KESATRIA: Jurnal Penerapan Sistem Informasi (Komputer & Manajemen)
Terakreditasi Nomor 204/E/KPT/2022 | Vol. 4, No. 2, April (2023), pp. 342-347

# Analisis Perbandingan Ruang dan Waktu pada Algoritma Sorting Menggunakan Bahasa Pemrograman Python

*Yayan Heryanto[1], Fauziah[2], Trinugi Wira Harjanti[3]*
*[1,2]Program Studi Teknologi Informasi Fakultas Teknologi Komunikasi dan Informatika Universitas Nasional*
*[3]Sekolah Tinggi Teknologi Informasi NIIT, Indonesia*
*E-mail: 2022.yayan.heryanto@student.unas.ac.id[1], fauziah@civitas.unas.ac.id[2], trinugi@i-tech.ac.id[3]*

***Abstract***

*The role of algorithms in software or programming is very important, so understanding the basic concepts of these algorithms is essential. A lot of programming logic has been made, and in sorting data, insertion sort, quick sort, bubble sort, selection sort, and merge sort algorithms have been used. If you get random data with values of 200, 300, 400, and 500, using the manual method will take a long time, so the five algorithms are used with the Python programming language to sort random data of integer data type. The time required and the size of memory used in each algorithm will be examined during the sorting process. An effective algorithm is one that has a short processing time and uses little memory, so that in this journal, the results for the time efficiency of the Quick Sort algorithm are superior, namely with inputs of 200, 300, 400, and 500, it takes 0.001 seconds, 0.001 seconds, 0.003 seconds, and for memory usage, the Bubble Sort algorithm is superior because it only requires a small amount of memory.*

***Keywords**: Bubble Sort, Selection sort, Merge Sort, Quick Sort, Insertion Sort.*

***Abstrak***

*Peran algoritma dalam perangkat lunak atau pemrograman sangatlah penting, sehingga perlu untuk memahami konsep dasar dari algoritma tersebut, banyak logika pemrograman yang telah dibuat, dalam pengurutan data telah banyak digunakan algoritma insertion sort, quick sort, bubble sort, selection sort, dan merge sort. Jika didapat sebuah data sebanyak 200, 300, 400 dan 500 berbentuk data random maka jika digunakan cara manual akan memakan waktu yang lama dalam proses pengurutan, sehingga digunakan kelima algoritma tersebut dengan menggunakan Bahasa pemrogramana python untuk mengurutkan data random yang bertipe data interger. Dalam proses pengurutan akan di teliti waktu yang diperlukan dan ukuran memory yang dipakai dalam setiap algortima. Algortima yang efektif adalah algoritma yang memiliki tingkat efisienesi waktu proses yang singkat dan penggunaan memory yang sedikit, sehingga didalam jurnal ini telah didapati hasil untuk efisiensi waktu algoritma Quick Sort lebih unggul yaitu dengan inputan 200, 300, 400 dan 500 diperlukan waktu proses 0,001 detik, 0,001 detik, 0,001 detik, 0,003 detik, dan untuk penggunaan memory, algoritma Bubble Sort lebih unggul karena hanya memerlukan ukuran memory yang sedikit dibandingkan dengan yang lainnya yaitu 3.83kB untuk inputan 200, 4.83 untuk inputan 300, 5,83 kB untuk inputan 400, dan 6,83 kB untuk inputan 500.*

***Kata kunci**: Bubble Sort, Selection sort, Merge Sort, Quick Sort, Insertion Sort.*

## 1. Pendahuluan

Teknologi yang berkembang telah membuat data menjadi besar dan berkembang. Data yang besar dan bervariasi akan membuat pengolahan data menjadi lebih susah dan kompleks, dalam tahapan pemrosesan data tentunya diperlukan proses pengurutan agar data lebih mudah di baca dan di proses nantinya. Terdapat banyak algoritma yang bisa

ISSN: 2720-992X

digunakan untuk mengurutkan data, pada algortima Quick sort dan Merge sort adalah algortima yang unggul dalam mengurutkan data dengan jumlah yang besar. Pada insertion sort, bubble sort dan selection sort memilki keunggulan dalam mengurutkan data dengan jumlah sedikit[11]. Didalam penelitian[2], dibandingkan algoritma Insertion sort dengan Merge Sort, hasil yang diperoleh adalah insertion sort merupakan algortima yang lebih cepat dari merge sort dengan jumlah data kurang dari 100, akan tetapi dengan jumlah data lebih dari 100, merge sort lebih cepat.

