CAK2BAB2 ANALISIS KOMPLEKSITAS ALGORITMA

TUGAS BESAR

SELECTION SORT VS HEAP SORT

KELAS IF-47-11

Dosen: KEMAS MUSLIM LHAKSMANA

JORDAN ADZANI (1301213307)

PROGRAM STUDI INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

BANDUNG

2024

1. Deskripsi Studi Kasus Permasalahan

Sorting merupakan salah satu proses fundamental dalam pengolahan data yang digunakan untuk mengatur elemen dalam urutan tertentu, seperti menaik atau menurun. Pemilihan algoritma sorting yang tepat menjadi penting karena efisiensi proses sorting dapat berdampak signifikan pada kinerja aplikasi, terutama saat menangani dataset berukuran besar. Dalam kasus ini, Selection Sort dan Heap Sort dipilih untuk dianalisis karena keduanya menawarkan pendekatan yang berbeda dalam menyelesaikan permasalahan sorting. Selection Sort adalah algoritma sederhana yang mudah dipahami, tetapi memiliki kompleksitas waktu yang tinggi, sedangkan Heap Sort lebih kompleks namun memiliki efisiensi yang lebih baik pada data berskala besar. Studi ini bertujuan untuk mengevaluasi performa kedua algoritma dalam berbagai skenario ukuran data dan memberikan rekomendasi algoritma yang paling sesuai untuk kebutuhan tertentu.

2. Deskripsi Dua Algoritma yang Dipilih

2.1. Selection Sort

Selection Sort adalah algoritma pengurutan sederhana yang bekerja dengan membagi array menjadi dua bagian: bagian yang sudah terurut dan bagian yang belum terurut. Pada setiap iterasi, algoritma ini mencari elemen terkecil dari bagian yang belum terurut, kemudian menukarnya dengan elemen pertama dari bagian tersebut. Proses ini diulang hingga seluruh elemen dalam array berada di bagian yang terurut.

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

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

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

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

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

$$T(n) = n^2 - n - \frac{n^2 - n}{2}$$

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

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

Cara Kerja:

a. Mulai dari indeks pertama array.
b. Cari elemen terkecil dari bagian array yang belum terurut.
c. Tukar elemen terkecil tersebut dengan elemen di indeks pertama dari bagian yang belum terurut.
d. Pindahkan batas bagian yang sudah terurut satu langkah ke kanan.
e. Ulangi langkah 2 hingga array sepenuhnya terurut.

Kompleksitas Waktu:

a. Best Case: $O(n^2)$
b. Average Case: : $O(n^2)$
c. Worst Case: : $O(n^2)$

2.2. Heap Sort

Heap Sort adalah algoritma pengurutan berbasis struktur data heap. Algoritma ini menggunakan konsep heap (binary heap) untuk memilih elemen terbesar (max heap) atau terkecil (min heap) dan secara berulang menempatkannya di posisi akhir array yang belum terurut. Heap Sort efisien dalam hal waktu karena kompleksitasnya $O(nlogn)$ pada semua kasus.

Penjelasan Kompleksitas Waktu Heap Sort :

a. Penghapusan pertama sebuah node membutuhkan waktu $log(n)$.
b. Penghapusan kedua membutuhkan waktu $log(n-1)$.
c. Penghapusan ketiga membutuhkan waktu $log(n-2)$.
d. Dan seterusnya hingga node terakhir, yang akan membutuhkan waktu $log(1)$.

