Al'adzkiya International of Computer Science and Information Technology (AIoCSIT) Journal
Vol. 6, No. 2, November 2025, pp. 37~42
ISSN: 2722-0001, DOI: 10.55311/aiocsit.v6i2.354

Comparative Analysis of Radix Sort, Quick Sort, and Bubble Sort Algorithms in Data Sorting Based on Array Size and Time

Kurnia Wati Fadhilah[1], Nandy Thaher Ulga[2], Raffi Ramadhan Oktaviansyah[3], Farah Sulistia[4], Jeni Amanda[5], Akhmad Jayadi[6]
[1,2,3,4,5,6]Department of InformationTechnology, Politeknik Negeri Lampung

ABSTRACT

An algorithm is a series of logical actions used to solve important problems in contemporary programming and data processing. The purpose of this study is to compare the time efficiency of three sorting algorithms: Bubble Sort, Radix Sort, and Quick Sort. All algorithms are used on small (10-100 elements), medium (1,000-10,000 elements), and large (more than 100,000 elements) arrays, with execution time using Java. The results show that Radix Sort and Quick Sort are generally more efficient and scalable than Bubble Sort, especially for large arrays and random or semi-sorted data. Radix Sort excels on small and medium arrays under various conditions, while Quick Sort excels on large arrays in the average and nearly sorted cases. Although Bubble Sort can be the fastest in the best case for large arrays, its performance drops drastically in the average and nearly sorted cases. In conclusion, the selection of the best sorting algorithm depends heavily on the type of input data, such as its size and the degree of initial sorting.

Keyword : Array size, java, time efficiency, algorithm comparison

Corresponding Author:
Akhmad Jayadi,
Department of Information Technology,
Politeknik Negeri Lampung,
Lampung,
Email: akhmad.jayadi@polinela.ac.id

1. INTRODUCTION

An algorithm is a series of systematic and logical steps [1] used to solve a problem. A programming algorithm, on the other hand, is a set of sequential steps used to solve a computer programming problem. We can say that an algorithm is the first step that must be taken before creating a program in basic programming. Besides being correct, an algorithm must be effective [2]. Therefore, a correct algorithm may not be useful for certain types and sizes of input because it requires a lot of memory space or time to run.

Computers can use algorithms to make fast, repeatable, and precise decisions. The output produced by an algorithm depends on the values and constraints set by its developer. Only algorithms created by programmers can be used by computers. An algorithm can be briefly defined as knowledge that considers how to approach a problem in a certain way and then structures it in a logical language for a specific purpose.

The goal of a sorting algorithm [3] is to reorder the items of a list without reducing their value or content. One of the important processes in programming and data processing is data sorting. As technology advances, various sorting algorithms have been developed, each with different characteristics, efficiency, and complexity. This study tests and compares three popular sorting algorithms: Bubble Sort [4], Radix Sort [5], and Quick Sort [6]. These three algorithms are tested with various array sizes [7], including small, medium, and large, as well as with various input data conditions, such as best case (data already sorted), average case (random data), and nearly sorted (approximately 90% of the data already sorted).

Each algorithm in its application has two unique characteristics that can be used as comparison parameters: the number of processes performed and the amount of memory used to perform the processes. The number of processes is called time complexity, and the amount of memory is called space complexity, respectively. The goal of this study is to determine how well the three algorithms perform under various conditions so that we can decide which algorithm is most suitable for use.

2. RESEARCH METHOD/MATERIAL AND METHOD/LETERATURE REVIEW

A. Radix Sort

The Radix Sort algorithm is one of the most efficient sorting algorithms because it doesn't perform direct comparisons. For integers, this algorithm groups data based on digits that have the same value and significant position. These groups of digits are stored in variables called buckets, which are usually represented as arrays.

B. Quick Sort

The Quick Sort method, also known as Partition Exchange Sort, was introduced by C. A. R. Hoare. To increase efficiency, this method first swaps elements that are far apart in the initial stage, before focusing on the elements in the middle. Quick Sort can be implemented in two ways:

- Recursive
- Non-recursive

To sort elements in ascending order, we can perform the swap N/2N/2N/2 times, namely by swapping the leftmost element with the rightmost element, then gradually moving towards the middle. However, this method can only be applied if the initial order of the data is known (i.e., descending).

3. RESULTS AND DISCUSSION

A. Comparison Table

To determine the performance and scalability of each algorithm, Bubble Sort, Radix Sort, and Quick Sort are compared based on their execution time. Algorithm speed is measured in milliseconds (ms).
This analysis focuses primarily on the best-case scenario, where the data elements are perfectly sorted or the desired element is found at its starting position, requiring few or no swap operations.

