Menyelesaikan Kasus Algoritma Pengurutan Insertion Sort Menggunakan Metode Divide and Conquer

Alif Ranadian Nadhifah - 1301184255

Program Studi S1 Informatika
Fakultas Informatika
Universitas Telkom
Jl. Telekomunikasi no. 1, Dayeuhkolot, Bandung, Jawa Barat 40257
e-mail: alifnadhifah@student.telkomuniversity.ac.id

PENDAHULUAN

Pemrosesan suatu data atau yang lebih dikenal sebagai struktur data. Salah satu contoh struktur data ialah table. Tabel atau yang biasa disebut array dalam dunia pemrograman merupakan sekumpulan informasi yang dikumpulkan dalam type yang sama dan diletakkan berurutan sesuai urutan indeks yang telah didefinisikan. Pengurutan adalah contoh pemrosesan data array berype integer. Dalam makalah ini, akan dibahas algoritma pengurutan Insertion Sort salah satunya. Insertion sort merupakan pengurutan data array baik secara ascending maupun descending. Cara kerjanya adalah dengan membandingkan dua data (dimulai dari data pertama) apakah sudah terurut sesuai definisi lalu menyisipkan data kedua ke data pertama dan sebaliknya bila kedua data belum terurut. Pembandingan ini dilakukan hingga data ke-n sehingga dilakukan sebanyak nMax. Hal ini menyebabkan penggunaan algoritma Insertion Sort kurang efisien, sehingga diperlukannya metode divide and conquer dalam penggunaannya.

Kata kunci: 1. Array, 2. Insertion Sort, 3. Divide and Conquer

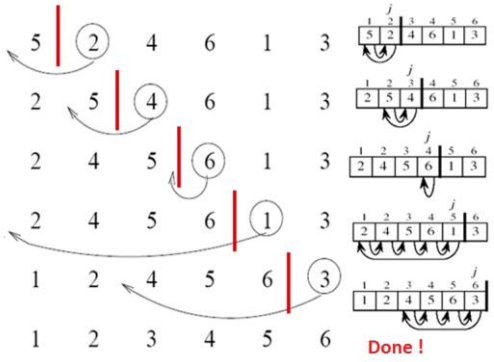
I. PENJELASAN PROGRAM

Program yang dibuat merupakan program yang berfungsi untuk memproses suatu data array bertipe integer untuk dapat diurutkan baik dari terkecil ke terbesar (ascending) ataupun dari terbesar ke terkecil (descending) [2]. Sehingga akan dimanfaatkan algoritma Insertion Sort dalam hal ini.

1.1 Algoritma Insertion Sort

Insertion Sort adalah salah satu algoritma sederhana untuk mengurutkan suatu data array. Algoritma ini dapat dibilang cukup efisien untuk digunakan dalam mengurutkan (sorting) baik secara ascending ataupun descending.

Cara kerja algoritma ini adalah dengan mengambil elemen array satu per satu, lalu menyisipkannya pada posisi yang benar. Pada array, list yang baru dan elemen sisanya berbagi tempat di array, meskipun cukup rumit. Untuk menghemat memori, implementasinya menggunakan pengurutan di tempat yang membandingkan elemen saat itu dengan elemen sebelumnya yang sudah diurut, lalu menukarnya terus hingga sampai pada posisi yang tepat [4]. Berikut ini merupakan gambar contoh cara kerja algoritma insertion sort.



***Gambar 1** [5]*

Dari gambar tersebut terlihat bahwa cara kerja algoritma insertion sort yang pertama adalah mengambil 2 elemen array pertama lalu bandingkan. Jika kedua elemen array telah memenuhi syarat sorting (ascending/descending) maka kedua elemen bertukar posisi. Kemudian ambil lagi 1 elemen array berikutnya, bandingkan, lalu tukar atau tetap pada posisinya. Hal ini dilakukan berulang kali hingga sejumlah n-1 dan semua elemen telah terletak pada posis yang terurut sesuai syarat.

