ANALISIS KINERJA INSERTION SORT DAN SELECTION SORT DALAM PEMROGRAMAN PYTHON

Vivielda Farmawaty Tambunan[1], Sabrina Akva[2], Sherly Davina[3], Zulfahmi Indra[4]
vivieldafarmawaty@gmail.com[1], sabrinaakva2005@gmail.com[2], sherlydavinaa@gmail.com[3], zulfahmi.indra@unimed.ac.id[4]
Universitas Negeri Medan

ABSTRAK

Algoritma pengurutan berperan penting dalam pengolahan data dan efisiensi program computer. Penelitian ini menganalisis kinerja algoritma Insertion Sort dan Selection Sort dalam pemrograman Python, dengan fokus pada waktu eksekusi dan penggunaan memori. Eksperimen dilakukan pada dataset acak berukuran 100, 1000, 10000, 15000 elemen. Insertion Sort menunjukkan efisiensi waktu lebih baik pada dataset kecil hingga sedang, sedangkan Selection Sort unggul pada dataset besar dengan waktu eksekusi yang lebih stabil. Kedua algoritma memiliki konsumsi memori yang hampir sama karena sifatnya yang in-place. Hasil penelitian ini memberikan wawasan bagi pengembangan perangkat lunak dalam memilih algoritma sorting yang sesuai dengan karakteksistik data. Pemilihan algoritma yang tepat dapat meningkatkan peforma aplikasi, terutama dalam konteks pengolahan data yang kompleks.

Kata Kunci: Algoritma Pengurutan, Insertion Sort, Selection Sort, Kinerja, Python.

ABSTRACT

Sorting algorithms play a crucial role in data processing and program efficiency. This study analyzes the performance of Insertion Sort and Selection Sort algorithms in Python programming, focusing on execution time and memory usage. Experiments were conducted on random datasets with sizes of 100, 1000, 10000, and 15000 elements. Insertion Sort demonstrated better time efficiency for small to medium datasets, while Selection Sort excelled in large datasets with more stable execution times. Both algorithms exhibited nearly identical memory consumption due to thei in-place nature. This study provides insights for software developers in Selection appropriate sorting algorithms based on data characteristic. Choosing the right algorithm can significantly enhance application performance, particularly in complex data processing scenarios.

Keywords: *Sorting Algorithm, Insertion Sort, Selection Sort, Performance, Python.*

PENDAHULUAN

Algoritma pengurutan (sorting) meletakkan elemen data kedalam kumpulan data urutan tertentu, proses pengurutan yang sebelumnya data disusun acak menjadi tersusun teratur menurut aturan tertentu. Pengurutan merupakan hal yang tidak bisa dipisahkan dari dunia komputer. Sekarang ini Google dikenal sebagai mesin pencari terbesar di dunia. Dalam hitungan detik dapat diperoleh informasi yang diinginkan. Adanya kebutuhan terhadap proses pengurutan memunculkan bermacam-macam metode pengurutan yang bertujuan untuk memperoleh metode pengurutan yang optimal.

Algoritma insertion sort, adalah metode pengurutan dengan cara menyisipkan elemen data pada posisi yang tepat. Pencarian posisi yang tepat dilakukan dengan melakukan pencarian berurutan didalam barisan elemen, selama pencarian posisi yang tepat dilakukan pergeseran elemen. Algoritma selection sort sering juga disebut dengan metode maksimum atau minimum. Metode maksimum karena didasarkan pada pemilihan data atau elemen maksimum sebagai dasar pengurutan. Konsepnya dengan memilih elemen maksimum kemudian mempertukarkan elemen maksimum tersebut dengan elemen paling akhir untuk urutan ascending dan elemen pertama untuk descending. Algoritma selection sort disebut juga dengan metode minimum karena didasarkan pada pemilihan elemen minimum sebagai

dasar pengurutan. Konsepnya dengan memilih elemen minimum kemudian mempertukarkan elemen minimum dengan elemen paling akhir untuk urutan ascending dan elemen pertama untuk urutan descending. Proses yang dilakukan oleh algoritma selection sort adalah mengambil nilai terbesar dari susunan data dan menggantikannya dengan data yang paling kanan

