LAPORAN ALGORITMA PENGURUTAN KELOMPOK 3

Oleh :

Muvidha Fatmawati Putri	(21091397011)
Diah Ayuning Tyas	(21091397013)
Shandy Ilham Alamsyah	(21091397015)
Fisma Meividianugraha Subani	(21091397017)
Alvin Febrianto	(21091397031)

Fakultas Vokasi

D4 Manajemen Informatika

UNIVERSITAS NEGERI SURABAYA

TAHUN AJARAN 2021/2022

1. Merge Sort (Sortingan Paling Cepat) Berdasarkan Worst Case

Kasus terburuk merge sort terjadi apabila selama memanggil fungsi rekursif merge, nilai terbesar setiap elemen terletak pada larik atau division yang berbeda. hal ini memicu pengurutan yang dilakukan akan berpindah-pindah antar lariknya, seperti yang digambarkan pada kasus berikut :

(4 5 2 9 8 1 7 3)

Pass 1

(4 5 2 9) (8 1 7 3)

Pass 2

(4 5) (2 9) (8 1) (7 3)

Pass 3

(4) (5) (2) (9) (8) (1) (7) (3)

Pass 4

(4 5) (2 9) (1 8) (3 7)

Pass 5

(2 4 5 9) (1 3 7 8)

(1 2 3 4 5 7 8 9)

Big O merge sort adalah O (n log n). Setiap melewati array dibagi menjadi dua sama besar. Jika terdapat 8 elemen dalam larik, dibutuhkan 5 lintasan untuk sampai ke hasil dengan membagi subarray elemen menjadi (2–4–8) seperti itu cara kerja n log (n). Merge sort termasuk algoritma yang sangat efisien dalam penggunaannya karena setiap list selalu dibagi bagi menjadi list yang lebih kecil, kemudian digabungkan lagi sehingga tidak perlu melakukan banyak perbandingan. Apabila semakin besar subarray, maka semakin lama waktu yang dibutuhkan algoritma.

2. Bubble Sort (Sortingan Paling Lambat) Berdasarkan Worst Case

Dalam kasus ini, data terkecil berada pada ujung array. Contoh worst case skenario dapat dilihat pada pengurutan data (4 3 2 1) di bawah ini.

Pass Pertama

(4 3 2 1) menjadi (3 4 2 1)
(3 4 2 1) menjadi (3 2 4 1)
(3 2 4 1) menjadi (3 2 1 4)

Pass Kedua

(3 2 1 4) menjadi (2 3 1 4)
(2 3 1 4) menjadi (2 1 3 4)
(2 1 3 4) menjadi (2 1 3 4)

Pass Ketiga

(2 1 3 4) menjadi (1 2 3 4)
(1 2 3 4) menjadi (1 2 3 4)

(1 2 3 4) menjadi (1 2 3 4)

Pass Keempat
(1 2 3 4) menjadi (1 2 3 4)
(1 2 3 4) menjadi (1 2 3 4)
(1 2 3 4) menjadi (1 2 3 4)

Dari langkah pengurutan di atas, terlihat bahwa setiap kali melakukan satu pass, data terkecil akan bergeser ke arah awal sebanyak satu langkah. Dengan kata lain, untuk menggeser data terkecil dari urutan keempat menuju urutan pertama, dibutuhkan pass sebanyak tiga kali, ditambah satu kali pass untuk verifikasi.

Sehingga notasi Big O yang didapat adalah $O(n^2)$. $O(n^2)$ berarti usaha yang dibutuhkan lebih besar dua kali lipat daripada O(n). Semakin besar atau semakin banyak total input, semakin lebih besar lagi usaha yang dibutuhkan hingga dua kali lipat. Sebagai contoh jika total inputnya 2 maka total operasinya nanti 4, jika total inputnya 6 maka total operasinya 36, dan seterusnya. Berikut adalah tabel algoritma berdasarkan metode pengurutan.