Rumus Perhitungan:

$$T(n) = log(n) + log(n-1) + log(n-2) + \cdots + log(1)$$

$$T(n) = log(n \cdot (n-1) \cdot (n-2) \cdot \cdots \cdot 1)$$

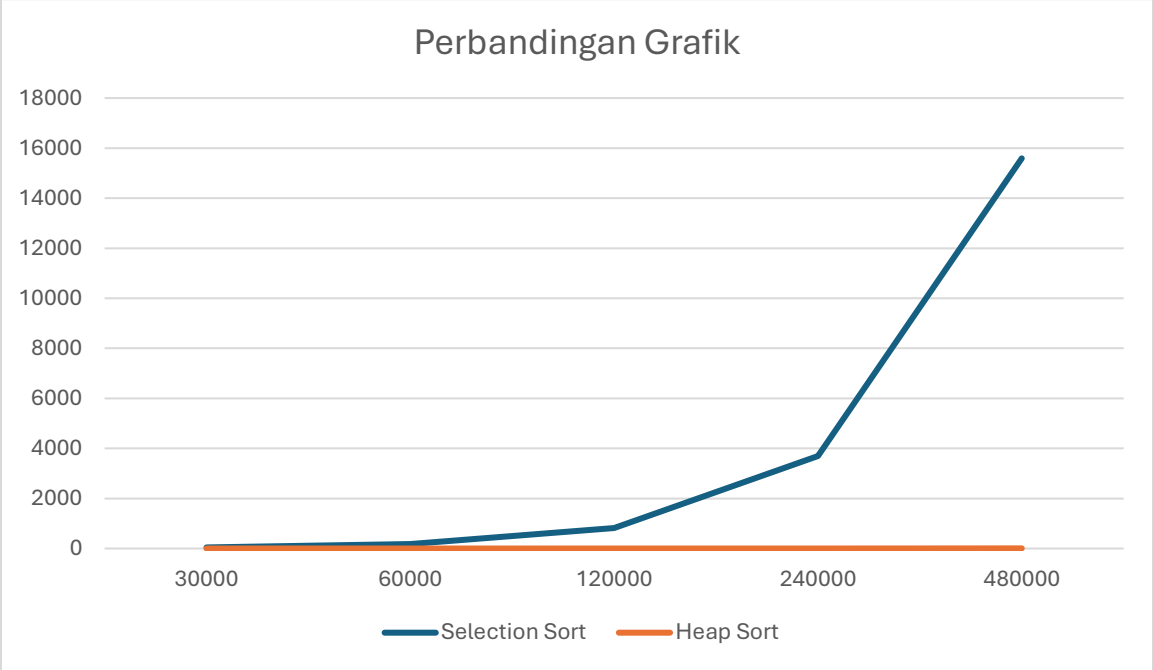
$$T(n) = log(n!)$$

$$T(n) = n \cdot log(n) - n + O(log(n))$$

$$T(n) = O(nlog(n))$$

Dari persamaan di atas didapatkan kompleksitas waktu heap sort, yaitu $O(nlog(n))$. Kompleksitas ini berlaku baik pada kondisi **best-case**, **average-case**, maupun **worst-case**.

3. Grafik Perbandingan Running Time




```
Array Size: 30000
Selection Sort Time: 44.352032 seconds
Heap Sort Time: 0.241776 seconds
-----------------------------
Array Size: 60000
Selection Sort Time: 178.351520 seconds
Heap Sort Time: 0.563709 seconds
-----------------------------
Array Size: 120000
Selection Sort Time: 820.087818 seconds
Heap Sort Time: 1.190917 seconds
-----------------------------
Array Size: 240000
Selection Sort Time: 3690.928538 seconds
Heap Sort Time: 2.613347 seconds
-----------------------------
Array Size: 480000
Selection Sort Time: 15591.345748 seconds
Heap Sort Time: 5.670492 seconds
-----------------------------
```

4. Analisis Perbandingan Kedua Algoritma

Dilihat dari kompleksitas waktu, selection sort memiliki kompleksitas waktu O(n^2), sedangkan heap sort memiliki kompleksitas waktu O(n log n).

Dilihat dari efisiensi, heap sort lebih efisien untuk jumlah data yang besar karena memiliki kompleksitas waktu yang lebih baik daripada selection sort.

5. Referensi

https://www.techiedelight.com/selection-sort-iterative-sive/
https://www.geeksforgeeks.org/heap-sort/
https://www.geeksforgeeks.org/measure-execution-time-function-cpp/
https://iq.opengenus.org/time-complexity-of-heap-sort/