**Maliki Interdisciplinary Journal (MIJ)** eISSN: 3024-8140
Volume 3, Issue Februari, 2025 pp. 289-294 http://urj.uin-malang.ac.id/index.php/mij/index

# Implementasi dan Analisis *Algoritma Binary Search*

**Rizyallatul Nurvita Meindika Putri**
*Program Studi Teknik Informatika, Universitas Islam Negeri Maulana Malik Ibrahim Malang*
*e-mail: rizyallatulnurvita@gmail.com*

***Kata Kunci:***
*Binary Search, Algoritma Pencarian, Kompleksitas Waktu, Implementasi, Studi Kasus*

**Keywords:**
Binary Search; Search Algorithm; Time Complexity; Implementation; Case Study

***A B S T R A K***

*Algoritma Binary Search adalah metode pencarian yang sangat efisien untuk menemukan elemen dalam array yang telah diurutkan. Penelitian ini bertujuan untuk menganalisis konsep dasar, implementasi, serta keunggulan dari algoritma ini dibandingkan dengan metode pencarian linier. Dengan kompleksitas waktu O(log n), Binary Search secara signifikan mengurangi jumlah elemen yang diperiksa pada setiap iterasi melalui pendekatan logaritmik. Implementasi algoritma ini diperlihatkan melalui bahasa pemrograman, dan studi kasus pencarian judul buku dalam perpustakaan digital digunakan sebagai contoh praktis. Hasil analisis menunjukkan bahwa Binary Search menawarkan efisiensi yang tinggi, terutama dalam skenario data besar yang terstruktur dengan baik. Pengetahuan ini akan membuka peluang untuk mengembangkan solusi perangkat lunak yang lebih cepat, efisien, dan handal dalam berbagai bidang aplikasi*

**A B S T R A C T**

The Binary Search algorithm is an extremely efficient search method for finding elements within a sorted array. This research aims to analyze the basic concepts, implementation, and advantages of this algorithm compared to linear search methods. With a time complexity of O(log n), Binary Search significantly reduces the number of elements inspected in each iteration through its logarithmic approach. The implementation of this algorithm is demonstrated through programming languages, and a case study of searching for book titles in a digital library is used as a practical example. The analysis results show that Binary Search offers high efficiency, particularly in scenarios involving large, well-structured data sets. This knowledge will open up opportunities to develop faster, more efficient and more reliable software solutions in various application fields

## Pendahuluan

Algoritma Binary Search adalah salah satu metode pencarian yang sangat efisien untuk menemukan posisi elemen dalam array yang sudah diurutkan. Algoritma ini bekerja dengan cara membagi array menjadi dua bagian, kemudian memeriksa elemen tengah, dan menentukan posisi elemen target berdasarkan perbandingan dengan elemen tengah tersebut. Jika elemen tengah bukan elemen yang dicari, proses ini terus berlanjut pada separuh array yang relevan, baik separuh atas maupun separuh bawah, sehingga mengurangi jumlah elemen yang perlu diperiksa pada setiap iterasi secara logaritmik. Hal ini membuat Binary Search memiliki keunggulan efisiensi dibandingkan dengan metode pencarian linier yang memerlukan waktu O(n). (Aminnur et al., 2023)

### Penerapan dalam Berbagai Bidang

Penerapan algoritma Binary Search dapat ditemukan dalam berbagai bidang, antara lain dalam pengelolaan basis data untuk mempercepat pencarian data, dalam sistem informasi geografis untuk mencari koordinat tertentu, dan dalam perpustakaan digital untuk menemukan judul buku secara cepat. Karena kemampuannya untuk mengurangi

kompleksitas pencarian secara signifikan, pemahaman mendalam tentang algoritma ini sangat penting bagi para pengembang perangkat lunak yang ingin menciptakan sistem yang efisien dan responsif.(Setiawan et al., 2024)

**Peran dalam Algoritma dan Struktur Data Lainnya**

Selain itu, algoritma Binary Search tidak hanya terbatas pada aplikasi tersebut tetapi juga memiliki peran penting dalam berbagai algoritma dan struktur data lainnya. Misalnya, dalam struktur data pohon biner atau AVL tree, prinsip dasar dari Binary Search sering digunakan untuk menjaga keseimbangan dan efisiensi struktur data tersebut. Oleh karena itu, mempelajari dan menguasai algoritma Binary Search tidak hanya memberikan keuntungan praktis dalam menyelesaikan masalah pencarian, tetapi juga memperluas wawasan dan keterampilan dalam bidang ilmu komputer secara umum.

