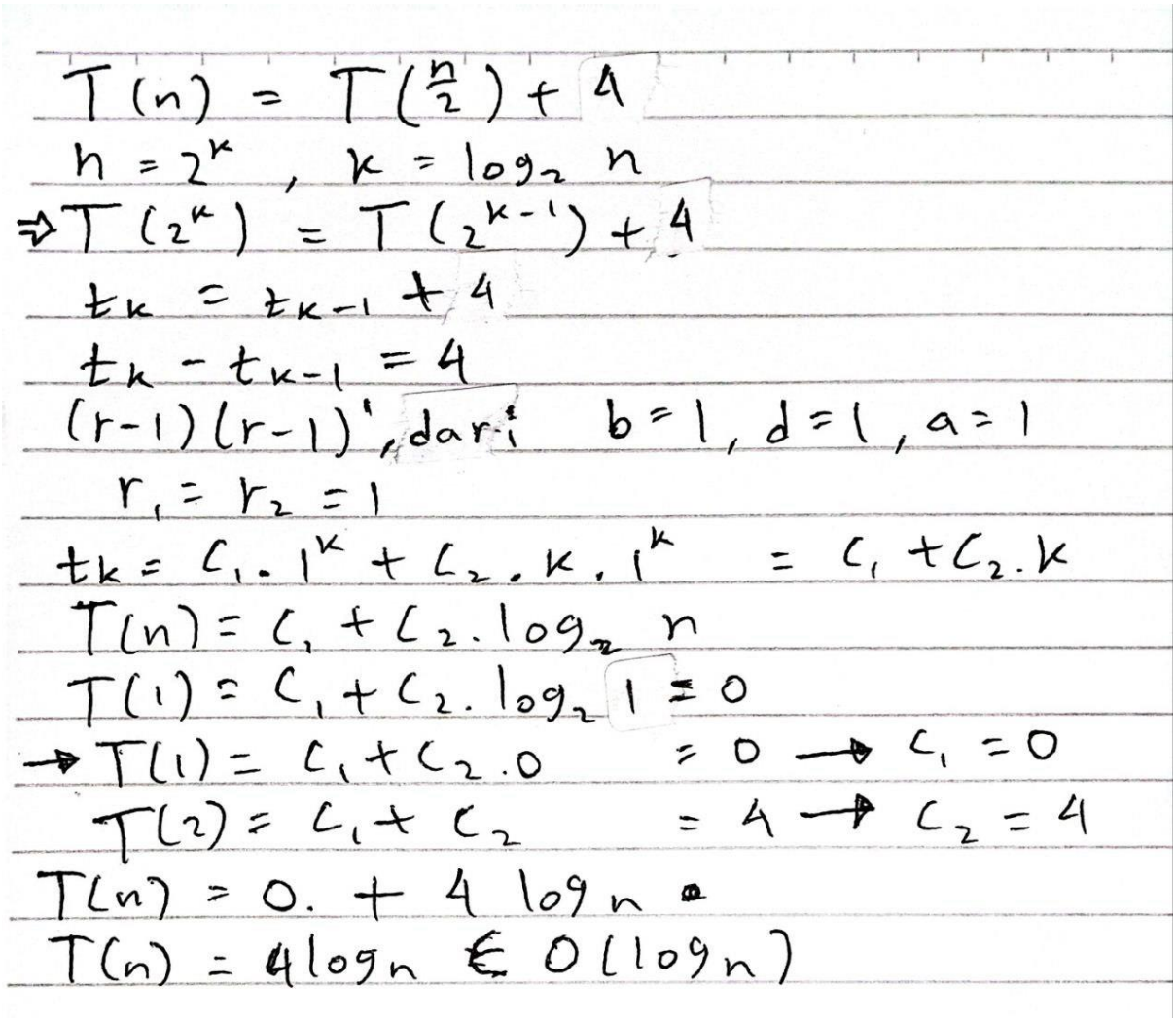
Dapat dilihat pada visualisasi implementasi algoritma, terdapat waktu eksekusi untuk setiap algoritma. Waktu eksekusi ini bergantung pada algoritma itu sendiri, biasanya juga disebut dengan kompleksitas waktu.