Penelitian ini bertujuan untuk menganalisis kinerja kedua algoritma tersebut dalam konteks pemrograman Python, dengan fokus pada dua aspek utama: waktu eksekusi dan penggunaan memori. Melalui eksperimen yang melibatkan dataset dengan ukuran bervariasi (100, 1000, 10000, dan 15000 elemen), penelitian ini akan memberikan wawasan tentang efektivitas dan efisiensi masing-masing algoritma dalam berbagai kondisi. Hasil dari penelitian ini diharapkan dapat memberikan panduan bagi pengembang perangkat lunak dalam memilih algoritma sorting yang tepat berdasarkan karakteristik data yang dihadapi.

METODE PENELITIAN

Dalam Penelitian ini, metode yang digunakan untuk menganalisis kinerja Insertion Sort dan Selection Sort dalam pemrograman Python meliputi tahapan berikut:

Desain Penelitian

Penelitian ini menggunakan metode eksperimen untuk menguji kinerja dua algoritma pengurutan, yaitu Insertion Sort dan Selection Sort, dengan focus pada waktu eksekusi dan penggunaan memori. Penelitian ini juga bertujuan untuk membandingkan peforma kedua algoritma pada dataset dengan ukuran yang bervariasi. Eksperimen dilakukan dengan memuji dataset acak yang terdiri dari 100 angka, 1000 angka, 10000 angka, dan 15000 angka. Berikut adalah implementasi algoritma Insertion Sort dan Selection Sort menggunakan bahasa pemrograman Python:

1. Implementasi Algoritma Insertion Sort

Algoritma Insertion sort bekerja dengan menyisipkan setiap elemen dataset ke dalam posisi yang tepat satu per satu, hingga seluruh dataset terurut. Kode program untuk algoritma ini ditunjukkan pada Gambar 1.

```python
def insertion_sort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i - 1
        while j >= 0 and arr[j] > key:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
    return arr
```

Gambar 1. Kode Program Insertion Sort

2. Implementasi Algoritma Selection Sort

Algoritma Selection Sort bekerja dengan memilih elemen terkecil dari sisa dataset, lalu menukarnya ke posisi yang sesuai. Kode program untuk algoritma ini ditunjukkan pada Gambar 2.

```python
def selection_sort(arr):
    for i in range(len(arr)):
        min_idx = i
        for j in range(i + 1, len(arr)):
            if arr[j] < arr[min_idx]:
                min_idx = j
        arr[i], arr[min_idx] = arr[min_idx], arr[i]
    return arr
```

Gambar 2. Kode Program Selection Sort

Metode Pengumpulan Data

Data yang digunakan dalam penelitian ini adalah dataset angka acak yang terdiri dari empat ukuran: 100, 1000, 10000, 15000 angka. Setiap dataset diuji menggunakan dua algoritma pengurutan yang berbeda. Penelitian ini mengukur dua aspek utama:

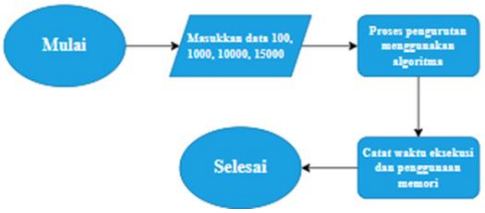
- Waktu Eksekusi: Waktu yang dibutuhkan untuk menyelesaikan proses pengurutan yang mengukur durasi eksekusi algoritma.
- Penggunaan Memori: Penggunaan memori yang dapat memberikan informasi tentang seberapa banyak memori(dalam KB) yang digunakan selama eksekusi algoritma.