**Pentingnya Pemahaman Mendalam**

Dengan demikian, penting bagi setiap pengembang perangkat lunak dan ilmuwan komputer untuk memahami dan mampu mengimplementasikan algoritma Binary Search dalam berbagai konteks, mengingat efisiensi dan keunggulannya yang sangat signifikan dalam mengelola dan mencari data secara optimal. Pengetahuan ini akan membuka peluang untuk mengembangkan solusi perangkat lunak yang lebih cepat, efisien, dan handal dalam berbagai bidang aplikasi.(Fauzi, 2023)

## Pembahasan

### *Hasil Implementasi Algoritma Binary Search*

Algoritma Binary Search telah diimplementasikan menggunakan bahasa pemrograman Java. Pengujian dilakukan dengan menggunakan data input berupa array yang telah diurutkan, baik dalam bentuk angka maupun teks. Berikut ini adalah hasil pengujian untuk beberapa skenario berbeda: (Nasution & Siddik, 2023)

Tabel 1: Hasil Pengujian Binary Search pada Data Terurut

| Ukuran Array | Posisi Target | Waktu Eksekusi (ms) | Jumlah Iterasi | Hasil |
|---|---|---|---|---|
| 10 | Tengah | 1 | 2 | Elemen ditemukan |
| 10 | Akhir | 1 | 3 | Elemen ditemukan |
| 10 | Tidak Ada | 1 | 4 | Elemen tidak ada |
| 100 | Tengah | 2 | 6 | Elemen ditemukan |
| 100 | Akhir | 2 | 7 | Elemen ditemukan |
| 100 | Tidak Ada | 2 | 7 | Elemen tidak ada |
| 1000 | Tengah | 5 | 10 | Elemen ditemukan |
| 1000 | Akhir | 6 | 11 | Elemen ditemukan |
| 1000 | Tidak Ada | 6 | 11 | Elemen tidak ada |

Berdasarkan hasil pengujian, berikut adalah beberapa analisis dan pembahasan mengenai performa algoritma Binary Search:

**Efisiensi Waktu Eksekusi**

1. **Data Ukuran Kecil:** Pada array yang berukuran kecil (10 elemen), algoritma ini hanya memerlukan waktu yang sangat singkat, yaitu sekitar 1 milidetik (ms), untuk menemukan elemen target.
2. **Data Ukuran Menengah hingga Besar**: Untuk array yang lebih besar, yaitu dengan jumlah elemen antara 100 hingga 1000, waktu eksekusi tetap efisien dan bertambah secara logaritmik seiring dengan peningkatan ukuran array.
3. Hal ini membuktikan bahwa Binary Search memiliki kompleksitas waktu O(log n), di mana jumlah elemen yang diperiksa berkurang setengahnya pada setiap iterasi.(Wibowo & Mico, 2021)

**Jumlah Iterasi**

1. Hasil pengujian menunjukkan bahwa jumlah iterasi yang diperlukan untuk menemukan elemen target berbanding lurus dengan logaritma ukuran data. Contoh: Pada array dengan 1000 elemen, algoritma ini hanya memerlukan 10-11 iterasi untuk menemukan elemen atau memastikan bahwa elemen tersebut tidak ada.
2. Perbandingan ini sesuai dengan prinsip dasar dari Binary Search, di mana array dibagi menjadi dua bagian pada setiap langkah.

**Lokasi Elemen Target**

**Elemen di Tengah Array**: Jika elemen target berada di tengah array, jumlah iterasi yang diperlukan lebih sedikit karena posisi target cepat ditemukan. **Elemen di Akhir atau Tidak Ditemukan**: Jika elemen target berada di akhir array atau tidak ditemukan, jumlah iterasi mencapai maksimum mendekati $\log_2(n)$.

