

LAPORAN TUGAS BESAR 2

**IF2211/Strategi Algoritma
Semester II Tahun 2020/2021**



Dipersiapkan oleh:

Kelompok Enemyster

Muhammad Tito Prakasa	13519007
-----------------------	----------

Fakhri Nail Wibowo	13519035
--------------------	----------

Habibina Arif Muzayyan	13519125
------------------------	----------

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

Daftar Isi

Deskripsi Tugas	3
Spesifikasi GUI:	3
Spesifikasi Wajib:	4
Landasan Teori	5
Dasar Teori Secara Umum	6
C# Desktop Application Development	7
Analisis Pemecahan Masalah	8
Langkah-langkah pemecahan masalah	8
Proses mapping persoalan menjadi elemen-elemen algoritma BFS dan DFS	8
Contoh ilustrasi kasus lain	8
Implementasi dan Pengujian	11
Implementasi Program	11
Struktur Data Program	12
Tata Cara Menggunakan Program	13
Hasil Pengujian	14
Analisis Desain BFS dan DFS pada Pemecahan Solusi	18
Analisa BFS pada Explore Friend	18
Analisa DFS pada Explore Friend	19
Analisa BFS pada Friend Recommendation	20
Kesimpulan dan Saran	21
Daftar Pustaka	22

I. Deskripsi Tugas

Aplikasi yang akan dibangun dibuat berbasis GUI. Berikut ini adalah contoh tampilan dari aplikasi GUI yang akan dibangun.

People You May Know

Graph File : Graph1.txt

Algorithm : ☐ DFS ☐ BFS

<Visualisasi Graph>

Choose Account :

Explore friends with :

Friends Recommendations for A:

- ☐ B (A -> C -> B, 1st Degree)
2 Mutual Friends : C, D
- ☐ C
3 Mutual Friends : E, F, G
-
-
-

Spesifikasi GUI:

1. Program dapat menerima input berkas file eksternal dan menampilkan visualisasi graph.
2. Program dapat memilih algoritma yang digunakan.
3. Program dapat memilih akun pertama dan menampilkan friends recommendation untuk akun tersebut.
4. Program dapat memilih akun kedua dan menampilkan jalur koneksi kedua akun dalam bentuk visualisasi graf dan teks bertuliskan jalur koneksi kedua akun.
5. GUI dapat dibuat sekreatif mungkin asalkan memuat 4 spesifikasi di atas.

Spesifikasi Wajib:

1. Buatlah program dalam bahasa C# untuk melakukan penelusuran social network facebook sehingga diperoleh daftar rekomendasi teman yang sebaiknya di-add. Penelusuran harus memanfaatkan algoritma BFS dan DFS.
2. Awalnya program menerima sebuah berkas file eksternal yang berisi informasi pertemanan di facebook. Baris pertama merupakan sebuah integer N yang adalah banyaknya pertemanan antar akun di facebook. Sebanyak N baris berikutnya berisi dua buah string (A, B) yang menunjukkan akun A dan B sudah berteman (lebih jelasnya akan diberikan contoh pada bagian 3).
3. Program kemudian dapat menampilkan visualisasi graf pertemanan berdasarkan informasi dari file eksternal tersebut. Graf pertemanan ini merupakan graf tidak berarah dan tidak berbobot. Setiap akun facebook direpresentasikan sebagai sebuah node atau simpul pada graf. Jika dua akun berteman, maka kedua simpul pada graf akan dihubungkan dengan sebuah busur. Proses visualisasi ini boleh memanfaatkan pustaka atau kakas yang tersedia. Sebagai referensi, salah satu kakas yang tersedia untuk melakukan visualisasi adalah MSAGL

<https://github.com/microsoft/automatic-graph-layout>

Berikut ini adalah panduan singkat terkait penggunaan MSAGL oleh tim asisten yang dapat diakses pada:

<https://docs.google.com/document/d/1XhFSpHU028Gaf7YxkmdbluLkQgVl3MY6gt1tPL30LA/edit?usp=sharing>

4. Terdapat dua fitur utama, yaitu:
 - a. Fitur friend recommendation
 - i. Program menerima sebuah pilihan akun dari user yang hendak dicari rekomendasi temannya. Pemilihan nama akun akan diterima melalui GUI. Cara pemilihan dibebaskan, bisa input dari keyboard atau meng-klik langsung sebuah node dari graf.
 - ii. Program akan menampilkan daftar rekomendasi teman seperti pada fitur People You May Know facebook berupa nama akun tersebut secara terurut mulai dari mutual friend terbanyak antar kedua akun beserta daftar nama akun mutual friend.