Metode Analisis Data

Analisis ini dilakukan dengan membandingkan waktu eksekusi dan penggunaan memori kedua algoritma pada berbagai ukuran dataset. Data yang dikumpulkan kemudian dianalisis secara deskriptif menggunakan table dan grafik untuk menunjukkan perbandingan peforma antara Insertion Sort dan Selection Sort. Analisis ini akan memberikan wawasan mengenai kelebihan dan kekurangan masing-masing algoritma, terutama dalam konteks pernggunaan waktu dan memori pada dataset dengan ukuran yang berbeda.

Langkah-langkah Pengujian

Langkah-langkah pengujian dilakukan untuk mengevaluasi peforma algoritma sorting, yaitu Insertion Sort dan Selection Sort. Data yang digunakan memiliki ukuran berbeda, dimulai dari datases kecil hingga besar. Proses pengujian meliputi memasukkan data, menjalankan algoritma pengurutan, dan mencatat hasil pengukuran berupa waktu eksekusi dan penggunaan memori. Langkah-langkah tersebut dirangkum dalam flowchart berikut:



Gambar 3. Langkah Pengujian dalam Flowchart

Flowchart di atas menggambarkan alur pengujian dengan jelas. Dimulai dengan memasukkan dataset, dilanjutkan dengan proses pengurutan menggunakan algoritma yang diuji, dan diakhiri dengan pencatatan hasil pengukuran. Dengan Langkah ini, kita dapat mengevaluasi efisiensi kedua algoritma berdasarkan waktu eksekusi dan konsumsi memori.

HASIL DAN PEMBAHASAN

Hasil pengujian kinerja Insertion Sort dan Selection Sort ditampilkan pada Tabel 1 dan Tabel 2, yang masing-masing menunjukkan waktu proses dan penggunaan memori untuk dataset berukuran 100, 1000, 10000, 15000 elemen atau data. Hasil ini juga divisualisasikan dalam Gambar 4 dan Gambar 5.

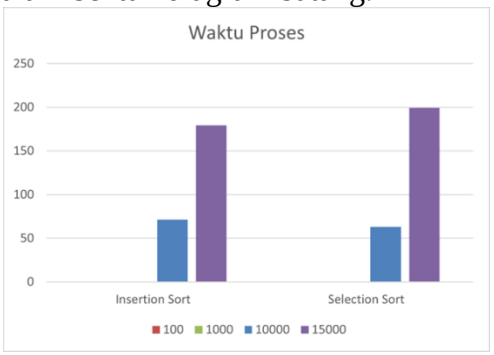
Tabel 1. Waktu Proses

Dataset	Insertion Sort	Selection Sort
100	0	0
1000	0,519542	0,599528
10000	71,25298	63,18164
15000	179,2996	199,2051

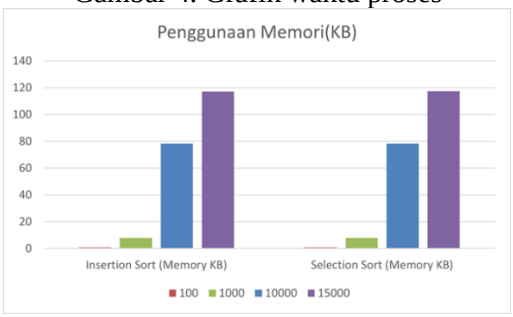
Tabel 2. Penggunaan Memori(KB)

Dataset	Insertion Sort (Memory KB)	Selection Sort (Memory KB)
100	0,86	0,89
1000	7,95	8,07
10000	78,26	78,39
15000	117,32	117,45

Gambar 4 dan 5 menunjukkan perbandingan hasil waktu proses dan penggunaan memori setiap algoritma dalam bentuk diagram batang.