Di Penelitian[3], perbandingan dilakukan pada lagortima Insertion sort dengan Merge Sort, akan tetapi jumlah datanya berbeda dengan penelitian[2], dan hasil yang diperolah adalah Merge sort lebih cepat dengan data inputan yang lebih besar.

Pada Penelitian[4], algoritma Quick Sort dibandingkan dengan Insertion sort, dari hasil penelitina tersebut insertion sort merupakan lagoritma dengan waktu proses lebih cepat jika dengan data inputan kurang dari 100, jika inputan lebih dari 100 maka quick sort lebih unggul.

Pada Buble sort, optimisasi juga sudah dilakukan dan menghasilkan optimisasi waktu proses yang lebih baik[5]. Di penelitian yang lain algoritma Bubble Sort juga pernah di bandingkan dengan algoritma Selection Sort dan menghasilkan Selection sort lebih efisien[6].

Pada hal lain, algoritma Quick Sort banyak di gunakan dalam system pemrosesan data, pencarian informasi, Bisnis keuangan dan Enkripsi[7]. Gnome sort, Selection and Bubble sort juga di analisis, hasil menunjukan untuk data yang sudah terurut algortima Gnome lebih cepat, sementara untuk data yang belum terurut algortima selection sort lebih cepat[8].

Penelitian-penelitian tersebut memilki tujuan untuk mendapatkan algortima dengan waktu prosess yang lebih cepat ketika mengurutkan data. Pada penelitian ini akan menggunakan lima algoritma yaitu Buble Sort, Insertion Sort, Merge Sort, Quick Sort and Selection Sort dengan inputan data dalam jumlah yang berbeda, yaitu 200, 300, 400 dan 500 dengan tipe data numeric random, dari kelima algoritma tersebut manakah yang memilki waktu proses lebih cepat dan penggunaan memeori yang lebih sedikit.

## 2. Metodologi Penelitian

Dalam penelitian ini menggunakan Bahasa pemrograman python[5] untuk menjalankan algortima Buble Sort, Insertion Sort, Merge Sort, Quick Sort and Selection Sort.

### 2.1. Python

Python adalah bahasa pemrograman yang interaktif dan berorientasi objek. mampu menyediakan struktur data tingkat tinggi[9].

### 2.2. Bubble Sort

Buble sort memiliki cara kerja seperti gelembung, yang ringan akan keatas dan yang berat akan tetap dibawah. Elemen paling kiri akan di bandingkan dengan elemen kanan. Apabila lebih besar dari data sebelah kanan maka akan ditukar. Hal ini akan di ulang terus menerus sampai datanya habis. Kelebihan dari Algoritma Bubble sort yaitu lebih sederhana, sementara kelemehaannya yaitu tidak efisien. Kompleksitas waktu Average Case dan Worst Case yaitu O(n2)[1][10]. Pada gambar 1 ditunjukan kode program algoritma Buble source dalam bahasa pemrograman python.

```
# membandingkan dua elemen yang berdekatan
# Ubah > menjadi < untuk mengurutkan dalam urutan menurun
if array[u] > array[u + 1]:
```

