Available *online* at : http://bit.ly/InfoTekJar

InfoTekJar : Jurnal Nasional Informatika dan Teknologi Jaringan

ISSN (Print) 2540-7597 | ISSN (Online) 2540-7600

Analisis Kompleksitas Diantara Algoritma Insertion Sort dan Selection Sort dan Diimplemntasikan dengan Bahasa Pemograman Java

Mulia Dhamma

Universitas Sumatera Utara

KEYWORDS

Insertion Sort; Selection Sort; Pemrograman Java

CORRESPONDENCE

Phone:

E-mail: mulia.dhamma@gmail.com

A B S T R A C T

Analisis kompleksitas algoritma merupakan hal yang sangat penting bagi orang ilmu komputer, dikarenakan dengan analisis kita akan bisa mengetahui kinerja performa dan keefisienan dari algoritma yang kita analisis tersebut. Algoritma pengurutan penyisipan(Insertion Sort) dan pemilihan(Selection Sort) akan menjadi pembahasan kita dengan mengimplementasikannya ke dalam bahasa pemograman java dengan mengukur kondisi terbaik(Best Case), terburuk(Worst Case), rata - rata(Average Case). Berdasarkan hasil dari analisis tersebut maka didaptkan algoritma pengurutan penyisipan(Insertion Sort) dan pemilihan(Selection Sort) memiliki efisiensi yang hampir sama, hanya untuk algoritma pengurutan penyisipan memiliki kelebihan apabila terdapat angka yang sudah hampir terurut.

PENDAHULUAN

Pada era globalisasi sekarang ini, perkembangan teknologi komputer sudah berkembang dengan sangat pesat, dan ini sudah pasti menjadi suatu hal yang sangat menguntungkan untuk manusia selaku penggunanya karena dengan teknologi komputer ini bisa memudahkan mereka dalam melakukan aktivitas secara efektif dan juga efisien dengan program - program yang ada didalam didalam teknologi tersebut. Seiring dengan perkembangan itu algoritma yang merupakan suatu bagian atau metode penyelesaian suatu masalah yang ada didalam program komputer itu masih tetap menjadi suatu permasalahan bagi ilmuan-ilmuan komputer yang ada diseluruh dunia.

Kata algoritma pada awalnya berasal dari seorang penulis buku di arab yang bernama Mohammed ibn-Musa al-Khwarizmi. Algoritma dapat diibaratkan dengan contoh seorang teman yang baru saja tiba di bandara dan ingin menuju ke rumah kita. Berdasarkan kondisi tersebut, maka akan ada 4 algoritma untuk menuju ke rumah kita yaitu dengan algoritma taksi, algoritma telepon, algorima sewa mobil dan algoritma bus. Semua algoritma tersebut sebenarnya memiliki tujuan yang sama, namun pastinya akan terdapat beberapa perbedaan pada waktu dan biaya yang akan dikeluarkan oleh teman, sehingga kita harus memilih algoritma - algoritma yang terbaik sesuai dengan kondisi pada waktu tersebut.

Algoritma pengurutan(Sort Algorithm) merupakan salah satu algoritma yang penting pada sebuah program yang ada didalam komputer. Tujuan dari algoritma ini adalah untuk menyusun object dari yang sebelumnya berantakan/acak menjadi tersusun secara berurutan. Algoritma pengurutan ini terdiri dari 2 macam yaitu pengurutan internal dan pengurutan eksternal. Algoritma pengurutan internal ini memiliki bemacam-macam metode pengurutan diantaranya Buble Sort, Selection Sort, Insertion Sort, Shell Sort, Merge Sort, Radix Sort, Quick Sort, Heap Sort, dll. Efisiensi dalam algoritma pengurutan merupakan hal yang sangat penting untuk mendukung algoritma agar bekerja secara optimal sehingga harus dianalisis dengan baik. Analisis bisa dilakukan dengan 2 cara yaitu analisis ruang dan analisis waktu, untuk analisis ruang sudah jarang namun untuk analisis waktu selalu dilakukan. Untuk menganalisi algoritma pengurutan penyisipan dan pemilihan ini maka akan kita analisis dengan Big O dan akan diimplementasikan dengan bahasa pemograman java.