1.2 Pseudocode Algoritma Insertion Sort

Berikut ini adalah contoh pseudocode Insertion sort dari suatu sumber.

```
procedure InsertionSort(input/output A : TabelInt,
                        input i, j : integer)
{ Mengurutkan tabel A[i..j] dengan algoritma Insertion Sort.
  Masukan: Tabel A dengan n elemen
  Keluaran: Tabel A yang terurut
}
Deklarasi:
   k : integer

Algoritma:
   if i < j then        { Ukuran(A)> 1}
      k←i
      InsertionSort(A, i, k)
      InsertionSort(A, k+1, j)
      Merge(A, i, k, j)
   endif
```

***Gambar 2** [3]*

1.3 Kompleksitas Waktu Insertion Sort

Algoritma Insertion Sort termasuk algoritma worst case dengan elemen arraynya selalu dalam urutan menurun. Perulangan yang dilakukan sebanyak (n-1) kal. Lalu dalam setiap iterasinya selalu menggerakkan semua elemen. Sehingga memiliki waktu komplesitas:

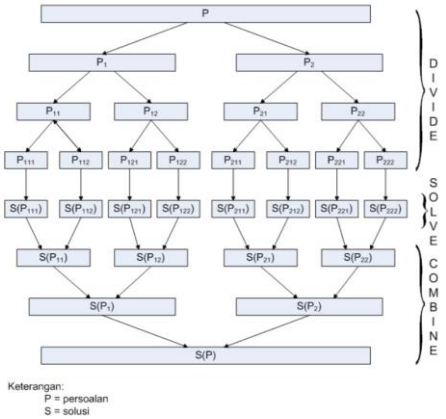
$$T(n) = O(n^2) \quad (1)$$

II. STRATEGI ALGORITMA

Pada umumnya Algoritma Insertion Sort cara kerjanya kurang efisien. Hal ini disebabkan oleh pertukaran 2 data satu per satu dalam setiap iterasi hingga n-1 dan setiap iterasinya akan menggerakkan semua elemen sehingga menghabiskan banyak waktu. Dan juga pada setiap iterasi terdapat pertukaran 1 elemen array berulang kali karena belum terletak pada posisi yang tepat. Untuk mengefisienkan hasil dari algoritma insertion sort, maka akan digunakan metode divide and conquer dalam program yang dirancang agar hasil yang diperoleh menghasilkan hasil iterasi sejumlah n Panjang array – 1 dan tidak terjadi perulangan 1 elemen berulang kali pada setiap iterasi.

2.1 Algoritma Divide and Conquer

Divide and conquer merupakan sebuah metode dalam menyelesaikan suatu masalah dengan cara membagi(divide), mengatasi(conquer), dan menggabungkan(combine). Divide artinya membagi suatu masalah menjadi beberapa sub-masalah yang memiliki kemiripan dengan masalah utama, namun sub-masalah ini berukuran lebih kecil. Conquer artinya menyelesaikan masing-masing sub-masalah (secara rekursif). Sedangkan Combine artinya menggabungkan solusi masing-masing sub-masalah sehingga membentuk 1 solusi masalah utama pada awalnya. [3]



***Gambar 3** [3]*

Gambar tersebut menunjukkan cara kerja divide and conquer pada umumnya. Diawali dengan melakukan divide atau memecah permasalahan utama menjadi beberapa sub-masalah yang lebih kecil. Lalu melakukan conquer atau menyelesaikan solusi dari masing-masing sub-masalah. Dan pada akhirnya melakukan combine atau penggabungan solusi sub-masalah sub-masalah menjadi satu solusi utama.

Metode divide and conquer ini dapat digunakan pada berbagai masalah. Salah satunya adalah pada masalah algoritma pengurutan Insertion Sort. Pendekatan divide and conquer pada algoritma Insertion Sort ini yaitu mudah membagi, namun sulit menggabung (easy split/hard join) . [3]

4 12 3 9 1 21 2 5

4 12 3 9 1 2 5 21

4 12 3 9 1 2 5 21

4 12 3 1 2 5 9 21

4 12 1 2 3 5 9 21

4 1 2 3 5 9 12 21

1 2 3 4 5 9 12 21

***Gambar 4** [3]*

Gambar tersebut menunjukkan cara kerja divide and conquer pada algoritma insertion sort. Dapat disimpulkan bahwa pada iterasi pertama lakukan divide atau mengambil 2 array pertama lalu conquer atau cari solusi untuk sorting dan combine atau gabungkan dengan seluruh elemen array, kemudian lakukan iterasi selanjutnya hinggan iterasi n-1 dengan n adalah jumlah Panjang array.

