# Aplikasi Algoritma DFS Untuk Mendeteksi Dependensi Sirkular Pada Bahasa C

Haidar Hamda - 13521105
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: 13521105@std.stei.itb.ac.id

***Abstract*— pada proyek pengembangan perangkat lunak, dependensi sirkular antara file dapat menyebabkan masalah seperti gagal kompilasi. Dalam konteks ini, dependensi sirkular mengacu pada situasi di mana dua atau lebih file saling bergantung secara siklikal, membentuk sebuah siklus struktur dependensi. Pada makalah ini, penulis mengusulkan penggunaan algoritma Depth First Search untuk mendeteksi dependensi sirkular pada Bahasa C. Algoritma DFS digunakan untuk melakukan pencarian graf dependensi antara file C.**

***Keywords—DFS; dependensi sirkular; Depth First Search***

## I. PENDAHULUAN

Dependensi sirkular terjadi ketika dua atau lebih file saling bergantung satu sama lain secara langsung maupun tidak langsung. Contoh sederhananya adalah Ketika file A bergantung pada file B, file B bergantung pada file C, tetapi File C bergantung pada file A. Dependensi sirkular menjadi masalah yang fatal pada pengembangan perangkat lunak.karena mengakibatkan gagal kompilasi atau perilaku yang tidak terduga saat program dijalankan. Oleh karena itu, pendeteksian dependensi sirkular merupakan hal yang penting dalam pengembangan perangkat lunak.

Salah satu metode yang efektif dalam mendeteksi keberadaan dependensi sirkular adalah dengan menggunakan algoritma Depth First Search (DFS). Algoritma DFS dapat menjelajahi graf dependensi dengan cara mendalam. Dengan Algoritma DFS, keberadaan siklus di dalam graf dependensi dapat ditentukan.

Dalam makalah ini, penulis memperkenalkan aplikasi algoritma Depth First Search untuk mendeteksi dependensi sirkular pada bahasa C. Dengan mendeteksi dependensi sirkular, pengembangan perangkat lunak dapat mengambil langkah yang tepat untuk mengatasi masalah dependensi sirkular seperti melakukan perubahan desain.

## II. TEORI DASAR

### A. *Dependensi Sirkular*

Dalam konteks pengembangan perangkat lunak, dependensi sirkular atau *circular dependency* merupakan suatu kondisi di mana suatu file atau modul bergantung pada file lain dan file tersebut juga bergantung pada file sebelumnya. Sebagai contoh file A bergantung pada file B dan file B bergantung pada file A.

Dependensi sirkular dapat menjadi masalah dalam pengembangan perangkat lunak. Dependensi sirkular dapat menyebabkan kegagalan kompilasi. Dependensi sirkular juga dapat membuat pemeliharaan perangkat lunak menjadi lebih kompleks.

### B. *Graf*

Graf didefinisikan sebagai G=(V,E) di mana V himpunan tidak kosong dari simpul-simpul, $V = \{v_1.v_2, ..., v_n\}$ dan E himpunan tidak kosong dari sisi-sisi yang menghubungkan simpul, $E = \{e_1, e_2, ..., e_n\}$.

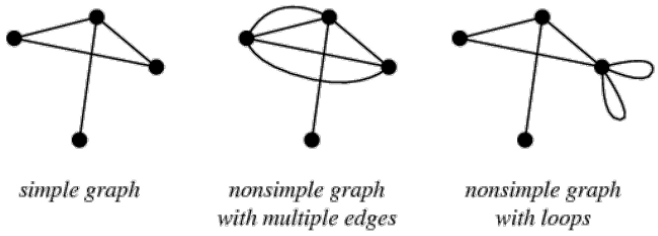
Graf dapat digolongkan menjadi dua jenis berdasarkan ada tidak nya gelang atau sisi ganda dalam sebuah graf:

1. Graf sederhana (simple graph)

   Graf yang tidak mengandung gelang maupun sisi ganda.

2. Graf tak-sederhana (non simple graph)

   Graf yang mengandung sisi ganda atau gelang




Gambar 1 ilustrasi graf sederhana dan graf tak-sederhana
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf

### C. *Penelusuran graf*

Penelusuran graf adalah proses mengunjungi setiap simpul pada graf dengan cara yang sistematik. Penelusuran graf bertujuan untuk mengunjungi setiap simpul yang ada di dalam graf.

Terdapat beberapa metode yang dapat digunakan untuk menelusuri graf, salah satunya adalah algoritma Depth First Search atau DFS.

