Mengurutkan Negara di Dunia Berdasarkan HDI Menggunakan Brute Force (Selection Sort) dan Divide and Conquer (Merge Sort)

Oleh:

Otniel Abiezer - 1301180469

Winico Fazry - 1301184316

Daffa Ulayya Suhendra - 1301184328

IF - 43 - 11

PROGRAM STUDI INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

BANDUNG

2021

Abstrak

HDI atau Human Development Index merupakan menjelaskan bagaimana penduduk dapat mengakses hasil pembangunan yang terdiri dari 3 bidang yaitu pendapatan, kesehatan, dan pendidikan, HDI bisa di ukur dalam bentuk bilangan. Maka dari itu kita menggunakan algoritma pengurutan dengan metode brute force yaitu selection sort dan divide and conquer yaotu merge sort untuk mengurutkan data HDI negara- negara dari terbesar hingga terkecil.

1. Pendahuluan

Laporan ini menggunakan judul "Mengurutkan Negara di Dunia Berdasarkan HDI Menggunakan Brute Force (Selection Sort) dan Divide and Conquer (Merge Sort)" dalam pengerjaan Tugas Besar ini.

Salah satu data yang biasanya dipakai untuk mengukur keberhasilan pembangunan di suatu daerah atau negara adalah HDI (Human Development Index) atau dalam Bahasa Indonesia adalah IPM (Indeks Pembangunan Manusia). HDI adalah indeks yang menjelaskan bagaimana penduduk dapat mengakses hasil pembangunan. Biasanya HDI tersusun dari 3 dimensi, yaitu umur panjang dan hidup sehat, pengetahuan, dan standar hidup layak (Badan Pusat Statistika, 2021). Hasil dari 3 dimensi tersebut menghasilkan suatu bilangan indeks dengan skala maksimal 100 berupa bilangan riil. Nilai indeks inilah yang akan dipakai untuk mengetahui sudah seberapa maju pembangunan di daerah tersebut sehingga pemerintah dapat mengetahui tindakan apa saja yang akan dilakukan untuk mengambil kebijakan berikutnya.

2. Dasar Teori

Algoritma Pengurutan adalah suatu algoritma yang menyelesaikan masalah pengurutan, yaitu menyusun kembali item/data yang diberikan senarai dengan berurut (dari terkecil ke besar atau sebaliknya) (Levitin, 2011).

Brute force merupakan pendekatan yang lempang (straightforward) untuk memecahkan suatu masalah. Strategi algoritma ini didasarkan pada pernyataan masalah (problem statement) dengan melibatkan definisi konsep (Saadah, 2021).

Algoritma Divide and Conquer adalah strategi pemecahan masalah yang besar dengan cara melakukan pembagian masalah yang besar tersebut menjadi beberapa bagian yang lebih kecil secara rekursif hingga masalah tersebut dapat dipecahkan secara langsung. Solusi yang didapat dari setiap bagian kemudian digabungkan untuk membentuk sebuah solusi yang utuh (Creatormedia, 2020).

Selection Sort adalah contoh algoritma pengurutan yang menerapkan metode Brute Force dengan menelusuri seluruh seluruh anggota senarai untuk mendapatkan nilai terbesar (jika mengurutkan dari terbesar) dan meletakkannya pada elemen pertama, lalu mencari yang terbesar selanjutnya dan diletakkan di elemen kedua, dan seterusnya sampai seluruh isi dari senarai terurut dari terbesar (Levitin, 2011).

Merge Sort adalah contoh algoritma pengurutan yang menerapkan metode Divide and Conquer dengan membagi senarai menjadi dua (*divide*), lalu mengurutkan masing-masing secara rekursif, setelah itu menggabungkan (*merge*) kedua senarai yang telah terurut menjadi satu senarai (Levitin, 2011).

3. Implementasi

Implementasi menggunakan bahasa pemrograman Python dan menggunakan library pandas untuk melakukan *import* dan *export* data dari dan ke Microsoft Excel dan library time untuk mendapatkan waktu eksekusi.