Gambar 4. Grafik waktu proses



Gambar 5. Penggunaan Memori(KB)

1. Waktu Proses

Berdasarkan Tabel 1, waktu proses untuk setiap algoritma meningkat seiring dengan bertambahnya ukuran dataset:

- Pada dataset kecil(100 elemen), waktu eksekusi hampir sama untuk kedua algoritma dan tidak signifikan.
- Pada dataset sedang(1000 elemen), Insertion Sort lebih cepat dibandingkan Selection Sort . Namun, pada dataset besar (10000 dan 15000 elemen), Selection Sort mulai menunjukkan keunggulan waktu proses yang lebih efisien dibandingkan Insertion Sort.
- Pada dataset dengan 15000 elemem, Selection Sort menyelesaikan pengurutan dalam waktu 199,2051 detik, sedangkan Insertion Sort membutuhkan waktu 172,2996 detik.

Grafik pada Gambar 4 menegaskan bahwa perbedaan waktu proses kedua algoritma semakin terlihat pada dataset besar.

2. Penggunaan Memori

Hasil penggunaan memori kedua algoritma ditampilkan pada Tabel 2:

- Pada dataset kecil (100 elemen), penggunaan memori oleh kedua algoritma sangat rendah dan hampir sama dengan selisih yang tidak signifikan.
- Ketika ukuran dataset meningkat, penggunaan memori oleh Selection Sort sedikit lebih tinggi dibandingkan Insertion Sort, tetapi perbedaannya tetap kecil(kurang dari 1 KB).
- Pada dataset terbesar (15000 elemen), Insertion Sort menggunakan memori sebesar 117,32 KB, sedangkan Selection Sort menggunakan 117,45 KB.

Visualisasi Gambar 5 menunjukkan bahwa perbedaan penggunaan memori kedua algoritma tidak signifikan di semua ukuran dataset.

Berdasarkan hasil pengujian yang ditampilkan pada Tabel 1 dan Tabel 2 serta divisualisasikan dalam Gambar 4 dan Gambar 5, dapat dilihat bahwa waktu proses dan penggunaan memorikedua algoritma memiliki karakteristik yang berbeda. Untuk waktu proses, Insertion Sort cenderung lebih cepat dibandingkan Selection Sort pada dataset kecil hingga sedang. Namun, pada dataset besar (10000 dan 15000 elemen), Selection Sort menunjukkan peforma yang lebih stabil dan efisien dalam waktu proses. Penggunaan memori kedua algoritma hampir sama karena keduanta merupakan algoritma in-place yang tidak memerlukan memori tambahan yang signifikan. Secara keseluruhan, Insertion Sort lebih cocok digunakan untuk dataset kecil, sementara Selection Sort lebih ungguk pada dataset yang lebih besar, meskipun perbedannya relative kecil dalam hal penggunaan memori. Visualisasi pada Gambar 4 dan Gambar 5 mendukung analisis ini dengan menunjukkan peningkatan waktu proses dan memori yang linear seiring bertambahnya ukuran dataset.[1]

KESIMPULAN

Insertion Sort dan Selection Sort memiliki karakteristik berbeda dalam waktu eksekusi dan penggunaan memori. Insertion Sort lebih efisien untuk dataset kecil hingga sedang, dengan waktu eksekusi yang lebih cepat, sementara Selection Sort menunjukkan performa yang lebih stabil pada dataset besar, meskipun waktu eksekusinya lebih lambat. Penggunaan memori kedua algoritma hampir sama, menandakan keduanya adalah algoritma in-place yang tidak memerlukan banyak memori tambahan. Oleh karena itu, pemilihan algoritma sorting harus disesuaikan dengan ukuran dan karakteristik data: Insertion Sort lebih disarankan untuk dataset kecil, sedangkan Selection Sort dapat menjadi pilihan yang lebih baik untuk dataset besar. Penelitian ini memberikan panduan berharga bagi pengembang perangkat lunak dalam memilih algoritma yang sesuai dengan kebutuhan aplikasi, serta menekankan pentingnya pemahaman tentang karakteristik masing-masing algoritma dalam konteks pengolahan data. Hasil ini juga membuka peluang untuk penelitian lebih lanjut mengenai algoritma sorting lainnya dan kinerjanya pada berbagai skenario.