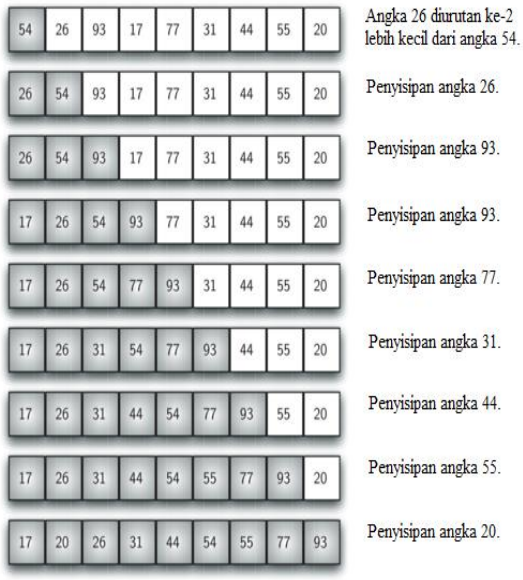
Tujuan dari pembahasan adalah untuk menganalisis kompleksitas algoritma pengurutan penyisipan dan pemilihan sehingga akan diketahui kondisi terbaik(Best Case), terburuk(Worst Case) dan rata(Average Case).

Kajian Literatur Dan Pengembangan Hipotesa

Insertion sort merupakan algoritma pengurutan penyisipan yang termasuk salah satu algoritma pengurutan yang sederhana.

Attribution-NonCommercial 4.0 International. Some rights reserved

Algoritma ini berkerja dengan cara dimana angka yang diambil untuk dijalankan duluan berada pada posisi ke 2 pada suatu deretan angka yang akan diurut tersebut. Angka yang telah diambil tersebut kemudian dicek ulang satu per satu semua angka yang berada pada posisi kiri dari angka yang telah diambil dan apabila angka yang diambil menemukan adanya angka yang lebih besar dari angka hasil pengecekan semua angka yang berada dibagian kiri, maka angka yang diambil tersebut langsung disisipkan kedalam. Algoritma pengurutan penyisipan ini akan berhenti apabila angka yang diambil sudah mencapai akhir dari angka yang ada pada deretan tersebut.



Gambar 1. Contoh Metode Pengurutan Penyisipan(Insertion Sort)

Pada gambar diatas kita melihat angka 26 dicek dengan angka 54 yang berada diposisi 1 maka langsung disisipkan keurutan ke-1 dan berulang seterusnya sampai dengan angka yang terkahir yaitu angka 20

PSEUDOCODE

```
u_sisip(a[1..n]){
    temp
    for(i = 1 < n; i++){
        for(j = i > 0; j--){
            if(a[j] < a[j-1]){
                temp = a[j]
                a[j] = a[j-1]
                a[j-1] = temp
            }
        }
    }
    return A;
}
```

function u_sisip((a[1..n]){				
temp	c1	1	1	c1
for(i = 1 < n; i++){	c2	n-1	n	c2n
for(j = i > 0; j--){	c2	(n-1)(n-1)	n^2	c2n^2
if(a[j] < a[j-1]){	c3	(n-1)(n-1)	n^2	c3n^2
temp = a[j]	c1	(n-1)(n-1)	n^2	c1n^2
a[j] = a[j-1]	c4	(n-1)(n-1)	n^2	c4n^2
a[j-1] = temp	c4	(n-1)(n-1)	n^2	c4n^2
}				
}				
}				
return a;	c5	1	1	c5
}				

Theoretical Running Time = (c1 + c5) + c2n + (c1 + c2 + c3 + 2c4)n^2
= O(n^2)

Kondisi terbaik(Best Case) hanya akan terjadi apabila deretan angka sudah terurut(Ascending), dikarenakan angka hanya melakukan pengecekan sekali saja atau tanpa penyisipan disebut O(n)

17	20	26	31	44	54	55	77	93

Kondisi terburuk(Worst Case) hanya akan terjadi apabila deretan angka terurut secara terbalik(Descending), dikarenakan angka melakukan pengecekan terhadap semua angka yang berada pada posisi kiri dari angka yang diambil tersebut atau disebut O(n^2).