**Perbandingan dengan Linear Search**

Untuk menilai performa Binary Search, algoritma ini dibandingkan dengan Linear Search pada data yang sama. (Ladjin, Litriani, Sahamony, Kusumaningrum, Maulina, Siregar, Hubbansyah, Solikin, Silitonga, Soeyatno, Asyari, Sinaga, 2022)Berikut hasil perbandingan waktu eksekusi:

| Ukuran Array | Binary Search (ms) | Linear Search (ms) |
|---|---|---|
| 10 | 1 | 2 |
| 100 | 2 | 8 |
| 1000 | 6 | 15 |

**Analisis:**

**Efisiensi Waktu Eksekusi**: Algoritma Binary Search menunjukkan waktu eksekusi yang jauh lebih cepat dibandingkan dengan Linear Search, terutama pada dataset yang besar. Pada array dengan 1000 elemen, Binary Search hanya memerlukan 6 milidetik (ms), sedangkan Linear Search membutuhkan waktu 15 ms. Hal ini menegaskan keunggulan

efisiensi Binary Search, khususnya ketika digunakan pada dataset besar yang sudah diurutkan.

**Kelebihan dan Keterbatasan Algoritma Binary Search**

**Kelebihan:** Algoritma ini memiliki waktu eksekusi yang sangat cepat dibandingkan dengan pencarian linier, Dengan kompleksitas waktu O(log n), Binary Search mampu memberikan kinerja optimal pada dataset yang besar.

**Keterbatasan:** Algoritma ini hanya dapat diterapkan pada data yang telah diurutkan. Untuk array yang belum diurutkan, diperlukan penanganan tambahan berupa proses pengurutan terlebih dahulu sebelum algoritma Binary Search dapat diterapkan.

***Visualisasi Performa Binary Search***

Untuk memperjelas hasil analisis, berikut adalah grafik perbandingan performa *Binary Search* dan *Linear Search* berdasarkan waktu eksekusi:

```
import matplotlib.pyplot as plt
    # Data ukuran array
        ukuran_array = [10, 100, 1000]
    # Waktu eksekusi dalam milidetik (ms)
        binary_search = [1, 2, 6]
        linear_search = [2, 8, 15]
# Plotting grafik
        plt.figure(figsize=(8, 6))
        plt.plot(ukuran_array, binary_search, marker='o', linestyle='-', label='Binary Search', color='blue')
        plt.plot(ukuran_array, linear_search, marker='o', linestyle='-', label='Linear Search', color='red')
# Menambahkan judul dan label
        plt.title("Perbandingan Performa Pencarian")
        plt.xlabel("Ukuran Array")
        plt.ylabel("Waktu Eksekusi (ms)")
        plt.legend()
```

```
        plt.grid(True)
    # Menampilkan grafik
        plt.show()
```

**Implikasi Hasil Penelitian**

Hasil implementasi dan analisis algoritma *Binary Search* memiliki implikasi sebagai berikut:

**Aplikasi di Dunia Nyata**

Algoritma ini dapat diterapkan dalam sistem pencarian yang memerlukan efisiensi tinggi, seperti database, sistem perpustakaan digital, dan perangkat lunak pencarian. Digunakan dalam *search engine* dan perangkat navigasi data terstruktur.

**Optimasi Performa Aplikasi**

Penggunaan *Binary Search* dapat meningkatkan performa aplikasi yang memproses data besar dengan cepat. Namun, perlu memastikan bahwa data input telah diurutkan sebelum implementasi algoritma.

## Kesimpulan dan Saran

Berdasarkan implementasi dan analisis algoritma **Binary Search** yang telah dilakukan, berikut adalah kesimpulan utama:

**Efisiensi Waktu Eksekusi**

Algoritma *Binary Search* memiliki performa yang jauh lebih cepat dibandingkan dengan *Linear Search*, terutama pada dataset yang besar, Waktu eksekusi *Binary Search* meningkat secara **logaritmik** terhadap ukuran array, sesuai dengan kompleksitas waktu **O(log n)**. Pada array berukuran 1000 elemen, *Binary Search* hanya memerlukan **6 ms**, sedangkan *Linear Search* memerlukan **15 ms**.