**Gambar 1.** Kode Program Bubble Sort

### 2.3. Insertion Sort

Pada algortima Insertion sort berkerja dengan mencari posisi elemen seharusnya dalam data. Setiap elemen di bandingkan dan mencari nilai yang paling kecil, sampai tidak ada lagi elemen untuk dibandingkan. Insertion sort memliki kompleksitas waktu average case dan worst case yaitu O(n2)[11][10]. Pada gambar 2 ditunjukan kode program algoritma insertion sort dalam bahasa pemrograman python

```
    # membandingkan key dengan setiap elemen di sebelah kirinya sampai
ditemukan yang lebih kecil darinya
    # Untuk urutan menurun, ubah key<array[b] menjadi key>array[b].
    while t >= 0 and key < array[t]:
      array[t + 1] = array[t]
      t = t - 1
```

**Gambar 2.** Kode Program Insertion Sort.

### 2.4. Selection Sort

Menurut Abidin[10], metode algoritma selection sort adalah peningkatan dari metode algoritma bubble sort yaitu dengan cara mengurangi jumlah perbandingan. Algortima selection sort akan memilih elemen maksimum atau minimum dan menempatkan elemen tersebut pada awal atau akhir array. Pada gambar 3 ditunujukan kode program insertion sort dalam bahasa pemrograman python.

```
        # Untuk mengurutkan secara menurun, ubah > menjadi < di baris ini
        # Pilih elemen terkecil di setiap loop
        if array[c] < array[min_idx]:
          min_idx = c
```

**Gambar 3.** Kode Program Insertion Sort.

### 2.5. Merge Sort

Merge sort merupakan metode pengurutan dengan pola divide and conquer[11]. Caranya adalah membagi sebuah kelompok data besar menjadi beberapa kelompok kecil yang terdiri dari dua nilai untuk dibandingkan dan nantinya di gabungkan lagi secara keseluruhan. Pada gambar 4 ditunjukan kode program untuk merge sort dalam bahasa pemrograman python.

```
# elemen Z dan M dan letakkan di posisi yang benar di A[p..r]
while i < len(Z) and j < len(M):
  if Z[i] < M[j]:
    array[k] = Z[i]
    i += 1
  else:
    array[k] = M[j]
    j += 1
  k += 1
# Saat kita kehabisan elemen di Z atau M,
# Ambil elemen yang tersisa dan masukkan ke A[p..r]
while i < len(Z):
  array[k] = Z[i]
  i += 1
  k += 1
while j < len(M):
  array[k] = M[j]
  j += 1
  k += 1
```

**Gambar 4.** Kode Program Merge Sort.

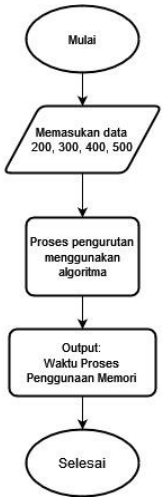
### 2.6. Quick Sort

Quick Sort merupakan algoritma sorting data yang bergantung pada elemen pivot yang cara kerja metodenya dengan mereduksi tahap demi tahap sehingga menjadi 2 bagian yang lebih kecil. Atau yang biasa disebut Divide and Conquer. Elemen angka atau n yang besar akan dipartisi menjadi dua sub-elemen yang mana salah satunya berisi elemen yang lebih kecil dari pivot. Kemudian quick sort akan memanggil dirinya sendiri secara rekursif untuk mengurutkan elemen angka[9]–[12]. Pada gambar 5 ditunujukan kode program quick sort dalam bahasa pemrograman python.

```
# bandingkan setiap elemen dengan pivot
for e in range(low, high):
  if array[e] <= pivot:
    # Jika elemen kurang dari pivot ditemukan
    # tukar dengan elemen yang lebih besar yang ditunjukkan oleh i
    i = i + 1

    # menukar elemen di i dengan elemen di j
    (array[i], array[e]) = (array[e], array[i])

# menukar elemen pivot dengan elemen yang lebih besar yang ditentukan oleh i
(array[i + 1], array[high]) = (array[high], array[i + 1])

# Mengembalikan tempat partisi selesai
return i + 1
```

**Gambar 5.** Kode Program Quick Sort.

Pada tabel 1 ditunjukan proses sorting dengan inputan yang berbeda beda, yaitu 200, 300, 400 dan 500, lalu di catat hasil untuk waktu proses pengurutan, dan pada table 2 di catat jumlah memori yang digunakan dari hasil inputan 200, 300, 400 dan 500.