Implementasi algoritma pengurutan untuk mengurutkan data HDI dengan metode Brute Force adalah Selection Sort dan implementasi dengan metode Divide and Conquer adalah Merge Sort yang dapat dilihat di bawah

```
def SelectionSort(arr):

    for i in range(0,len(arr)):

        maks = i

        for j in range(i+1,len(arr)):

            if (arr[j][2] > arr[maks][2]):

                maks = j

            arr[maks], arr[i] = arr[i], arr[maks]
```

Selection Sort

```
def MergeSort(arr, low, high):

    if low >= high:

        return



    mid = (low + high)//2

    MergeSort(arr, low, mid)

    MergeSort(arr, mid + 1, high)

    Merge(arr, low, high, mid)



def Merge(arr, low, high, mid):

    low_copy = arr[low:mid + 1]

    high_copy = arr[mid+1:high+1]



    low_copy_idx = 0
```

```
    high_copy_idx = 0

    sorted_index = low



    while low_copy_idx < len(low_copy) and high_copy_idx < len(high_copy):

        if low_copy[low_copy_idx][2] >= high_copy[high_copy_idx][2]:

            arr[sorted_index] = low_copy[low_copy_idx]

            low_copy_idx = low_copy_idx + 1

        else:

            arr[sorted_index] = high_copy[high_copy_idx]

            high_copy_idx = high_copy_idx + 1



        sorted_index = sorted_index + 1



    while low_copy_idx < len(low_copy):

        arr[sorted_index] = low_copy[low_copy_idx]

        low_copy_idx = low_copy_idx + 1

        sorted_index = sorted_index + 1



    while high_copy_idx < len(high_copy):

        arr[sorted_index] = high_copy[high_copy_idx]

        high_copy_idx = high_copy_idx + 1

        sorted_index = sorted_index + 1
```

Merge Sort

4. Analisis

Hasil output dari kodingan tersebut yang dilakukan sebanyak 3 kali. Waktu eksekusi bisa mengalami sedikit perbedaan karena juga bergantung dari ketersediaan memori selain karena kompleksitas waktu.

```
PS D:\Kuliah Online\Semester 6\Strategi Algoritma\Tubes> python sort_hdi.py
Sort Negara , banyak data =  186
Waktu Eksekusi MergeSort =  0.0009965896606445312  detik
Waktu Eksekusi SelectionSort =  0.005682706832885742  detik


Sort SubNegara (Setara Provinsi), banyak data =  1750
Waktu Eksekusi MergeSort =  0.036077022552490234  detik
Waktu Eksekusi SelectionSort =  0.9210348129272461  detik
```

```
PS D:\Kuliah Online\Semester 6\Strategi Algoritma\Tubes> python sort_hdi.py
Sort Negara , banyak data =  186
Waktu Eksekusi MergeSort =  0.0009965896606445312  detik
Waktu Eksekusi SelectionSort =  0.04343414306640625  detik


Sort SubNegara (Setara Provinsi), banyak data =  1750
Waktu Eksekusi MergeSort =  0.03491926193237305  detik
Waktu Eksekusi SelectionSort =  0.8267743587493896  detik
```

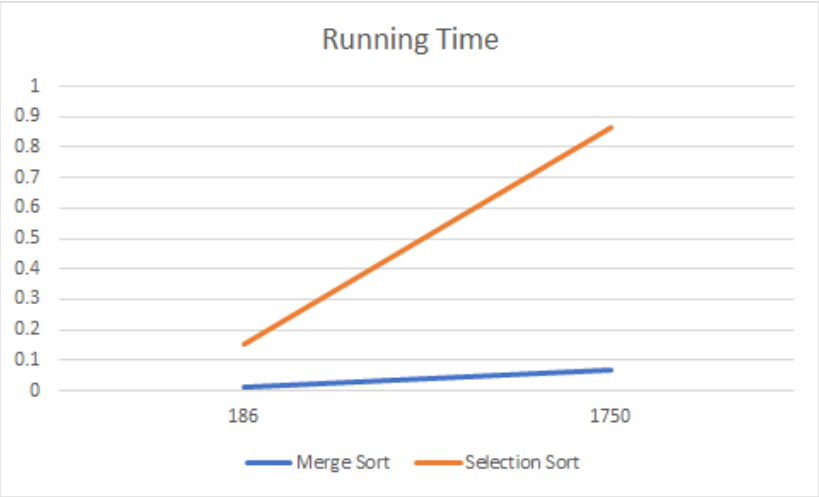
```
PS D:\Kuliah Online\Semester 6\Strategi Algoritma\Tubes> python sort_hdi.py
Sort Negara , banyak data =  186
Waktu Eksekusi MergeSort =  0.0015094280242919922  detik
Waktu Eksekusi SelectionSort =  0.036660194396972656  detik


Sort SubNegara (Setara Provinsi), banyak data =  1750
Waktu Eksekusi MergeSort =  0.13580608367919922  detik
Waktu Eksekusi SelectionSort =  0.8480522632598877  detik
```