Table 1. Best case

Algoritma	Waktu (ms) – Ukuran 100	Waktu (ms) – Ukuran 1000	Waktu (ms) – Ukuran 100000
Radix	0.000017	0.000236	12.437
Quick	0.0253	0.1234	0.1453
Bubble	0.0052	0.0061	0.0376

Random data conditions are often the best representation of variable data scenarios, representing the most common conditions in real-world applications. Therefore, this analysis is important for understanding the practical performance of the three sorting methods.

Table 2. Avarage case

Algoritma	Waktu (ms) – Ukuran 100	Waktu (ms) – Ukuran 1000	Waktu (ms) – Ukuran 100000
Radix	0.000027	0.000197	15.104
Quick	0.0286	0.5321	0.4798
Bubble	0.2775	55.577	226.295.610

This condition indicates that approximately 90 percent of the data elements are in their correct positions. This demonstrates the algorithm's ability to optimize the sorting process with minimal adjustments. This condition is particularly important for applications where data often has a certain degree of structure or order prior to sorting. The obtained results allow for a comprehensive analysis of how the efficiency of each algorithm changes when only a few elements need to be rearranged in small, medium, and large arrays.

Table 3. Hampir terurut 90%

Algoritma	**Waktu (ms) – Ukuran 100**	**Waktu (ms) – Ukuran 1000**	**Waktu (ms) – Ukuran 100000**
Radix	0.000020	0.000149	13.840
Quick	0.0279	0.1938	0.1937
Bubble	0.2012	44.100	226.295.610

B. Algorithm Comparison Results Graph

This graph shows the time comparison results more clearly, with increases and decreases in the graph corresponding to the times shown in the table.

Radix Sort is excellent for small and medium-sized arrays, and its scalability for large arrays remains good, although the measured execution time is longer.

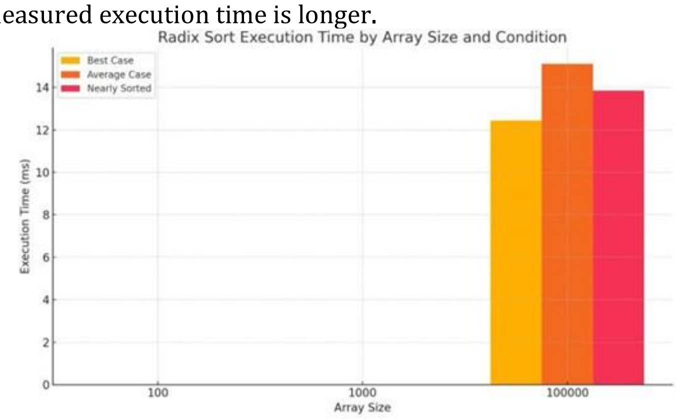

Fig. 1. Radix sort

Quick Sort is very effective and exhibits fast and scalable performance for most scenarios, especially on large arrays in the average and nearly sorted cases. However, its execution time increases with array size, but remains relatively low compared to Bubble Sort.

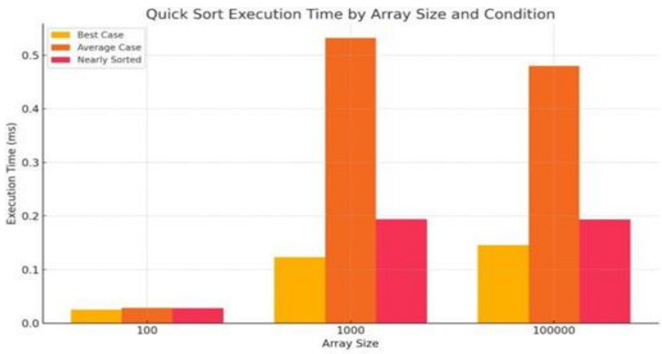

Fig. 2. Quick sort

Bubble sort is not recommended for large data sets or when time efficiency is critical, as its performance drops dramatically in average and near-sorted cases. However, in the best case for large arrays, bubble sort may be the fastest.