2.2 Pseudocode Algoritma Insertion Sort Dengan Divide and Conquer

Berikut ini adalah gambar contoh pseudocode divide and conquer pada umumnya dan pseudocode divide and conquer pada algoritma Insertion Sort dari berbagai sumber.

```
procedure DIVIDE_and_CONQUER(input n : integer)
{ Menyelesaikan masalah dengan algoritma D-and-C.
  Masukan: masukan yang berukuran n
  Keluaran: solusi dari masalah semula
}
Deklarasi
     r, k : integer
Algoritma
  if n ≤ n0 then  {ukuran masalah sudah cukup kecil }
     SOLVE upa-masalah yang berukuran n ini
  else
     Bagi menjadi r upa-masalah, masing-masing berukuran n/r
     for masing-masing dari r upa-masalah do
        DIVIDE_and_CONQUER(n/k)
     endfor
     COMBINE solusi dari r upa-masalah menjadi solusi masalah semula }
  endif
```

***Gambar 5** [3]*

```
procedure InsertionSort(input/output A : TabelInt,
                        input i, j : integer)
{ Mengurutkan tabel A[i..j] dengan algoritma Insertion Sort.
  Masukan: Tabel A dengan n elemen
  Keluaran: Tabel A yang terurut
}
Deklarasi:
   k : integer

Algoritma:
   if i < j then          { Ukuran(A)> 1}
     k←i
     InsertionSort(A, k+1, j)
     Merge(A, i, k, j)
   endif
```

***Gambar 6** [3]*

2.3 Kompleksitas Waktu Algoritma Insertion Sort Dengan Divide and Conquer

Algoritma Insertion Sort dengan Divide and conquer termasuk algoritma best case dengan elemen arraynya sudah terurut Perulangan yang dilakukan sebanyak (n-1) kal. Namun dalam setiap iterasinya tidak menggerakkan semua elemen dan menghasilkan waktu eksekusi sebanyak 0 kali. Sehingga memiliki waktu komplesitas:

$$T(n) = O(n) \qquad (2)$$

III. FUNGSIONALITAS PROGRAM

Program Insertion Sort menggunakan metode divide and conquer ini dibuat menggunakan Bahasa python v.3.7 dan text editor yang digunakan ialah Spyder v.4.1.2. Di dalam ini dibuat pendefisian function insertionSort1, function insertionSort2, dan program utama untuk dapat mengeksekusi program dengan baik.

Pendefinisian function insertionSort1 digunakan untuk mendefinisikan algoritma insertion sort secara ascending dengan divide and conquer. Seperti pada gambar

```python
#function insertion sort algorithm ascending
def insertionSort1(list, n):
    print("")
    print("Output Array: ", list, '(iterasi ke-0)')
    #Algoritma Utama Insertion Sort ascending
    for i in range(1,n):
        value = list[i]
        j = i - 1
        #Algoritma Utama Divide & Conquer
        while j >= 0 and list[j] > value:
            list[j+1] = list[j]
            j -= 1
        list[j+1] = value
        print("Output Array: ", list, '(iterasi ke-',i,')')
```

Gambar 7

Untuk pendefinisian function insertionSort2 digunakan untuk mendefinisikan algoritma insertion sort secara descending dengan divide and conquer. Seperti pada gambar

```python
def insertionSort2(list, n):
    print("")
    print("Output Array: ", list, '(iterasi ke-0)')
    #Algoritma Utama Insertion Sort descending
    for i in range(1,n):
        value = list[i]
        j = i - 1
        #Algoritma Utama Divide & Conquer
        while j >= 0 and list[j] < value:
            list[j+1] = list[j]
            j -= 1
        list[j+1] = value
        print("Output Array: ", list, '(iterasi ke-',i,')')