Atau dapat diambil rata-ratanya sebagai tabel berikut:

Percobaan ke-	Banyak data = 186		Banyak data = 1750	
	Merge Sort	Selection Sort	Merge Sort	Selection Sort
1	0.0009	0.0568	0.0360	0.9210
2	0.0009	0.0434	0.0349	0.8267
3	0.0015	0.3666	0.1358	0.8480
Rata-rata	0.0011	0.1556	0.0689	0.8652

Berikut grafik pertumbuhan running time banyaknya data (n) terhadap waktu eksekusi



Kompleksitas waktu untuk Selection Sort dapat dilihat di bawah

$$T(n) = \sum_{i=0}^{n} \sum_{j=i+1}^{n} 1$$

$$= \sum_{i=0}^{n} n - i - 1 + 1$$

$$= \sum_{i=0}^{n} n - \sum_{i=0}^{n} i$$

$$= n\sum_{i=0}^{n} 1 - \sum_{i=0}^{n} i$$

$$= n(n+1) - (\frac{n(n+1)}{2})$$

$$= \frac{2n(n+1)}{2} - (\frac{n(n+1)}{2})$$

$$= \frac{2n(n+1) - n(n+1)}{2}$$

$$= \frac{n(n+1)}{2}$$

$$= \frac{1}{2}n^2 + \frac{1}{2} \in O(n^2)$$

Kompleksitas waktu untuk Merge Sort dapat dilihat di bawah

T(n) { a , n=1

{ 2T(n/2) + c n , n > 1

Substitusi

T(n) = 2T(n/2) + cn

= 2(2T(n/4) + cn/2) + cn = 4T(n/4) + 2cn

= 4(2T(n/8) + cn/4) + 2cn = 8T(n/8) + 3cn

= …

= 2^kT(n/2^k) + kcn

Ukuran terkecil jika n = 1

n/2^k = 1 → k = ^{2}log n

Sehingga

T(n) = nT(1) + c n log n

= na + c n log n

= O(n log n)

5. Kesimpulan

Setelah menganalisis kedua metode tersebut, dari hasil percobaan pada setiap metode dengan tiga kali uji coba, dengan menggunakan data yang sama, terlihat perbedaan dari waktu eksekusi kedua metode tersebut terlihat berbeda, apalagi untuk ukuran data yang lebih besar. Algoritma Selection Sort yang merupakan Brute Force melakukan pengurutan lebih lama daripada algoritma Merge Sort yang merupakan Divide and Conquer karena kompleksitas waktu Selection Sort lebih besar daripada Divide and Conquer, yaitu $O(n^2) > O(n \log n)$

Daftar Pustaka

Budan Pusat Statistika. (2021, Juni 10). *Badan Pusat Statistika*. Diambil kembali dari BPS Website: https://www.bps.go.id/subject/26/indeks-pembangunan-manusia.html

Creatormedia. (2020, Maret 21). *Creatormedia*. Diambil kembali dari Creatormedia web site: https://creatormedia.my.id/rangkuman-algoritma-divide-and-conquer/

Levitin, A. (2011). *Introduction to The Design and Analysis Algorithm 3rd Edition.*

Saadah, S. (2021). Lecture Note 2_CII2K3_Part 2 : Brute Force. Bandung, Jawa Barat, Indonesia: Telkom University.

Lampiran

Source code: https://github.com/Otniel113/TugasBesarSA