93	77	55	54	44	31	26	20	17

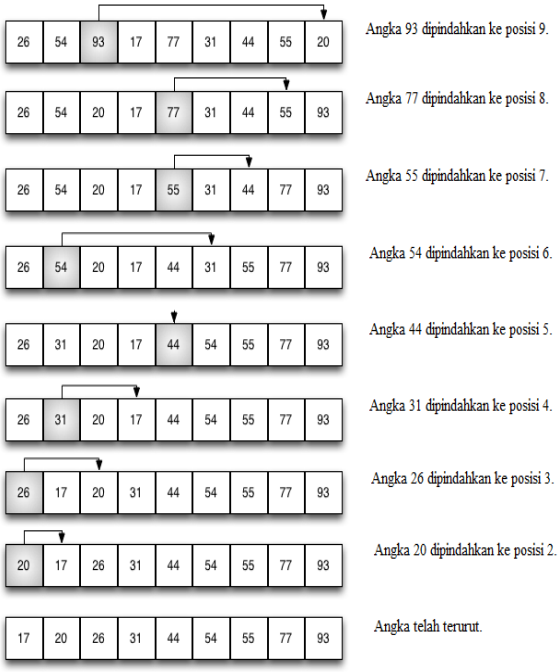
Kondisi rata-rata(Average Case) juga sama dengan kondisi terburuk(Worst case) juga disebut O(n^2).

54	26	93	17	77	33	44	55	20

Keuntungan dari algoritma pengurutan penyisipan yaitu penerapannya yang sederhana, apabila angka dari deretan sudah hampir terurut maka akan semakin cepat proses dari algoritma tersebut juga akan menjadi sangat efisien apabila deretan angka yang ingin diurut semakin sedikit, stabil prosesnya, bisa langsung mengurutkan angka yang baru saja dimasukan, cukup O(1) memori tambahan yang dibutuhkan.

Selection Sort merupakan algoritma pengurutan pemilihan yang juga sederhana pengunaannya. Algoritma ini juga merupakan perbaikan hasil dari algoritma pengurutan pengelumbungan(Bubble Sort). Algoritma ini cenderung lebih tidak stabil apabila dibandingkan dengan algoritma pengurutan penyisipan(Insertion Sort). Algoritma ini bekerja dengan cara dimana angka yang dimulai untuk diambil duluan adalah angka yang paling kecil apabila ingin mengurutkan deretan angka dari kecil ke besar(Ascending) dan yang paling besar apabila ingin mengurutkan deretan angka dari besar ke kecil(Descending) atau sering dinamakan dengan pivot. Setelah angka sudah diambil, angka tersebut langsung dipindahkan/digantikan dengan angka yang berada pada posisi paling kanan(Ascending) atau pada posisi paling kiri(Descending). Apabila angka yang paling besar ataupun kecil sudah menempati posisinya dengan benar, maka angka tersebut tidak akan dipindahkan/digantikan lagi. Proses

ini akan terus berlanjut sampai dengan habisnya angka yang pada deretan tersebut.



Gambar 2. Contoh metode pengurutan pemilihan(Selection Sort)

Pada gambar diatas kita melihat angka 93 merupakan angka yang paling besar sehingga angka tersebut dipindahkan ke posisi angka 20 dan berulang seterusnya sampai dengan n - 1 kali.

PSEUDOCODE

```
u_pilih(a[1..n]){
    for(i = 0 < n - 1; i++){
        index = i
        for(j=i + 1 < n; j++){
            if(a[j]        < a[index]){
                index = j
            }
        }
        smallerNumber = a[index]
        a[index] = a[i]
        a[i] = smallerNumber
    }
    return a;
}
```

function u_pilih(a[1..n]){				
for(i = 0 < n - 1; i++){	c1	n-1	n	c1n
index = i	c2	n-1	n	c2n
for(j=i + 1 < n; j++){	c1	(n-1)(n-1)	n^2	c1n^2
if(a[j] < a[index]){	c3	(n-1)(n-1)	n^2	c3n^2
index = j	c2	(n-1)(n-1)	n^2	c2n^2
}				
}				
smallerNumber = a[index]	c4	n-1	n	c4n
a[index] = a[i]	c5	n-1	n	c5n
a[i] = smallerNumber	c5	n-1	n	c5n
}				
return a;	c	1	1	c6
}				

Theoretical Running Time = c6 + (c1 + c2 + c4 + 2c5)n + (c1 + c2 + c3 + 2c4)n^2

= O(n^2)