Quick Sort

```python
def quick_sort(arr):
    if len(arr) <= 1:
        return arr
    else:
        pivot = arr[len(arr) // 2]
        left = [x for x in arr if x > pivot]  # Change < to > for descending order
        middle = [x for x in arr if x == pivot]
        right = [x for x in arr if x < pivot]  # Change > to < for descending order
        return quick_sort(left) + middle + quick_sort(right)
```

Rumus T(n) :

$$T(n) = T\left(\frac{n}{2}\right) + 4$$



Bubble Sort

```python
# Recursive Bubble Sort function (Descending Order)
def bubble_sort(arr, n=345):

    # Base case: If the array size is 1, return
    if n == 1:
        return arr
    else:
        # One pass of bubble sort (Descending Order)
        for i in range(n - 1):
            if arr[i] < arr[i + 1]:  # Change > to < for descending order
                arr[i], arr[i + 1] = arr[i + 1], arr[i]

        # Recursive call for the remaining elements
        return bubble_sort(arr, n - 1)
```

Rumus T(n):

$$T(n)\begin{cases} 0, n = 0 \\ T(n-1) + n - 1, n \geq 0 \end{cases}$$

Perhitungan T(n) menggunakan metode subtitusi

T(n) = T(n – 1) + n – 1

T(n – 1) = (T(n – 2) + n – 1) + n – 1 = T(n – 2) + 2n – 2

T(n – 2) = (T(n – 3) + 2n – 2) + n – 1 = T(n – 3) + 3n – 3

→ T(n) = T(n – i) + in – i, akan ada saat n = i maka pada saat itu menjadi

$$T(n) = T(n-n) + n^2 - n = T(0) + n^2 - n = 0 + n^2 - n \rightarrow T(n) = n^2 - n$$
$$\in O(n^2)$$

Insertion Sort :

```python
# Iterative Insertion Sort function (Descending Order)
def iterative_insertion_sort(arr):
    n = len(arr)
    for i in range(1, n):
        key = arr[i]
        j = i - 1

        # Change > to < for descending order
        while j >= 0 and arr[j] < key:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
    return arr

```

Rumus T(n) :

$$T(n) = \sum_{i=1}^{n-1} 1 \sum_{j=i-1}^{n-2} 1 = \sum_{i=1}^{n-1} 1\,(n-2-(i-1)+1) = \sum_{i=1}^{n-1} n-i$$

$$= n \sum_{i=1}^{n-1} 1 - \sum_{i=1}^{n-1} i = n\,(n-1-1+1) - \frac{n-1\,(n-1+1)}{2}$$

$$= n^2 - n - \frac{n^2-n}{2} \in O(n^2)$$

BAB 3 Kesimpulan

Berdasarkan pengimplementasian diatas maka dapat kita ketahui bahwa Algoritma Quick Sort adalah algoritma paling efisien dari ketiga algoritma yang lain dengan waktu eksekusi tercepat, sementara Bubble Sort paling tidak efisien dengan waktu eksekusi terlama diantara ketiga algoritma, menjadikan algoritma Quick Sort pilihan utama untuk pengurutan data nilai siswa yang optimal.

Daftar Pustaka

https://youtu.be/cUKqsnLGQBw?si=CcsdiPtG17Gfp1z9

https://digilent.com/blog/recursive-sorting-algorithms/?srsltid=AfmBOoqSwv1NFRd9KzB9D4fQZhvPfJe8kqd1n4qHFADMAnwzpD9oLsmB

https://builtin.com/data-science/bubble-sort-time-complexity#:~:text=The%20bubble%20sort%20algorithm's%20average,complexity%3A%20O(n%C2%B2).

https://www.gramedia.com/literasi/pengertian-algoritma/?srsltid=AfmBOoptWaiEAecTfc2dXj1yrvXpKj-F7zfEsdIube1HNhODA1pVpgm3#google_vignette

https://www.geeksforgeeks.org/quick-sort-algorithm/