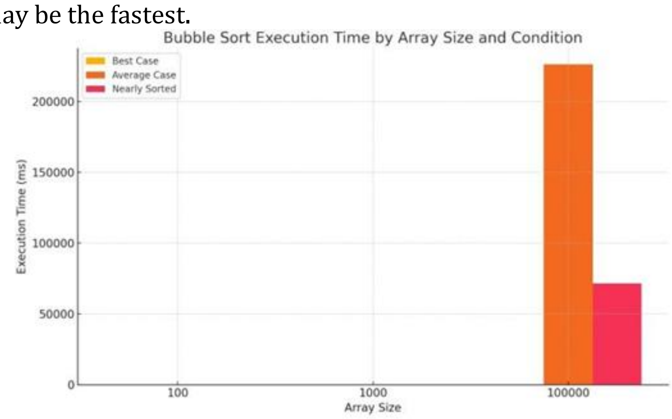

Fig. 3. Bubble sort

4. CONCLUSION

Radix Sort is a highly effective and stable algorithm for a wide range of input types and sizes. This makes it a good choice, especially for sorting numeric data. For most scenarios, Quick Sort is also very effective, demonstrating fast and scalable performance. Conversely, Bubble Sort is not recommended for very small data sizes or when the data is known to be nearly perfectly sorted. In other cases, time efficiency is not important.

1) Best Case
 • Fastest:
 o Small Array: Radix Sort
 o Medium Array: Radix Sort
 o Large Array: Bubble Sort
 • Worst:
 o Small Array: Quick Sort
 o Medium Array: Quick Sort
 o Large Array: Radix Sort

2) Average Case
 • Fastest:
 o Small Array: Radix Sort
 o Medium Array: Radix Sort
 o Large Array: Quick Sort
 • Worst:
 o Small Array: Bubble Sort
 o Medium Array: Bubble Sort
 o Large Array: Bubble Sort

3) Nearly 90% Sorted

• Fastest:
o Small Array: Radix Sort
o Medium Array: Radix Sort
o Large Array: Quick Sort
• Worst:
o Small Array: Bubble Sort
o Medium Array: Bubble Sort
o Large Array: Bubble Sort

REFERENCES

[1] Selviana Anggreani and Yahfizham Yahfizham, "Pengantar dan Pengenalan Konsep Dasar Algoritma Pemrograman," Pendek. J. Pendidik. Berkarakter, vol. 2, no. 1, pp. 282–294, Jan. 2024, doi: 10.51903/pendekar.v2i1.599.

[2] R. Rismayani, N. Sambo Layuk, S. Wahyuni, H. Wali, and N. K. Marselina, "Pencarian Kata Pada Aplikasi Kamus Istilah Komputer dan Informatika Menggunakan Algoritma Brute Force Berbasis Android," Komputika J. Sist. Komput., vol. 10, no. 1, pp. 43–52, Mar. 2021, doi: 10.34010/komputika.v10i1.3644.

[3] Y. A. Sandria, M. R. A. Nurhayoto, L. Ramadhani, R. S. Harefa, and A. Syahputra, "Penerapan Algoritma Selection Sort untuk Melakukan Pengurutan Data dalam Bahasa Pemrograman PHP," Hello World J. Ilmu Komput., vol. 1, no. 4, pp. 190–194, Dec. 2022, doi: 10.56211/helloworld.v1i4.187.

[4] E. Sunandar, "Implementation Of Bubble Sort Algorithm On 2 Fruit Models Of Data Selection Using The Java Program Language," PETIR, vol. 14, no. 2, pp. 159–169, Sep. 2021, doi: 10.33322/petir.v14i2.946.

[5] A. Andersson and S. Nilsson, "Implementing radixsort," ACM J. Exp. Algorithmics, vol. 3, p. 7, Sep. 1998, doi: 10.1145/297096.297136.

[6] R. Rijaya and M. E. Al Rivan, "Perbandingan Penempatan Pivot Pada Quick Sort Berdasarkan Ukuran Pemusatan Data," J. Algoritm., vol. 4, no. 1, pp. 13–20, Oct. 2023, doi: 10.35957/algoritme.v4i1.5735.

[7] R. J. L. Purba, Y. S. M. Gurning, D. P. Purba, and A. K. Nasution, "Implementasi Array dalam Pengolahan Data Nilai Mahasiswa di Perguruan Tinggi," J. Account. Law Commun. Technol., vol. 2, no. 1, pp. 71–80, Dec. 2024, doi: 10.57235/jalakotek.v2i1.4150.

[8] Sari, I.P., Al-Khowarizmi,A.K., Apdilah, D., Manurung, A.A., & Basri, M. (2023). Perancangan Sistem Pengaturan Suhu Ruangan Otomatis Berbasis Hardware Mikrokontroler Berbasis AVR. sudo Jurnal Teknik Informatika 2 (3), 131-142

[9] Wardani., S, & Dewantoro., RW. (2024). Internet of Things: Home Security System based on Raspberry Pi and Telegram Messenger. Indonesian Journal of Applied Technology, Computer and Science 1 (1), 7-13

[10] Sari, I.P., Al-Khowarizmi, A.K., Hariani, P.P., Perdana, A., & Manurung, A.A. (2023). Implementation And Design of Security System On Motorcycle Vehicles Using Raspberry Pi3-Based GPS Tracker And Facedetection. Sinkron: jurnal dan penelitian teknik informatika 8 (3), 2003-2007

[11] Y.Efendi, "Internet of Things (IoT) Light Control System Using Mobile-Based Raspberry Pi", Scientific Journal of Computer Science, Vol. 4, no. 1, April 2018.