- b. Fitur explore friends
 - i. Dua akun yang tidak memiliki mutual friend, masih memiliki peluang untuk berteman jika kedua akun mempunyai common Nth degree connection, yaitu jalur yang menghubungkan kedua akun yang terpisah sejauh N akun (node pada graf).
 - ii. Program menerima pilihan dua akun yang belum berteman.
 - iii. Program akan menampilkan nilai N-th degree connection antar kedua akun dan memberikan jalur melalui akun mana saja sampai kedua akun bisa terhubung.
 - iv. Dari graph yang sudah dibentuk, aplikasi harus dapat menyusun jalur koneksi hasil explore friends antara akun satu dengan akun yang ingin dituju. Aplikasi juga harus dapat menunjukkan langkah-langkah pencarian, baik dengan algoritma BFS maupun DFS.
 - v. Jika tidak ditemukan jalur koneksi sama sekali antar kedua akun karena graf not fully connected, maka tampilkan informasi bahwa kedua akun tidak dapat terhubung.
- 5. Mahasiswa tidak diperkenankan untuk melihat atau menyalin library lain yang mungkin tersedia bebas terkait dengan pemanfaatan BFS dan DFS.

II. Landasan Teori

A. Dasar Teori Secara Umum

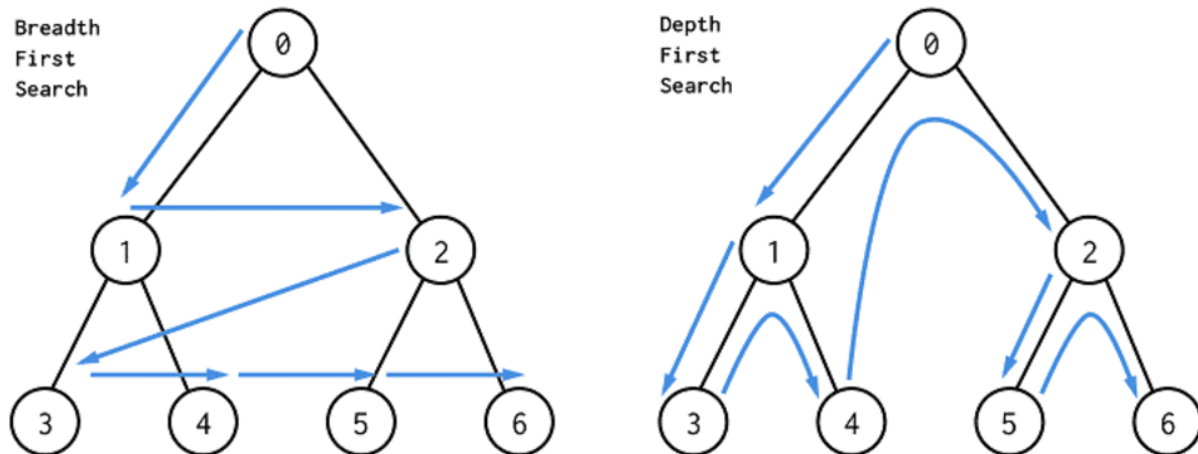
Graf adalah sebuah struktur data yang terdiri dari simpul dan sisi, sisi akan menghubungkan simpul-simpul dengan satu sama lain atau dengan simpul itu sendiri. Graf bisa digunakan untuk merepresentasikan berbagai masalah di dunia nyata. *Graph traversal* adalah kumpulan algoritma atau cara sistematis untuk menelusuri simpul-simpul dari graf sesuai dengan kebutuhan. *Graph traversal* berguna untuk mencari solusi dari masalah yang direpresentasikan dengan graf.

Terdapat dua jenis pendekatan graf, yaitu graf statis dan dinamis. Pada pendekatan graf statis, graf direpresentasikan sebagai struktur data dan sudah terbentuk sebelum proses pencarian. Sedangkan, pada graf dinamis graf dibangun selama pencarian solusi. Pada tugas besar ini, kami akan menggunakan pendekatan graf statis dengan menggunakan Breadth First Search (BFS) dan Depth First Search (DFS).

Breadth First Search (BFS) adalah algoritma untuk menelusuri graf dengan mengunjungi semua simpul tetangga dari simpul pusat terlebih dahulu sebelum lanjut ke level berikutnya. BFS biasanya menggunakan struktur data queue untuk membantu mengingat simpul yang akan dikunjungi dan menggunakan struktur data array dengan nilai boolean untuk mengingat simpul yang sudah dikunjungi. BFS memiliki kompleksitas waktu $O(|E|+|V|)$ dengan E adalah jumlah sisi dan V jumlah simpul. BFS lebih optimal dalam mencari jarak terpendek dan lebih cocok untuk mencari simpul yang dekat dengan pusat.