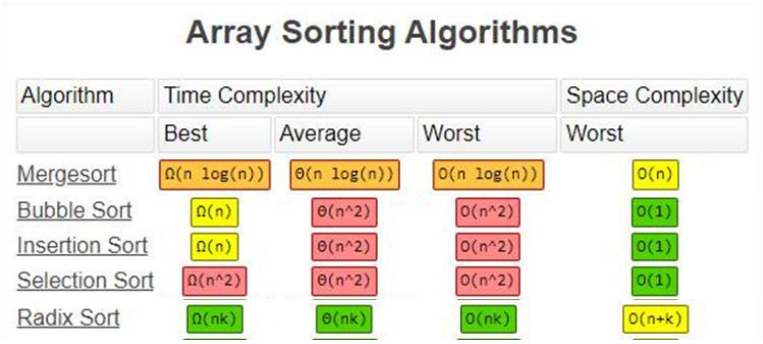
Tabel 2.1 Algoritma Berdasarkan Metode Pengurutan

Algoritma	Worst Case	Kelebihan	Kekurangan
Insertion Sort	$O(n^2)$	Metode pengurutan efisien	Tidak cocok digunakan untuk data berjumlah besar
Merge Sort	O(n log(n))	Metode pengurutan efisien	Tidak cocok digunakan untuk data berjumlah besar
Selection Sort	$O(n^2)$	Metode pengurutan efisien	Sulit dalam membagi masalah
Radix Sort	O(nk)	Metode pengurutan efisien	Sulit dalam membagi masalah
Bubble Sort	$O(n^2)$	Metode pengurutan simpel	Metode pengurutan paling tidak efisien

Menurut tabel algoritma pengurutan di atas menunjukkan perbedaan yang terjadi dalam proses pengurutan. Pada algoritma insertion sort, merge sort, selection sort, dan radix sort memiliki pengurutan yang efisien sedangkan pada algoritma bubble sort memiliki

pengurutan yang paling tidak efisien. Proses pengurutan (sort) merupakan salah satu indikator yang digunakan dalam menentukan proses sorting paling cepat pada algoritma. Hal tersebut membuktikan bahwa algoritma bubble sort tergolong dalam algoritma dengan sort paling lambat berdasarkan worst case.

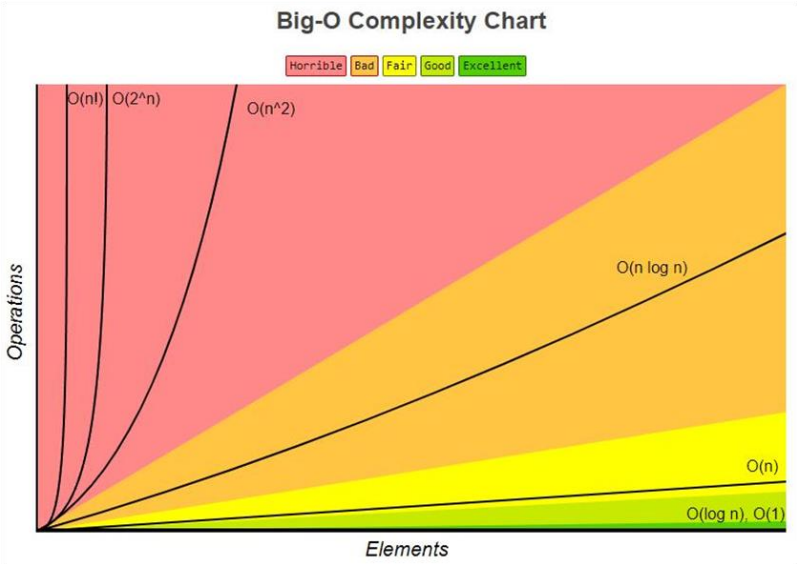
3. Kesimpulan



Array Sorting Algorithms

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	Worst
Mergesort	Ω(n log(n))	Θ(n log(n))	O(n log(n))	O(n)
Bubble Sort	Ω(n)	Θ(n^2)	O(n^2)	O(1)
Insertion Sort	Ω(n)	Θ(n^2)	O(n^2)	O(1)
Selection Sort	Ω(n^2)	Θ(n^2)	O(n^2)	O(1)
Radix Sort	Ω(nk)	Θ(nk)	O(nk)	O(n+k)