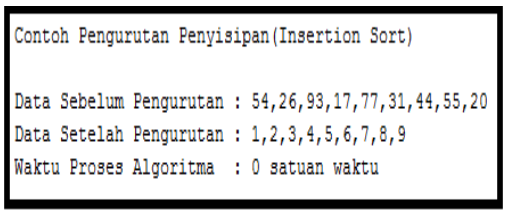
Kondisi terbaik (Best Case), terburuk(Worst Case), rata – rata (Average Case) mendapatkan hasil yang sama yaitu O(n^2) dikarenakan algoritma sudah pasti melakukan perputaran(looping) untuk melakukan pengecekan berulang - ulang terhadap semua angka yang masih belum terurut. Keuntungan dari algoritma pengurutan pemilihan(Selection Sort) yaitu mempunya peforma yang bagus apabila dibandingkan dengan beberapa algortima pengurutan yang kompleks untuk situasi tertentu, tidak membutuhkan kapasitas penyimpanan tambahan dikarenakan angka melakukan pergantian pada tempatnya.

IMPLEMENTASI

Implementasi untuk algoritma pengurutan penyisipan(Insertion Sort) dan algoritma pengurutan pemilihan(Selection Sort) akan menggunakan bahasa java.

Insertion Sort

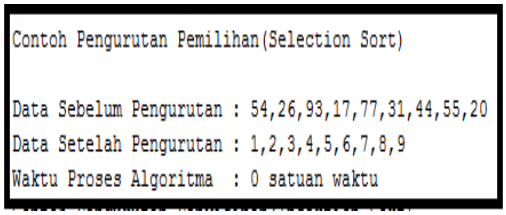
```
void  doInsertionSort(int[] input){
    int temp;
    for (int i = 1; i < input.length; i++){
        for(int j = i ; j > 0 ; j--){
            if(input[j] < input[j-1]){
                temp = input[j];
                input[j] = input[j-1];
                input[j-1] = temp;
            }
        }
    }
}
```



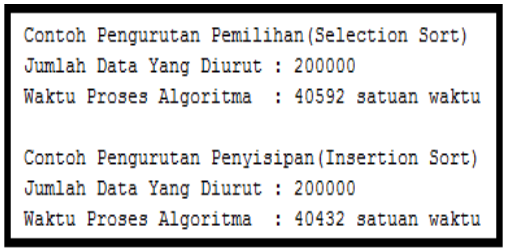
Gambar 1. Contoh program algoritma pengurutan penyisipan

Selection Sort

```
void  doSelectionSort(int[] input){
    for (int i = 0; i < input.length - 1; i++){
        int index = i;
        for(int j=i+1; j < input.length; j++){
            if(input[j] < input[index]){
                index = j;
            }
            int smallerNumber = input[index];
            input[index] = input[i];
            input[i] = smallerNumber;
        }
    }
}
```



Gambar 3. Contoh program algoritma pengurutan pemilihan



Gambar 4. Perbandingan algoritma pengurutan pemilihan(Selection Sort) dan pengurutan penyisipan(Insertion Sort) dimana angka dalam urutan posisi acak dengan N = 200000

KESIMPULAN

Algoritma pengurutan penyisipian(Insertion Sort) dan pemilihan(Selection Sort) merupakan algoritma yang cukup mudah untuk diimplementasikan dan hampir memilki persamaan dalam analisis kompleksitas karena masing - masing algoritma tersebut memiliki kompleksistas waktu O(n^2) untuk kejadian terburuk(Worst Case) namun untuk algoritma penyispian dapat bekerja lebih baik lagi dengan kompleksitas waktu O(n) apabila deretan angka sudah hampir dalam keadaan terurut.

REFERENSI

[1] *What is a computer algorithm?*, http://computer.howstuffworks.com/question717.htm, (24 Oktober 2015).

[2] A. Allain, *Sorting Algorithm Comparison*, http://www.cprogramming.com/tutorial/computersciencetheory/sortcomp.html, (24 October 2015).

[3] *The Insertion Sort*, http://interactivepython.org/courselib/static/pythonds/SortSearch/TheInsertionSort.html, (24 Oktober 2015).

[4] *The Selection Sort*, http://interactivepython.org/courselib/static/pythonds/SortSearch/TheSelectionSort.html, (25 Oktober 2015).

[5] G.Nataraja, *Program: Implement insertion sort in java*, http://www.java2novice.com/java-sorting-algorithms/insertion-sort/, (25 Oktober 2015).

[6] B.Peter, *Slightly Skeptical View on Sorting Algorithms*, http://www.java2novice.com/java-sorting-algorithms/selection-sort/, (25 Oktober 2015).

[7] R.Margaret, *Algorithm*, http://whatis.techtarget.com/definition/algorithm, (25 Oktober 2015).