Depth First Search (DFS) adalah algoritma untuk menelusuri graf dengan mengunjungi simpul yang bertetangga dengan simpul pusat lalu mengunjungi simpul yang bertetangga dengan simpul tersebut dan seterusnya. DFS biasanya menggunakan struktur data stack dan membutuhkan memori yang lebih sedikit daripada BFS. DFS cocok untuk mencari simpul yang jauh dari simpul pusat.

Gambar di bawah ini menggambarkan perbedaan algoritma BFS dengan DFS.



B. C# Desktop Application Development

Pada pengembangan aplikasi dengan C#, terdapat bermacam-macam pilihan termasuk Windows Forms, WPF, UWP, dan lain lain. Untuk cara deploy aplikasi juga terdapat beberapa pilihan seperti .NET Core dan .NET Framework.

Pada tugas ini, kami menggunakan Windows Forms untuk membuat GUI interaktif. Kami juga menggunakan .NET Core sebagai framework deployment agar aplikasi bisa berjalan dengan hanya membuka file .exe.

Windows Forms pertama kali dirilis pada tahun 2002 dan merupakan yang paling sering digunakan untuk membuat aplikasi Windows berdasarkan Windows graphics device interface (GDI) engine. Dengan Windows Forms, pengembangan GUI aplikasi dimudahkan karena bisa menggunakan drag-and-drop sehingga tidak perlu terlalu banyak menulis kode. Selain itu, Windows Forms juga didukung oleh semua versi Windows dan juga didukung oleh .NET Core 3.0 dan versi seterusnya.

.NET Core adalah versi baru dari .NET Framework yang merupakan framework gratis buatan Windows yang berfungsi untuk membuat berbagai macam aplikasi. Framework ini dirilis oleh Microsoft dengan tujuan untuk bisa mendukung berbagai macam OS untuk .NET, tidak hanya Windows saja. Namun, pada tugas kali ini kami hanya mencoba merilis untuk Windows saja.

III. Analisis Pemecahan Masalah

A. Langkah-langkah pemecahan masalah

Dalam pembuatan program, kami memecahkan masalah dengan menyelesaikan cara kerja programnya dahulu. Kemudian, kami membuat tampilan program dan menyatukannya dengan cara kerja program. Terakhir, kami melakukan pengujian dan memperbaiki kesalahan. Untuk menyelesaikan cara kerja program, kami bagi menjadi 3 bagian yaitu friend recommendation, explore friend BFS, dan explore friend DFS. Untuk membuat tampilan program dan menyatukan dengan cara kerja program, kami bagi menjadi 3 bagian juga yaitu membuat visualisasi graf, membuat tampilan desain program, dan membuat tampilan program menjadi interaktif. Untuk pengujian dan perbaikan, kami melakukannya secara bersama-sama. Adapun untuk pembagian tugasnya adalah sebagai berikut:

1. Muhammad Tito Prakasa : explore friend DFS dan Membuat tampilan program menjadi interaktif.
2. Fakhri Nail Wibowo : friend recommendation dan membuat visualisasi graph
3. Habibina Arif Muzayyan : explore friend BFS dan membuat tampilan desain program.

B. Proses mapping persoalan menjadi elemen-elemen algoritma BFS dan DFS

Algoritma BFS dan DFS mempunyai elemen-elemen untuk membantu penyelesaian masalah. Untuk algoritma BFS, mempunyai elemen-elemen sebagai berikut:

1. Queue berupa List untuk menyimpan simpul yang ingin dikunjungi.
2. Dictionary yang keynya bernilai semua simpul yang ada dan valuenya bernilai apakah simpul tersebut sudah dikunjungi.
3. List yang berisi simpul-simpul solusi

Sedangkan algoritma DFS mempunyai elemen-elemen sebagai berikut:

1. Stack yang menyimpan simpul-simpul solusi
2. Fungsi rekursif untuk melakukan perulangan

C. Contoh ilustrasi kasus lain

1. testcase1.txt

```
10
A B
A C
B C
B E
B F
```


C F
C G
D G
E H
E F

2. testcase2.txt

16
A B
A C
A D
B C
B E
B F
C F
C G
D G
D F
E H
E F
F H
J I
H K
K L

3. testcase3.txt

4. testcase4.txt

5
A C
A E
B C
E D
E F

5. testcase5.txt

```
15
A B
A C
B D
B E
C D
C J
D H
E G
E F
G A
G E
H I
H J
I J
I B
```

6. testcase6.txt

```
10
test aih
aih kece
kece broo
test kece
broo aih
asd test
aih sec
hai aih
kece asd
vii broo
```