DAFTAR PUSTAKA

Basir, R. R. (2020). ANALISIS KOMPLEKSITAS RUANG DAN WAKTU TERHADAP LAJU PERTUMBUHAN ALGORITMA HEAP SORT, INSERTION SORT DAN MERGE DENGAN PEMROGRAMAN JAVA. STRING (Satuan Tulisan Riset dan Inovasi Teknologi), 5(2): 109-118.

Heryanto, Y., Fauziah, F., & Harjanti, T. W. (2023). Analisis Perbandingan Ruang dan Waktu pada Algoritma Sorting Menggunakan Bahasa Pemrograman Python. Kesatria: Jurnal Penerapan Sistem Informasi (Komputer dan Manajemen), 4(2), 342-347.

Iskandar, J., Suhendra, H., & Pamungkas, B. D. (2024). Analisis Strategi Algoritma Sorting Menggunakan Metode Komparatif pada Bahasa Pemrograman Java dengan Python. G-Tech: Jurnal Teknologi Terapan, 8(1): 104-113.

Isyqi, A., Astuti, G. D., Saputro, A. D., Barryananda, M. A., & Suharson. (2024). PERBANDINGAN KINERJA ALGORITMA BUBBLE SORT, INSERTION SORT, DAN SELECTION SORT MENGGUNAKAN DATA BILANGAN NUMERIK PADA PROGRAM PYTHON. Prosiding Seminar Nasional Sains dan Teknologi Seri 02, 1(2): 226-271.

Lasriana & Gunaryati, 1. (2022). SISTEM INFORMASI APOTEK BERBASIS WEB MENGGUNAKAN ALGORITMA SEQUENTIAL SEARCH DAN SELECTION SORT.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika), 7(2): 392-401.
Mahrozi, N. & Faisal, M. (2023). ANALISIS PERBANDINGAN KECEPATAN ALGORITMA SELECTION SORT DAN BUBBLE SORT. Scientica, 1(2): 89-98.
Rahayuningsih, P. A. (2016). Analisis Perbandingan Kompleksitas Algoritma Pengurutan Nilai (Sorting). Jurnal Evolusi, 4 (2): 64-75.
Rahayuningsih, P. A. Analisis Perbandingan Kompleksitas Algoritma Pengurutan Nilai (Sorting). Jurnal Evolusi, 4(2): 64-75.
Ratna, S. (2020). PENGOLAHAN CITRA DIGITAL DAN HISTOGRAM DENGAN PHYTON DAN TEXT EDITOR PHYCHARM. Jurnal Ilmiah Technologia, 11(3): 181-186.
Retnoningsih, E. (2018). Algoritma Pengurutan Data (Sorting) Dengan Metode Insertion Sort dan Selection Sort. Information Management for Educators and Professionals, 3(1), 95–106. 2548-3331.
Rivan, M. Z. A. (2017). Perbandingan Kecepatan Gabungan Algoritma Utama Quick Sort dan Merge Sort dengan Algoritma Tambahan Insertion Sort, Bubble Sort dan Selection Sort. Jurnal Teknik Informatika dan Sistem Informasi, 3(2): 319-331.
Saputri & Yahfizham. (2023). Analisis Study Komperatif Bubble Sort Dan Selection Sort Pada Algoritma Dan Pemograman Berdasarkan Seleksi Pengurutan. Jurnal Arjuna: Publikasi Ilmu Pendidikan, Bahasa dan Matematika, 1(6): 151-161.
Sunandar, E. (2019). Perbandingan Metode Selection Sort dan Insertion Sort dalam Pengurutan Data Menggunakan Bahasa Program Java. Jurnal Pengkajian dan Penerapan Teknik Informatika, 12(2): 172-178.