

Implementasi Algoritma Pencarian Linear pada Data Mahasiswa Menggunakan Bahasa Pemrograman Go

Dika Aldiansyah¹, Rifqi Anwar Sidiq²

¹ Teknik Informatika, STMIK Tazkia

² Teknik Informatika, STMIK Tazkia

Abstrak

Pencarian data merupakan salah satu permasalahan dasar dalam bidang ilmu komputer yang sering digunakan dalam berbagai sistem informasi. Salah satu metode pencarian yang paling sederhana adalah algoritma pencarian linear (linear search).

Pada tugas UAS ini, dilakukan implementasi algoritma pencarian linear untuk mencari data mahasiswa berdasarkan Nomor Induk Mahasiswa (NIM) menggunakan bahasa pemrograman Go (Golang). Program ini bersifat interaktif sehingga pengguna dapat memasukkan NIM yang ingin dicari, baik RIFQI maupun DIKA. Data mahasiswa disimpan dalam struktur struct dan diolah menggunakan slice serta perulangan. Hasil dari program menunjukkan bahwa algoritma pencarian linear dapat bekerja dengan baik untuk jumlah data kecil, mudah diimplementasikan menggunakan Golang, dan berhasil menemukan mahasiswa berdasarkan NIM yang dimasukkan pengguna.

Kata kunci: Linear Search, Golang, Pencarian Data, Mahasiswa, Interaktif

1. Pendahuluan

Perkembangan teknologi informasi menuntut pengolahan data yang cepat dan efisien, termasuk dalam pengelolaan data mahasiswa. Salah satu proses dasar dalam pengolahan data adalah pencarian data. Algoritma pencarian berperan penting untuk menemukan informasi tertentu dari sekumpulan data.

Algoritma pencarian linear (Linear Search) merupakan algoritma pencarian paling sederhana yang bekerja dengan cara membandingkan data satu per satu secara berurutan hingga data yang dicari ditemukan atau seluruh data telah diperiksa. Meskipun sederhana, algoritma ini masih banyak digunakan, terutama pada data berukuran kecil hingga menengah.

Bahasa pemrograman Go dipilih karena memiliki performa tinggi, sintaks yang sederhana, serta dukungan kuat untuk pengolahan data. Oleh karena itu, penelitian ini bertujuan untuk mengimplementasikan algoritma pencarian linear pada data mahasiswa menggunakan bahasa pemrograman Go.

2. Metodologi

Metodologi penelitian ini menjelaskan representasi data, algoritma pencarian, dan rumus analisis secara singkat dan terstruktur.

A. Representasi Data :

Data mahasiswa direpresentasikan menggunakan struktur *struct* dan disimpan dalam *slice* pada bahasa pemrograman Go.

```
type Mahasiswa struct {  
    NIM   string  
    Nama  string  
}
```

Slice digunakan karena bersifat dinamis dan mudah diakses secara berurutan.

Algoritma Pencarian (Linear Search) :

Algoritma pencarian linear bekerja dengan membandingkan data satu per satu dari awal hingga akhir.

for i = 0 sampai n-1

jika $\text{data}[i].\text{NIM} == \text{key}$

data ditemukan

akhir jika

akhir for

Algoritma berhenti ketika data ditemukan atau seluruh data telah diperiksa.

Rumus dan Analisis :

Jika jumlah data = n , maka jumlah perbandingan dapat dinyatakan sebagai:

- Best case : 1
 - Average : $n / 2$
 - Worst case : n

Kompleksitas waktu:

$T(n) = O(n)$

Kompleksitas ruang:

$S(n) = O(1)$

Metode ini sederhana, tidak memerlukan data terurut, dan cocok untuk pencarian data mahasiswa dalam jumlah kecil.