IV. Implementasi dan Pengujian

A. Implementasi Program

Program kali ini bekerja dalam paradigma *object oriented programming* sehingga program bekerja berdasarkan interaksi antar tombol-tombol. Berikut gambarannya:

```
if (isBrowseButtonClicked){
    listOfEdge = readFile.convertToList;
    graph = makeGraph(listOfEdge);
    show.graph;
}
```

```
if (isBFSButtonClicked){
    if (comboBox1.choosed){
        hasilFR =
processFiturRecommendation(valueOf.comboBox1);
    }
    if (comboBox2.choosed){
        hasilEF = processExploreFriendBFS(valueOf.comboBox1,
valueOf.comboBox2);
    }
    when submitButtonClicked{
        show.hasilFR.panel2;
        show.hasilEF.panel1;
    }
    }else if (isDFSButtonClicked){
        if (comboBox1.choosed){
            hasilFR =
processFiturRecommendation(valueOf.comboBox1);
        }
        if (comboBox2.choosed){
            hasilEF = processExploreFriendDFS(valueOf.comboBox1,
valueOf.comboBox2);
        }
        when submitButtonClicked{
            show.hasilFR.panel2;
            show.hasilEF.panel1;
        }
    }
}
```

```
}
```

B. Struktur Data Program

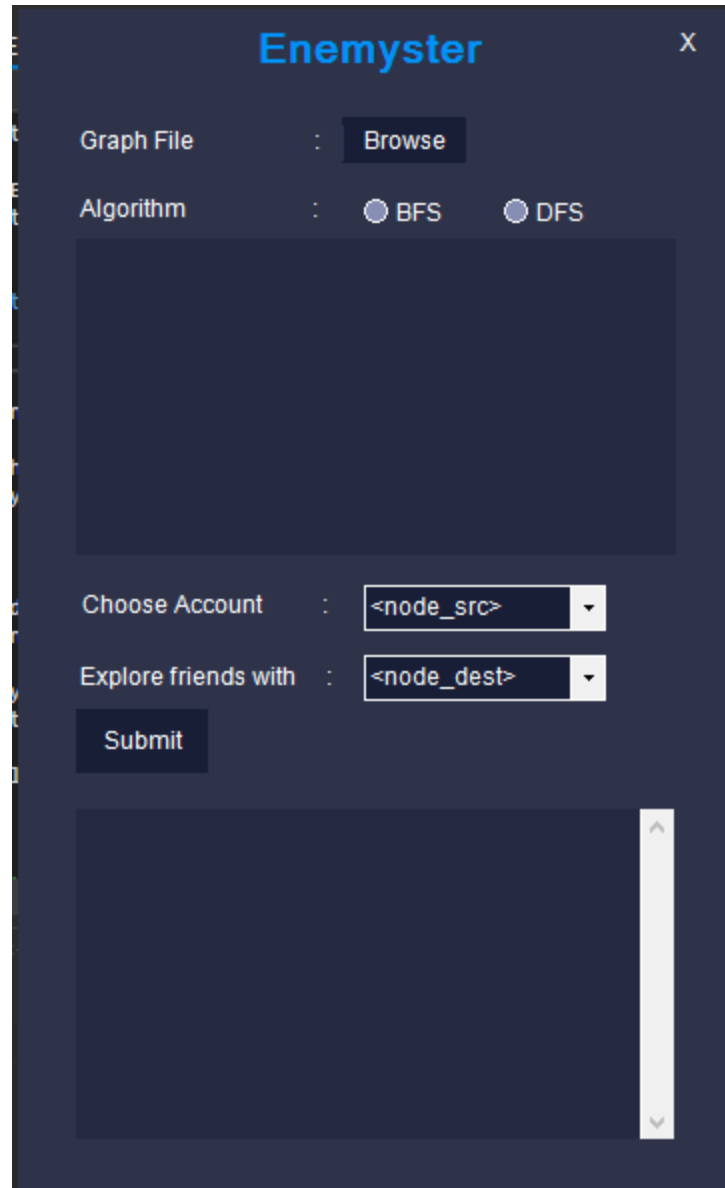
Program kami menggunakan *derived class Form* dari *WindowsFormsApp* dengan bantuan *class Graph* buatan kami sendiri untuk membantu pengolahan data. *Class Graph* memiliki atribut:

1. Dictionary<string, List<string>>, dengan *key*-nya merupakan node dan *value*-nya adalah edge kemana saja node tersebut terhubung
2. countVertices, jumlah node pada graph
3. countEdges, jumlah edge pada graph