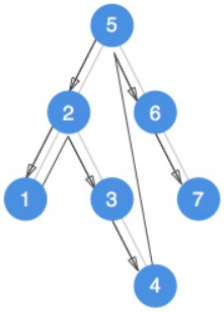
Selain DFS, terdapat algoritma lain untuk melakukan penelusuran graf yaitu Breadth First Search (BFS). Kedua algoritma DFS dan BFS dikategorikan ke dalam jenis penjelajahan blind search atau uninformed search.

### D. *Depth First Search*

Depth First Ssearch (DFS) adalah salah satu algoritma pencarian graf. Algoritma ini melakukan pencarian secara mendalam. Berawal dari simpul awal yang ditentukan, kemudian melanjutkan pengunjungan ke simpul tetangga yang belum dikunjungi. DFS menjelajahi simpul secara mendalam sebelum memeriksa simpul lain.

Pada umumnya, cara kerja algoritma DFS sebagai berikut:

1. Pilih simpul awal sebagai simpul saat ini dan menandainya sebagai simpul yang sudah dikunjungi.
2. Memilih salah satu simpul tetangga yang belum dikunjungi, tandai tetangga tersebut sebagai simpul yang sudah dikunjungi, dan memanggil fungsi DFS dengan simpul tetangga tersebut sebagai simpul saat ini.
3. Ulangi langkah 2 sampai semua simpul telah dikunjungi.




Gambar 2. Visualisasi algoritma DFS

Sumber: https://nick.balestrafoster.com/2015/depthFirst-vs-breadthFirst/

Algoritma DFS umum nya digunakan untuk permasalahan seperti pencarian jalur atau penelusuran graf. Algoritma DFS memiliki kompleksitas waktu O(b^n).

Selama penelusuran graf, setiap simpul yang dikunjungi ditandai sebagai simpul yang sudah pernah dikunjungi. Hal ini dilakukan untuk mengindari penjelajahan secara tak terbatas dalam siklus.

## III. IMPLEMENTASI

Dalam pendeteksian dependensi sirkular, akan diaplikasikan algoritma DFS. Algoritma DFS ini akan menelusuri graf dependensi suatu file C. berikut adalah implementasi program dalam Bahasa python yang penulis buat

```
import re


def getDependencies(file):
    f=open(file, 'r').read()
    includeRegex
=r'#include\s*[<"](.*)[>"]'
    res = re.findall(includeRegex, f)
    return res


def checkCircularDependency(file):
    visited = []
    stack = []
    def dfs(file):
        visited.append(file)
        stack.append(file)
        dependencies
=getDependencies(file)
        for dependency in dependencies:
            if dependency not in visited:
                if dfs(dependency):
                    return True
            elif dependency in stack:
                return True
        stack.remove(file)
        return False
    return dfs(file)
```

Berdasarkan kode di atas, terdapat fungsi getDependencies dengan parameter file, parameter file ini merupakan nama dari file yang akan didapatkan dependensi nya. Fungsi ini akan mencari dependensi dengan mencocokkan regex “#include [<“](.*)[>”]” dan mengembalikan isi dari bagian “(.*)” yang memenuhi.

Terdapat pula fungsi checkCircularDependency dengan parameter file yang merupakan nama file yang akan dicheck apakah ada dependensi sirkular atau tidak. Di dalam fungsi ini terdapat fungsi dfs yang akan menelurusi file C. Fungsi dfs akan mengembalikan True jika dependency pada file yang sedang diperiksa telah ada pada stack. Stack berisikan kedalaman dependensi, jika terdapat file 1.c yang meng-

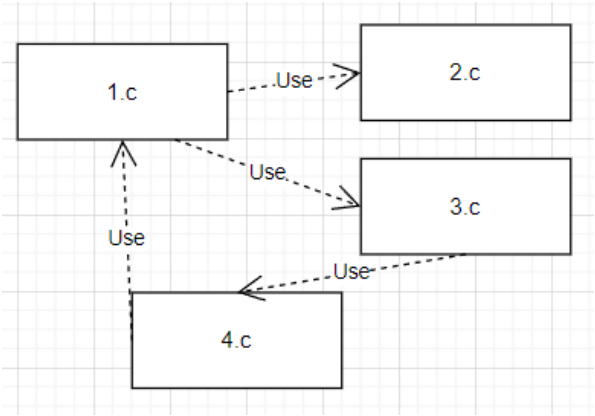
include 2.c dan 2.c meng-include 3.c, stack akan berisi [1.c, 2.c, 3.c], jika 1 juga meng-include 4.c, setelah menelusuri sampai 3.c, stack akan berisi [1.c, 4.c].