Kode Program:

```

1  package main
2  import "fmt"
3
4  type Mahasiswa struct {
5      NIM string
6      Nama string
7  }
8
9  func sequentialSearch(data []Mahasiswa, nim string) int {
10    for i, mhs := range data {
11        if mhs.NIM == nim {
12            return i
13        }
14    }
15    return -1
16 }
17
18 }
```

```

1  func main() {
2      // Data mahasiswa
3      mahasiswa := []Mahasiswa{
4          {NIM: "251552010030", Nama: "RIFQI"},
5          {NIM: "251552010044", Nama: "DIKA"},
6      }
7
8      var nimDicari string
9      fmt.Println("Masukkan NIM yang dicari: ")
10     fmt.Scanln(&nimDicari)
11
12     index := sequentialSearch(mahasiswa, nimDicari)
13
14     // Tampilkan hasil
15     if index != -1 {
16         fmt.Printf("Mahasiswa ditemukan: %s\n", mahasiswa[index].Nama)
17     } else {
18         fmt.Println("Mahasiswa tidak ditemukan")
19     }
20 }
```

3. Hasil Dan Pembahasan

Hasil implementasi menunjukkan bahwa algoritma pencarian linear dapat bekerja dengan baik dalam mencari data mahasiswa berdasarkan NIM atau nama. Algoritma melakukan pengecekan data secara berurutan dari indeks awal hingga akhir.

Jika data yang dicari ditemukan, program akan menampilkan informasi mahasiswa yang sesuai. Namun, jika data tidak ditemukan setelah seluruh data diperiksa, program akan menampilkan pesan bahwa data tidak tersedia.

Kelebihan algoritma pencarian linear adalah kemudahan implementasi dan tidak memerlukan data yang terurut. Namun, kekurangannya terletak pada efisiensi waktu, karena pada kondisi terburuk algoritma harus memeriksa seluruh data. Hal ini membuat algoritma pencarian linear kurang optimal untuk data berukuran besar.

Penggunaan bahasa Go membuat proses implementasi lebih sederhana dan cepat karena sintaks yang ringkas serta manajemen memori yang baik.

Data mahasiswa yang digunakan:

- a. 251552010030 – RIFQI
- b. 251552010044 – DIKA

Hasil Pengujian

1. Input: 251552010030

Output:

Mahasiswa ditemukan: RIFQI

2. Input: 251552010044

Output:

Mahasiswa ditemukan: DIKA

3. Input NIM tidak tersedia:

Output:

Mahasiswa tidak ditemukan

4. Kesimpulan

Berdasarkan hasil penelitian, dapat disimpulkan bahwa algoritma pencarian linear dapat diimplementasikan dengan baik pada data mahasiswa menggunakan bahasa pemrograman Go (Golang). Algoritma ini cocok digunakan untuk pencarian data sederhana dengan jumlah data yang tidak terlalu besar. Meskipun memiliki keterbatasan dari segi efisiensi waktu, algoritma pencarian linear tetap relevan sebagai dasar pembelajaran algoritma pencarian dan cocok untuk aplikasi sederhana.

DAFTAR PUSTAKA

- [1] Zhang, Y., & Li, X. (2018). Implementation and performance analysis of linear search algorithms. *International Journal of Computer Applications*, 181(7), 25–30.
- [2] Putra, R. A., & Sari, D. P. (2020). Analisis algoritma pencarian data pada sistem informasi akademik. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, 7(3), 521–528.
- [3] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.
- [4] Sedgewick, R., & Wayne, K. (2011). Algorithms (4th ed.). Addison-Wesley.
- [5] Donovan, A. A., & Kernighan, B. W. (2016). The Go Programming Language. Addison-Wesley.
- [6] Go Team. (2024). The Go Programming Language Documentation. Diakses dari <https://golang.org>
- [7] Kernighan, B. W., & Pike, R. (1999). The Practice of Programming. Addison-Wesley.
- [8] Pressman, R. S. (2014). Software Engineering: A Practitioner's Approach. McGraw-Hill Education.
- [9] Sutabri, T. (2012). Analisis Sistem Informasi. Andi Offset.
- [10] Jogiyanto, H. M. (2010). Analisis dan Desain Sistem Informasi. Andi Offset.
- [11] Munir, R. (2015). Algoritma dan Pemrograman. Informatika Bandung.
- [12] Wirth, N. (1976). Algorithms + Data Structures = Programs. Prentice-Hall.
- [13] Kadir, A. (2018). Dasar Pemrograman dengan Go. Andi Offset.
- [14] Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education.
- [15] Knuth, D. E. (1998). The Art of Computer Programming, Volume 3: Sorting and Searching. Addison-Wesley.
- [16] Rosen, K. H. (2012). Discrete Mathematics and Its Applications. McGraw-Hill.
- [17] Rouse, M. (2023). Linear Search Algorithm. TechTarget.