Selanjutnya *class Graph* ini memiliki method-method:

1. exploreFriendBFS(), yaitu method yang mengolah atribut dictionary pada graph menjadi sebuah list of string (node) solusi. Method ini melakukan pencarian lintasan apakah seseorang dengan orang yang lain terhubung dengan cara iteratif algoritma BFS. Tambahan struktur data pada method ini adalah queue.
2. exploreFriendDFS(), yaitu method yang mengolah atribut dictionary pada graph menjadi sebuah list of string (node) solusi. Method ini melakukan pencarian lintasan apakah seseorang dengan orang yang lain terhubung dengan cara rekursif algoritma DFS. Tambahan struktur data pada method ini adalah stack.
3. friendRecommendationBFS(), yaitu method yang mengolah atribut dictionary pada graph menjadi sebuah dictionary baru dengan key adalah teman yang direkomendasikan dan valuenya adalah mutual friend. Method ini melakukan pencarian solusi menggunakan cara iteratif algoritma BFS. Tambahan struktur data pada method ini adalah queue.

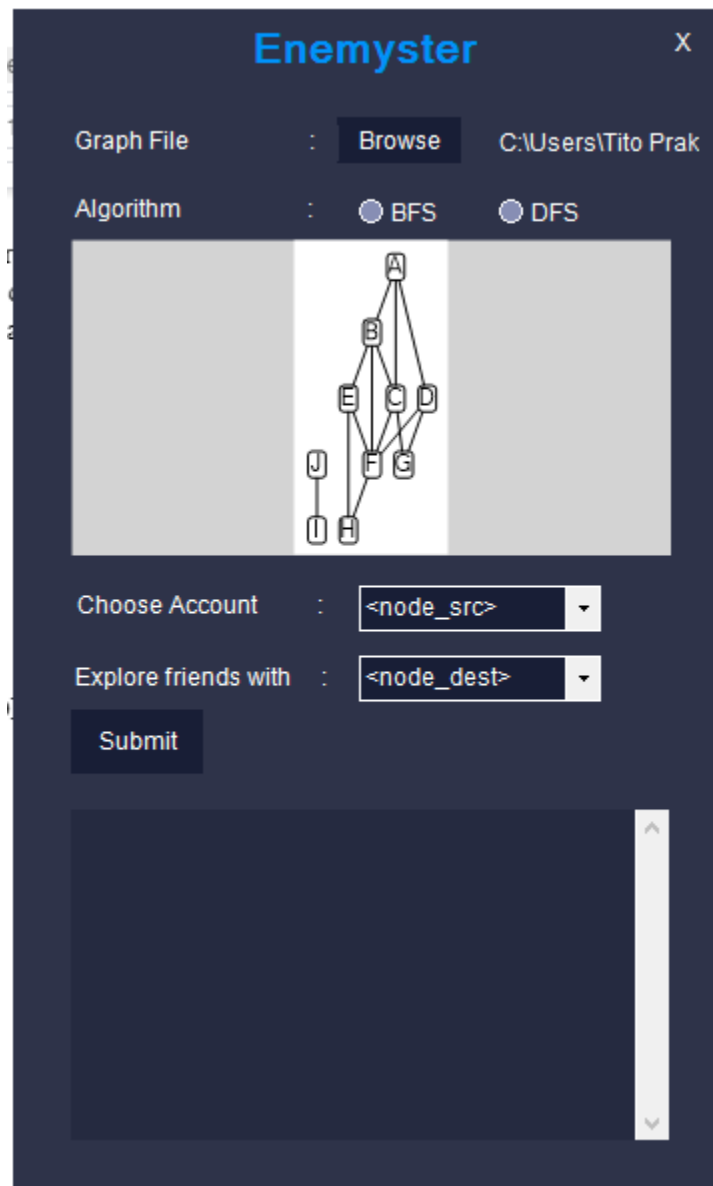
C. Tata Cara Menggunakan Program

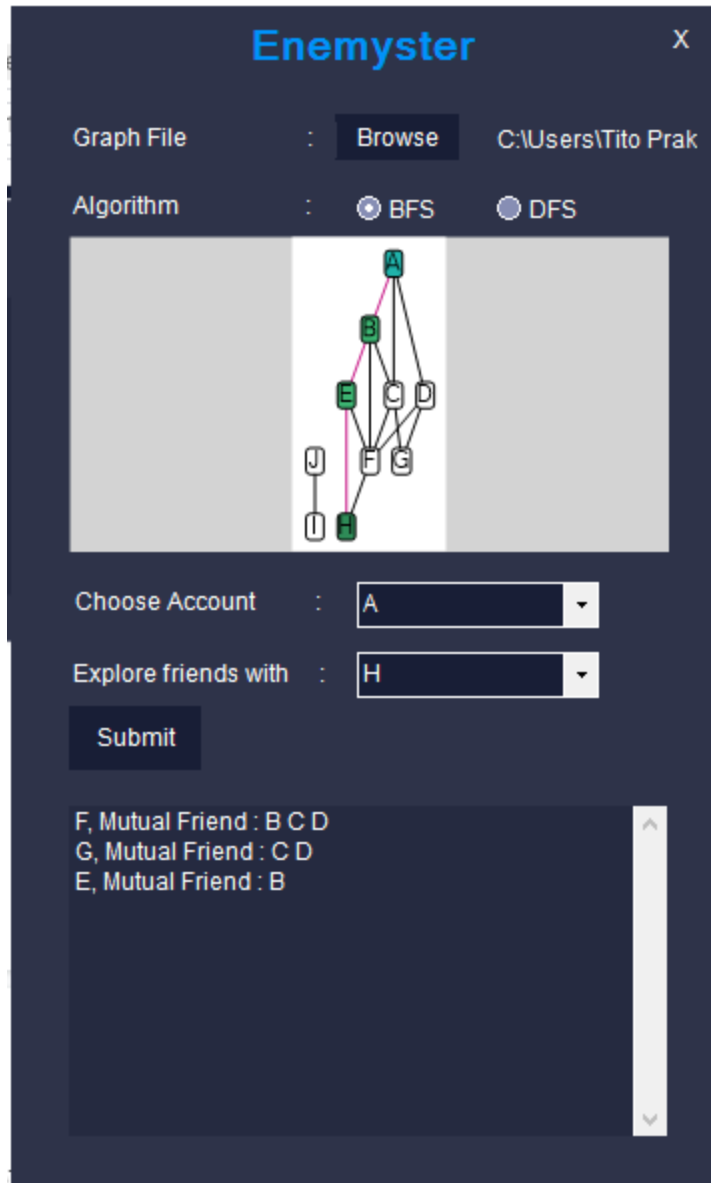


1. Buka file .exe pada folder bin untuk menjalankan program.