Gambar 3. 1 Array Sorting Algorithms



Gambar 3. 2 Big-O Complexity Chart

Berdasarkan gambar diatas tentang algoritma pengurutan array, kompleksitas worst case pada algoritma bubble sort, insertion sort, selection sort adalah $O(n^2)$ yang menempati kategori *horrible* pada Big O complexity sesuai pada gambar di atas yang ditandai dengan warna merah. Kompleksitas worst case pada algoritma merge sort adalah O(n log(n)) yang memasuki kategori *bad* pada Big O complexity sesuai pada gambar yang ditandai dengan warna orange. Dari kategori warna tersebut dapat dilihat perbandingan proses kecepatan eksekusi worst case mulai dari yang paling cepat ke paling lambat. Warna merah (*Horrible*) menunjukkan proses kecepatan eksekusi paling lambat, sedangkan warna orange (*Bad*) menunjukkan proses kecepatan eksekusi lebih cepat dibandingkan dengan warna merah.

Radix sort tidak dianggap sebagai yang tercepat karena merupakan algoritma non-komparatif, sehingga berbeda dengan algoritma sorting komparatif seperti keempat lainnya. Jika kita mengambil angka digit yang sangat besar maka dapat bekerja dalam waktu linier namun membutuhkan ruang yang besar. Hal ini membuat ruang sorting radix tidak efisien, inilah alasan mengapa jenis ini tidak digunakan di *software libraries*.

Tabel 3.1 Kelebihan dan Kekurangan Algoritma Pengurutan

Algoritma	Worst Case	Kelebihan	Kekurangan
Insertion Sort	$O(n^2)$	Proses sorting lebih rapi, sederhana dan mudah dipahami.	Tidak cocok digunakan untuk data berjumlah besar.
Merge Sort	O(n log(n))	Setiap list selalu dibagi-bagi menjadi list yang lebih kecil sehingga tidak perlu banyak perbandingan.	Tidak cocok digunakan untuk data berjumlah besar karena butuh memori yang sangat banyak.
Selection Sort	$O(n^2)$	Operasi pertukaran hanya dilakukan sekali sehingga kompleksitas relatif lebih kecil.	Sulit dalam membagi masalah.
Radix Sort	O(nk)	Algoritma radix sort sangat efektif untuk jumlah data yang sangat besar.	Tidak efisien karena membutuhkan ruang yang besar
Bubble Sort	$O(n^2)$	Metode pengurutan sangat simpel sehingga mudah dipahami	Metode pengurutan yang paling tidak efisien.

Menurut tabel kelebihan dan kekurangan di atas menunjukkan perbedaan yang terjadi pada setiap algoritma. Pada algoritma merge sort proses yang terjadi, yaitu setiap list selalu dibagi-bagi menjadi list yang lebih kecil kemudian digabungkan lagi sehingga tidak memerlukan banyak perbandingan. Hal tersebut membuktikan bahwa algoritma merge sort merupakan algoritma paling cepat berdasarkan kondisi worst case. Pada algoritma insertion sort, selection sort, dan bubble memiliki kompleksitas worst case yang sama, yaitu $O(n^2)$ namun ketiganya tidak tergolong dalam pengurutan paling lambat berdasarkan kondisi worst case. Akan tetapi, pengurutan paling lambat yang terjadi dari ketiga algoritma tersebut adalah algoritma bubble sort karena metode pengurutan yang paling tidak efisien. Hal inilah yan membuktikan bahwa algoritma bubble sort tergolong dalam algoritma dengan sort paling lambat berdasarkan worst case.