**Jumlah Iterasi yang Dibutuhkan**, *Binary Search* memerlukan iterasi yang lebih sedikit dibandingkan *Linear Search* karena prinsip pembagian array menjadi dua bagian di setiap langkah. Jumlah iterasi pada *Binary Search* mendekati **log2(n)**, sehingga sangat optimal untuk pencarian data pada struktur yang terurut. **Pengaruh Lokasi Elemen Target** Jika elemen target berada di tengah, *Binary Search* menemukan elemen lebih cepat dengan jumlah iterasi minimal. Jika elemen target berada di akhir atau tidak ada dalam array, algoritma akan mencapai jumlah iterasi maksimum.

**Kelebihan dan Keterbatasan Binary Search**

**Kelebihan** yaitu: Performa sangat efisien untuk data besar yang sudah **terurut**. Kompleksitas waktu **O(log n)** menjadikannya lebih unggul dibandingkan algoritma pencarian linier.**Keterbatasan** yaitu: Algoritma hanya dapat bekerja jika data telah diurutkan sebelumnya. Jika data belum terurut, perlu dilakukan proses pengurutan terlebih dahulu yang dapat menambah waktu pemrosesan. **Implikasi Praktis yaitu:** Algoritma *Binary Search* sangat cocok diterapkan dalam sistem pencarian pada

**database**, aplikasi pencarian teks, *search engine*, dan sistem informasi dengan data terstruktur.

## Daftar Pustaka

Aminnur, M., Pakpahan, R. S., Alfarizi, D. G., Apriana, D., Rahmat, S. M., Fauzi, A., & Rosyani, P. (2023). Implementasi Metode Sequential Search Untuk Pengelolaan Data Barang Pada Sistem Aplikasi Sikilat Cargo. *Jurnal Ilmu Komputer Dan Pendidikan*, *1*(2), 283–287. https://journal.mediapublikasi.id/index.php/logic

Fauzi, M. M. (2023). Penggunaan Teknologi Informasi Untuk Meningkatkan Efektivitas Produksi Para Perusahaan. *Journal of Creative Power and Ambition*, *1*(1), 29–41. https://edujavare.com/index.php/jcpaWebsite:https://edujavare.com/

Ladjin, Litriani, Sahamony, Kusumaningrum, Maulina, Siregar, Hubbansyah, Solikin, Silitonga, Soeyatno, Asyari, Sinaga, A. (2022). *Www.Penerbitwidina.Com Www.Penerbitwidina.Com*.

Nasution, A., & Siddik, M. (2023). Implementasi Algoritma Binary Search Pada Aplikasi Kamus Indonesia-Inggris Berbasis Android. *Journal of Science and Social Research*, *6*(3), 711–716. https://www.jurnal.goretanpena.com/index.php/JSSR/article/view/1443

Mahrozi, N., & Faisal, M. (2023). Analisis perbandingan kecepatan algoritma selection sort dan bubble sort. *Scientica: Jurnal Ilmiah Sains dan Teknologi*, *1*(2), 89-98. http://repository.uin-malang.ac.id/17888/

Maghfiroh, J., Turmudi, T., & Susanti, E. (2023). Pengamanan pesan menggunakan algoritma one time paddengan linear congruential generator sebagai pembangkit kunci. *JRMM: Jurnal Riset Mahasiswa Matematika*, *2*(3), 122-131. http://repository.uin-malang.ac.id/13042/

Setiawan, M. N., Roring, R. S., Atma, Y. D., & Tetiawadi, H. (2024). Studi Empiris Terhadap Asistensi Artificial Intelligence (AI) Dalam Rancang Bangun Aplikasi. *Digital Transformation Technology*, *4*(1), 364–373. https://doi.org/10.47709/digitech.v4i1.4115

Wibowo, A., & Mico, F. (2021). Metode Parsing Tree Dalam Rancang Bangun Penerjemah Bahasa Indonesia Ke Bahasa Lampung. *Jurnal Portal Data*, *13*(1), 1–28. http://portaldata.org/index.php/portaldata/article/view/15%0Ahttp://portaldata.org/index.php/portaldata/article/download/15/13