2. Untuk memasukkan file.txt yang diinginkan klik tombol Browse dan pilih file yang diinginkan. Selesai anda browse maka akan muncul tampilan graph pada panel pertama.
3. Untuk menggunakan fitur explore friend pilih algoritma yang ingin digunakan, akun pertama dan akun kedua, kemudian klik submit untuk memunculkan lintasan hubungan antara akun pertama dan akun kedua. Akan tidak berubah jika akun pertama tidak memiliki jalur koneksi dengan akun kedua.

4. Untuk menggunakan fitur friend recommendation pilih algoritma BFS dan pilih akun yang ingin dicari friend recommendationnya (combo box choose account), kemudian klik submit untuk memunculkan daftar friend recommendation beserta mutual friendnya yang akan muncul di panel ke-2.

D. Hasil Pengujian





Enemyster

Graph File

:

Browse

C:\Users\Tito Prak

Algorithm

:

☐ BFS

☒ DFS

Choose Account

:

A

Explore friends with

:

H

Submit

F, Mutual Friend : B C D

G, Mutual Friend : C D

E, Mutual Friend : B

Enemyster

Graph File

:

Browse

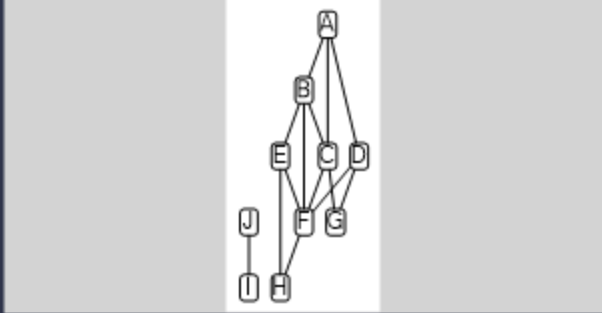
C:\Users\Tito Prak

Algorithm

:

☐ BFS

☒ DFS



Choose Account

:

A

Explore friends with

:

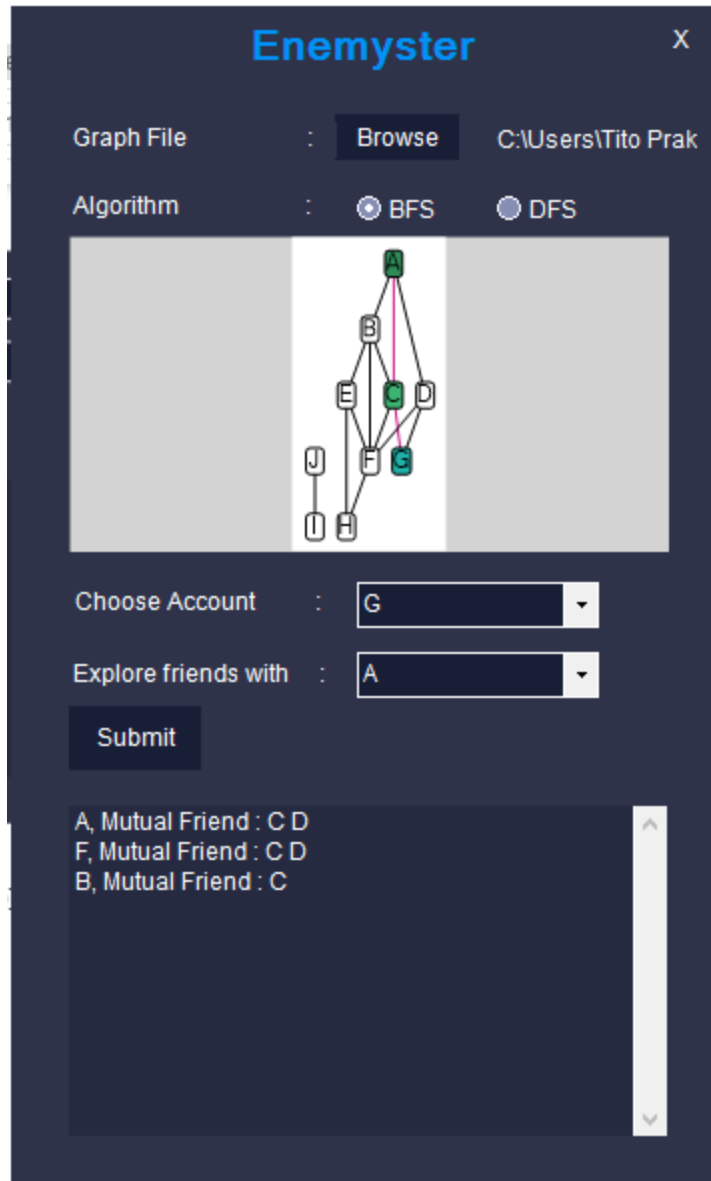
J

Submit

F, Mutual Friend : B C D

G, Mutual Friend : C D

E, Mutual Friend : B



E. Analisis Desain BFS dan DFS pada Pemecahan Solusi

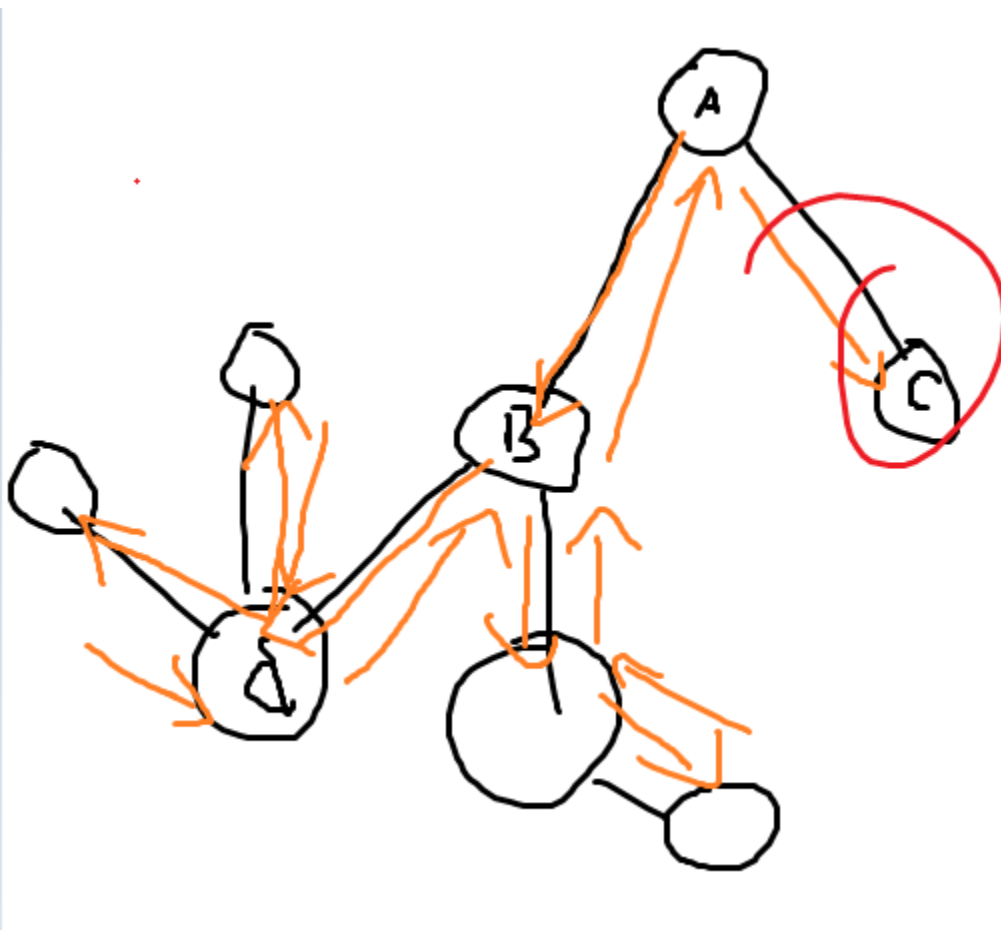
Analisa BFS pada Explore Friend

Menurut saya, pencarian solusi untuk fitur explore friend menggunakan BFS lebih baik daripada DFS karena lintasan solusi yang didapatkan lebih pendek. Untuk pencarian solusi, BFS menggunakan queue sebagai urutan pengunjungan simpul. Semua simpul yang bertetangga dengan simpul yang dikunjungi akan ditambahkan ke queue. Simpul yang telah dikunjungi tidak bisa dikunjungi lagi. Pencarian solusi dimulai dari simpul awal hingga simpul yang dikunjungi merupakan simpul tujuan. Tetapi untuk segi kompleksitas memori membutuhkan memori yang lebih besar dari DFS.

Analisa DFS pada Explore Friend

Menurut saya sendiri DFS untuk fitur explore friend merupakan algoritma yang cukup unik karena secara kasat mata algoritma ini seperti melalui semua node dua kali (kasus worst case tidak ditemukan solusi). Padahal sebenarnya hal itu hanyalah lanjutan pencarian solusi selanjutnya (pendekatan rekursif) jika ditemukan lintasan yang bukan solusi. Untuk membantu kasus backtrack, DFS memerlukan sebuah stack (setidaknya di program kami) yang mencatat node terakhir dilalui. Untuk segi kompleksitas waktu sepertinya mirip-mirip dengan BFS hanya saja algoritma ini kurang efektif jika graf yang dikerjakan adalah graf kompleks dan solusinya berada dekat dengan node awal.

Contoh visualisasi node A ingin ke node C tetapi karena menggunakan pendekatan BFS, mau tidak mau dia harus mendalami node B terlebih dahulu:



Analisa BFS pada Friend Recommendation

Pada Friend Recommendation diberikan pilihan untuk menggunakan algoritma BFS atau DFS. Simpul yang dicari pada Friend Recommendation adalah simpul yang memiliki simpul tetangga yang sama dengan simpul pusat, sehingga jarak dari simpul pusat ke simpul target hanya 2. Karena jarak yang dekat, maka algoritma BFS lebih cocok daripada DFS dalam mencari solusi.

V. Kesimpulan dan Saran

Tugas besar ini mengasah pemahaman kami tentang algoritma BFS dan DFS, terutama dalam mengembangkan ide bagaimana mengkonversi algoritma kasat mata ke dalam kode. Dan kesimpulan yang kami dapatkan BFS cukup efektif jika solusi yang dibutuhkan “dari” node awal, sedangkan DFS cukup efektif jika solusi yang dibutuhkan “jauh” dari node awal.

Seru sih, apalagi sebenarnya kondisi kami, salah satu *contributor* belum terlalu memahami DFS tetapi sebagian tugas DFS (bikin ide sendiri, konversi sendiri, eh terus bisa jalan aaa seneng banget >.<). Selanjutnya, saat nyambungin dari console app (class graph) ke windows app (MSAGL) sebenarnya engga kepikiran tapi terus nyoba-nyoba dan akhirnya jadi lebih paham OOP ketika udah jadi interactive apps juga hehe, mantups.

Saran untuk tubes selanjutnya, jangan janji gitu deadlinenya. Ga baik. But, It's not worth if u not having fun ^.^)b

Daftar Pustaka

- ❑ [Tugas-Besar-2-IF2211-Strategi-Algoritma-2021.pdf \(itb.ac.id\)](#)
- ❑ [Breadth/Depth First Search IF2211 ITB 2021](#)
- ❑ [Difference between Depth First Search and Breadth First Search - DEV Community](#)
- ❑ [Tutorialspoint.com | Syntax of C#](#)
- ❑ [GeeksforGeeks | Syntax of C#](#)