**Gambar 6.** Flowchart Metodologi.

Pada gambar 6 merupakan flowchart metodologi yang digunakan dalam penelitian ini, data inputan merupakan data random dengan jumlah inputan 200, 300, 400 dan 500, data data tersebut di olah kedalam lima algortima tersebut dan di catat untuk hasil waktu proses dan jumlah memori yang digunakan.

## 3. Hasil Dan Pembahasan

Dari hasil penelitian dihasilkan data seperti pada table berikut ini. Untuk tingkat efiseinsi waktu proses bisa dilihat pada Tabel 1.

**Tabel 1.** Waktu Proses.

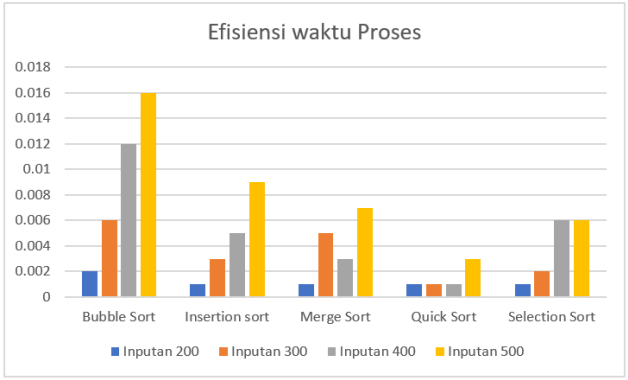
| Algortima | Inputan 200 | Inputan 300 | Inputan 400 | Inputan 500 |
|---|---|---|---|---|
| Bubble Sort | 0.002 | 0.006 | 0.012 | 0.016 |
| Inserton sort | 0.001 | 0.003 | 0.005 | 0.009 |
| Merge Sort | 0.001 | 0.005 | 0.003 | 0.007 |
| Quick Sort | 0.001 | 0.001 | 0.001 | 0.003 |
| Selection Sort | 0.001 | 0.002 | 0.006 | 0.006 |

Untuk hasil penggunaan memori bisa dilihat pada table 2

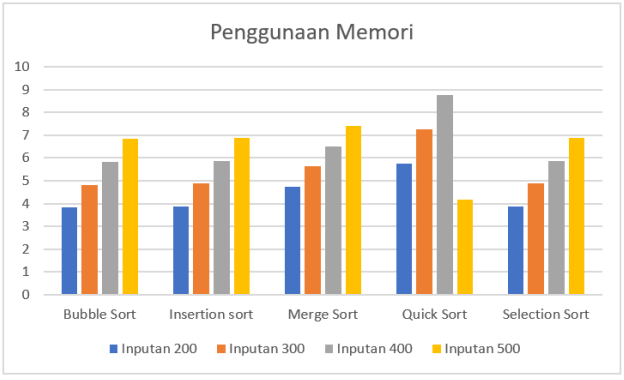
**Tabel 2.** Penggunaan Memori (kB)

| Algortima | Inputan 200 | Inputan 300 | Inputan 400 | Inputan 500 |
|---|---|---|---|---|
| Bubble Sort | 3.83 | 4.83 | 5.83 | 6.83 |
| Inserton sort | 3.87 | 4.87 | 5.87 | 6.87 |
| Merge Sort | 4.72 | 5.63 | 6.52 | 7.42 |
| Quick Sort | 5.77 | 7.27 | 8.77 | 4.19 |
| Selection Sort | 3.87 | 4.87 | 5.87 | 6.87 |

Gambar 7dan 8 menampilkan perbandingan hasil waktu proses untuk setiap algoritma dalam bentuk grafik.