## IV. Uji Coba

Akan dilakukan dua pengujian pada makalah ini. Pengujian yang akan dilakukan adalah pengujian terhadap file C yang terdapat dependensi sirkular dan file C yang tidak terdapat dependensi sirkular.

### *A. Pengujian pada file dengan dependensi sirkular*

Pengujian pertama dilakukan terhadap file C yang memiliki graf dependensi sebagai berikut:




Gambar 3. Graf depedensi file pengujian 1

Sumber: Dokumentasi pribadi

Dapat dilihat bahwa file 1.c memiliki dependensi sirkular dengan meng-include 3.c, 3.c include 4.c dan 4.c include 1.c

Hasil eksekusi program:

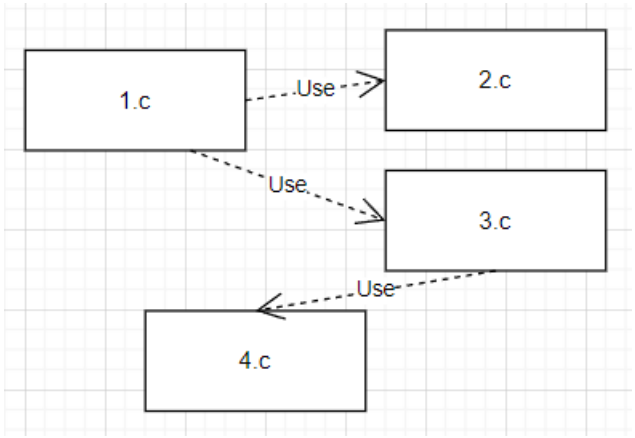
```
PS C:\Users\User\Documents\makalah> & C
dependesi sirkular ditemukan
execution time:  0.00099945068359375 s
```

Gambar 4. Hasil pengujian untuk file dengan dependensi sirkular

Sumber: Dokumentasi pribadi

### *B. Pengujian pada file tanpa dependensi sirkular*

Pengujian kedua dilakukan terhadap file C yang memiliki graf dependensi sebagai berikut:




Gambar 5. Graf depedensi file pengujian 1

Sumber: Dokumentasi pribadi

Berbeda dengan file yang digunakan pada uji coba sebelum nya, 4.c tidak meng-include 1.c yang mengakibatkan dependensi sirkular.

Hasil eksekusi program:

```
PS C:\Users\User\Documents\makalah> & C:/
tidak ada dependensi sirkular
execution time:  0.0010137557983398438 s
```

Gambar 6. Hasil pengujian untuk file tanpa dependensi sirkular

Sumber: Dokumentasi probadi

## V. Kesimpulan

Algoritma DFS dapat digunakan untuk mendeteksi keberadaan dependensi sirkular pada suatu file C. Dengan program yang dikembangkan, pemrogram tidak perlu pusing lagi untuk mencari penyebab error saat kompilasi.

Pada penelitian ini, program yang dibuat ialah program yang hanya bisa mengembalikan ada atau tidaknya dependensi sirkular, sebagai saran, program dapat dikembangkan lagi dengan menambah fitur seperti menampilkan file mana saja yang mengakibatkan dependensi sirkular.

## Ucapan Terima Kasih

Pertama-tama penulis panjatkan rasa syukur kepada Tuhan Yang Maha Esa karena atas berkat dan rahmatnya penulis dapat menyelesaikan makalah ini. Penulis juga mengucapkan terimakasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T. selaku dosen pengajar mata kuliah strategi algoritma kelas 01 tahun ajaran 2022/2023 yang telah memberikan bimbingan dan pengajaran selama satu semester penuh.

## References

[1] Munir, Rinaldi. 2020. "Graf (bagian 1)". https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 22 mei 2023

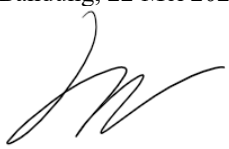
[2] https://nick.balestrafoster.com/2015/depthFirst-vs-breadthFirst/ diakses pada 22 mei 2023

[3] Munir, Rinaldi. 2021 “Breadth/Depth First Search (BFS/DFS)”. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag1.pdf diakses pada 22 mei 2023

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 22 Mei 2023



Haidar Hamda, 13521105