[12] Sari, I.P., Basri, M., Ramadhani, F., & Manurung, A.A. (2023). Penerapan Palang Pintu Otomatis Jarak Jauh Berbasis RFID di Perumahan. Blend Sains Jurnal Teknik 2 (1), 16-25

[13] SJ Sokop et.al, "Peripheral Interface Trainer Based on Arduino Uno Microcontroller", E-Journal of Electrical and Computer Engineering vol.5 no.3 (2016).

[14] Sari, I.P., & Batubara, I.H. (2020). Aplikasi Berbasis Teknologi Raspberry Pi Dalam Manajemen Kehadiran Siswa Berbasis Pengenalan Wajah. JMP-DMT 1 (4), 6

[15] M. Saleh and M. Haryanti, "Design of a Home Security System Using Relays", Journal of Electrical Technology, Mercu Buana University, Vol. 8 No. May 2, 2017

[16] Sari, I.P., Batubara, I.H., & Basri, M. (2022). Implementasi Internet of Things Berbasis Website dalam Pemesanan Jasa Rumah Service Teknisi Komputer dan Jaringan Komputer. Blend Sains Jurnal Teknik 1 (2), 157-163

[17] Matondang, M.H.A., Asadel, A., Fauzan, D., & Setiawan, A.R. (2024). Smart Helmet for Motorcycle Safety Internet of Things Based. Tsabit Journal of Computer Science 1 (1), 35-39

[18] Sari, I.P., Novita, A., Al-Khowarizmi, A., Ramadhani, F., & Satria, A. (2024). Pemanfaatan Internet of Things (IoT) pada Bidang Pertanian Menggunakan Arduino UnoR3. Blend Sains Jurnal Teknik 2 (4), 337-343

[19] Husaini, A., & Sari, I.P. (2023). Konfigurasi dan Implementasi RB750Gr3 sebagai RT-RW Net pada Dusun V Suka Damai Desa Sei Meran. sudo Jurnal Teknik Informatika 2 (4), 151-158

[20] Sari, I.P., Apdilah, D., & Guntur, S. (2025). Sistem Smart Class Berbasis Internet of Things (IoT). sudo Jurnal Teknik Informatika 4 (1), 33-39

[21] Indah Purnama Sari. Algoritma dan Pemrograman. Medan: UMSU Press, 2023, pp. 290.

[22] Indah Purnama Sari. Buku Ajar Pemrograman Internet Dasar. Medan: UMSU Press, 2022, pp. 300.

[23] Indah Purnama Sari. Buku Ajar Rekayasa Perangkat Lunak. Medan: UMSU Press, 2021, pp. 228.
[24] Janner Simarmata Arsan Kumala Jaya, Syarifah Fitrah Ramadhani, Niel Ananto, Abdul Karim, Betrisandi, Muhammad Ilham Alhari, Cucut Susanto, Suardinata, Indah Purnama Sari, Edson Yahuda Putra. Komputer dan Masyarakat. Medan: Yayasan Kita Menulis, 2024, pp.162.
[25] Mahdianta Pandia, Indah Purnama Sari, Alexander Wirapraja Fergie Joanda Kaunang, Syarifah Fitrah Ramadhani Stenly Richard Pungus, Sudirman, Suardinata Jimmy Herawan Moedjahedy, Elly Warni, Debby Erce Sondakh. Pengantar Bahasa Pemrograman Python. Medan : Yayasan Kita Menulis, 2024, pp.180
[26] Zelvi Gustiana Arif Dwinanto, Indah Purnama Sari, Janner Simarmata Mahdianta Pandia, Supriadi Syam, Semmy Wellem Taju Fitrah Eka Susilawati, Asmah Akhriana, Rolly Junius Lontaan Fergie Joanda Kaunang. Perkembangan Teknologi Informatika. Medan: Yayasan Kita Menulis, 2024, pp.158
[27] Muharman Lubis Ilham Firman Ashari, Debby Erce Sondakh, Rahmawati Rolly Junius Lontaan, Mustarum Musaruddin Indah Purnama Sari, Muh. Nadzirin Anshari Nur, Hanalde Andre Muh. Rais, Janner Simarmata. Internet of Things (IoT) Dan Multimedia: Integrasi Dan Aplikasi. Medan: Yayasan Kita Menulis, 2024, pp.182