```

Gambar 8

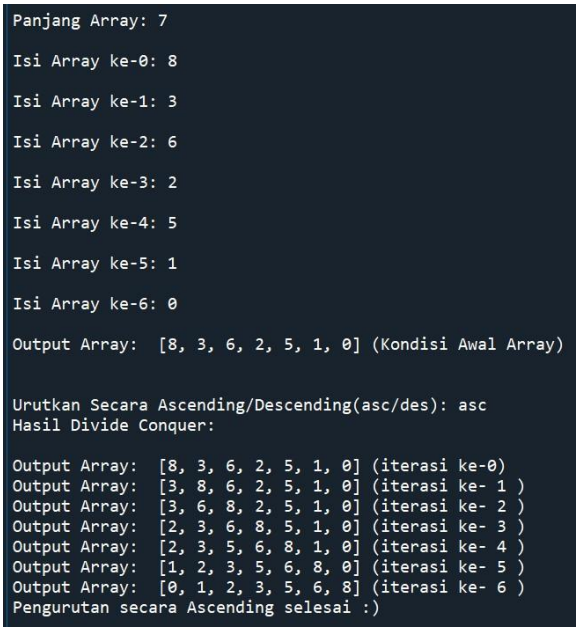
Sedangkan program utama berisi algoritma diantaranya inisialisasi variable yang akan digunakan, pembuatan array dengan input yang diinginkan oleh user saat dieksekusi, serta print value hasil program. Berikut merupakan gambar program utama:

```python
#start program with initialization and variable assignment
arr = []

k = int(input('Panjang Array: '))

for l in range(k):
    x = int(input(f'Isi Array ke-{l}: '))
    arr.append(x)

#print variable value
print('')
print("Output Array: ", arr,"(Kondisi Awal Array)")
print('')
ascordesc = str(input('Urutkan Secara Ascending/Descending(asc/des): '))
if (ascordesc=='asc'):
    print('Hasil Divide Conquer: ')
    insertionSort1(arr, k)
    print('Pengurutan secara Ascending selesai :)')
elif (ascordesc=='des'):
    print('Hasil Divide Conquer: ')
    insertionSort2(arr, k)
    print('Pengurutan secara Descending selesai :)')
```

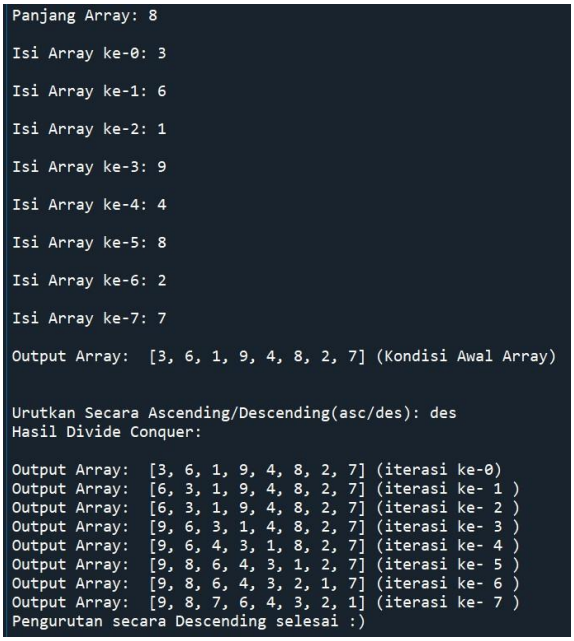
Gambar 9

IV. SCREEN SHOOT OUTPUT PROGRAM

Berikut ini merupakan hasil output program insertion sort menggunakan metode divide and conquer baik secara ascending ataupun descending:



Gambar 10



Gambar 11

V. REFERENSI

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/PTI/2013-2014/KU1072_Array_CPP.pdf. Diakses pada tanggal 24 April 2020

[2] https://socs.binus.ac.id/2019/12/30/insertion-sort/. Diakses pada tanggal 24 April 2020

[3] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Divide-and-Conquer-(2018).pdf. Diakses pada tanggal 24 April 2020

[4] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2008-2009/Makalah2008/Makalah0809-022.pdf. Diakses pada tanggal 24 April 2020

[5] http://student.blog.dinus.ac.id/mansyurhidayat/2017/03/29/insertion-sort/. Diakses pada tanggal 24 April 2020