**Gambar 7.** Efisiensi Waktu Proses.




**Gambar 8.** Penggunaan Memori (kB)

## 4. Kesimpulan

Dengan inputan yang berbeda beda memberikan pengaruh terhadap waktu proses dan penggunaan memori, namun dari pengujian tersebut algoritma dengan efisiensi waktu proses lebih singkat belum tentu menjadi algoritma yang memiliki jumlah memori lebih

sedikit. Pada pengujian didapat algoritma dengan waktu proses lebih cepat yaitu quick sort dimana dengan inputan 200 (0.001 detik), inputan 300 (0.001 detik), inputan 400 (0.001 detik), inputan 500 (0.003 detik). Sementara untuk algoritma dengan penggunaan memori lebih sedikit yaitu algoritma Buble Sort, dimana inputan 200 (3.83 kB), 300 (4.83 kB), 400 (5.83 kB), 500 (6.83 kB).

## Daftar Pustaka

[1] Muhammad Ezar Al Rivan, "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 Volume 3 Nomor 2 Agustus 2017, E-Issn : 2443-2229.

[2] Arief Hendra Saptadi And D. W. Sari, "Analisis Algoritma Insertion Sort, Merge Sort Dan Implementasinya Dalam Bahasa Pemrograman C++," Vol. 8, Pp. 1–8, 2012.

[3] R. Hibbler, "Quick Sort," Dept. Comput. Sci. Florida Inst. Technol. Florida, Usa., 2008.

[4] P. Sareen, "International Journal Of Advanced Research In Computer Science And Software Engineering Comparison Of Sorting Algorithms ( On The Basis Of Average Case )," Int. J. Adv. Res. Comput. Sci. Softw. Eng., Vol. 3, No. 3, Pp. 522–532, 2013.

[5] W. Min, "Analysis On Bubble Sort Algorithm Optimization", 2010 International Forum On Information Technology And Applications, Ieee, 2010.

[6] R. Edjlal, A. Edjlal, T. Moradi, "A Sort Implementation Comparing With Bubble Sort And Selection Sort", Pp 380-381, Ieee, 2011.

[7] W. Xiang, "Analysis Of The Time Complexity Of Quick Sort Algorithm" Pp 408 - 410, Ieee, 2011.

[8] J. Hammad, A Comparative Study Between Various Sorting Algorithms", Vol.15 No.3, Pp 11-16, March 2015. [5] Dwipo Setyantoro, Rika Astuti Hasibuan, "Analisis Dan Perbandingan Kompleksitas Algoritma Exchange Sort Dan Insertion Sort Untuk Pengurutan Data Menggunakan Python", Tekinfo Vol. 21 No. 1, April 2020.

[9] M. F. Sanner, "Python: A Programming Language For Software Integration And Development", The Scripps Research Institute 10550 North Torrey Pines Road, La Jolla, Ca-92037.

[10] Abidin, Taufik Fuadi. 2008. Struktur Data. Informatika. Universitas Syah Kuala Banda Aceh.

[11] Karve, S. Insertion Sort Example. Http://Www.Dreamincode.Net/Code/Snippet279.Htm.

[12] Rahayuningsih, P. 2016. Analisis Perbandingan Kompleksitas Algoritma Pengurutan Nilai (Sorting). Jurnal Evolusi. No.4, Vol.1, 64–75.

[13] Verma, M., Dan Chowdhary, K. R. 2018. Analysis Of Energy Consumption Of Sorting Algorithms On Smartphones. Proceedings Of 3rd International Conference On Internet Of Things And Connected Technologies (Iciotct). No.3, Vol.89, 472-475.

[14] Chauhan, Y., Dan Duggal, A. 2020. Different Sorting Algorithms Comparison Based Upon The Time Complexity. International Journal Of Research And Analytical Reviews. No.3, Vol.7, 114-121.

[15] Kumar, S., & Singla, P. 2019. Sorting Using A Combination Of Bubble Sort, Selection Sort & Amp; Counting Sort. International Journal Of Mathematical Sciences And Computing. No.2, Vol.