Ghema Allan Ferdiansyah 140810170048

# Studi Kasus

## **Studi Kasus 2: SELECTION SORT**

Selection sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma selection sort, lakukan langkah-langkah berikut:

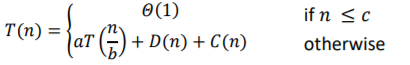
* Pelajari cara kerja algoritma selection sort

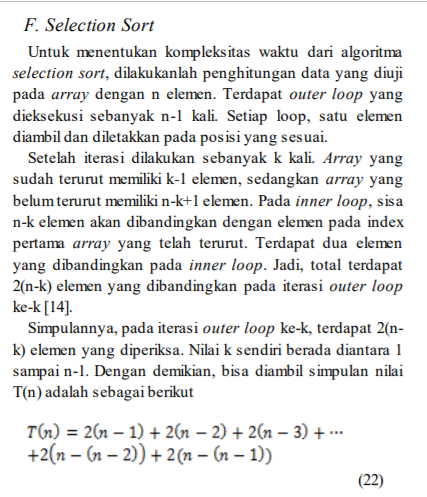
JAWABAN:

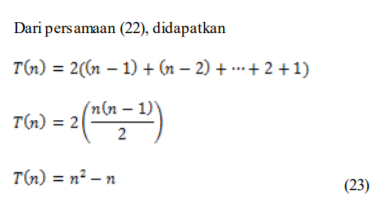
Algoritma selection sort adalah algoritma yang melakukan pengurutan secara berulang-ulang hingga didapatkan hasil pengurutan yang sesuai. Algoritma selection sort akan memindai nilai terkecil dari suatu kumpulan data dan jika ada, data teresebut akandiletakkan pada urutan pertama. Begitu selanjutnya untuk urutan kedua dan seterusnya. Cara kerja dari selection sort ini dalah sebagai berikut.

* Melakukan pengecekan dimulai dari data pertama hingga data ke-n.
* Menentukan data dengan indeks minimum (jika acending) atau maksimum (jika descending) dalam sebuah data tersebut.
* Menukarkan data dengan indeks minimum (jika ascending) atau maksimum (jika descending) dengan bilangan pertama (i = 1) dari data tersebut.
* Mengulangi langkah di atas untuk sisa data bilangan berikutnya (i = i +1) sampai didapatkan urutan yang sesuai.
* Tentukan T (n) dari rekurensi (pengulangan) selection sort berdasarkan penentuan rekurensi divide & conquer:

JAWABAN:







* Selesaikan persamaan rekurensi T(n) dengan **metode recursion-tree** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ

Jadi, Big-O, Big-Ω, dan Big-Θ masing-masing adalah n2

### **Studi Kasus 3: INSERTION SORT**

Insertion sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma insertion sort, lakukan langkah-langkah berikut:

* Pelajari cara kerja algoritma insertion sort

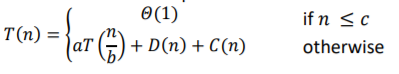
JAWABAN:

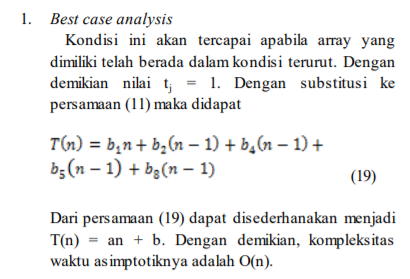
Insertion sort adalah sebuah algoritma pengurutan yang membandingkan dua elemen data pertama, mengurutkannya, kemudian mengecek elemen data berikutnya satu persatu dan membandingkannya dengan elemen data yang telah diurutkan. Karena algoritma ini bekerja dengan membandingkan elemen-elemen data yang akan diurutkan, algoritma ini termasuk pula dalam comparison-based sort.

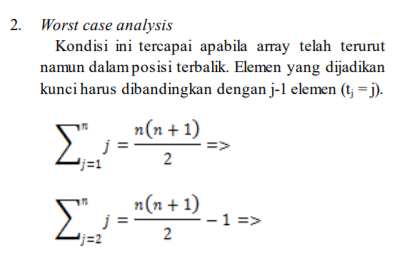
Ide dasar dari algoritma Insertion Sort ini adalah mencari tempat yang “tepat” untuk setiap elemen array, dengan cara sequential search. Proses ini kemudian menyisipkan sebuah elemen array yang diproses ke tempatnya yang seharusnya. Proses dilakukan sebanyak N-1 tahapan (dalam sorting disebut sebagai “pass“), dengan indeks dimulai dari 0.

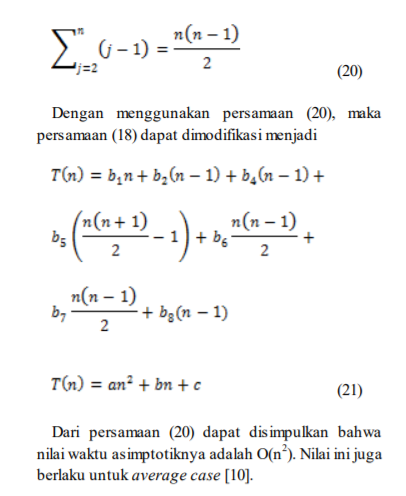
Proses pengurutan dengan menggunakan algoritma Insertion Sort dilakukan dengan cara membandingkan data ke-i (dimana i dimulai dari data ke-2 sampai dengan data terakhir) dengan data berikutnya. Jika ditemukan data yang lebih kecil maka data tersebut disisipkan ke depan sesuai dengan posisi yang seharusnya.

* Tentukan T(n) dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:









* Selesaikan persamaan rekurensi T(n) dengan **metode** **subtitusi** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ

Jadi, Big-O = n

Big-Ω = n2

Big-Θ n2

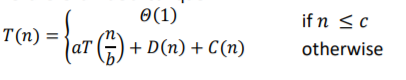
### **Studi Kasus 4: BUBBLE SORT**

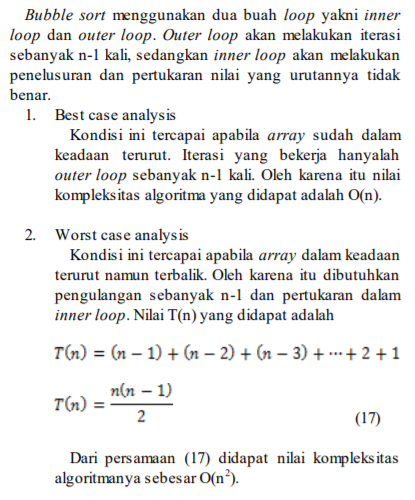
Bubble sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma bubble sort, lakukan langkah-langkah berikut:

* Pelajari cara kerja algoritma bubble sort

JAWABAN:

* Metode sorting termudah
* Cara pengurutannya : bandingkan dua data kemudian swap.
* Diberi nama “Bubble” karena proses pengurutan secara
* berangsur-angsur bergerak/berpindah ke posisinya yang tepat,
* seperti gelembung.
* Bubble Sort mengurutkan data dengan cara membandingkan elemen sekarang dengan elemen berikutnya.
* Ascending : Jika elemen sekarang lebih besar dari elemen berikutnya maka kedua elemen tersebut ditukar/swap.
* Descending : Jika elemen sekarang lebih kecil dari elemen berikutnya, maka kedua elemen tersebut ditukar/swap.
* Tentukan T(n) dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:





* Selesaikan persamaan rekurensi T(n) dengan **metode** **master** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ

Jadi, Big-O = n

Big-Ω = n2

Big-Θ n2