

LAPORAN TUGAS BESAR

Pengaplikasian Algoritma BFS dan DFS dalam Fitur People You May Know Jejaring Sosial Facebook

Dibuat dalam rangka:
Tugas Besar 2 IF-2211 Strategi Algoritma

Oleh:

Temannya manusia paling ganteng sedunia	Manusia paling ganteng sedunia	Gatau siapa
Wisnu Aditya Samiadji	Syihabuddin Yahya Muhammad	Ahmad Saladin
13519093	13519149	13519187

“EchoChamber”



PROGRAM STUDI TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
2020

DAFTAR ISI

DAFTAR ISI

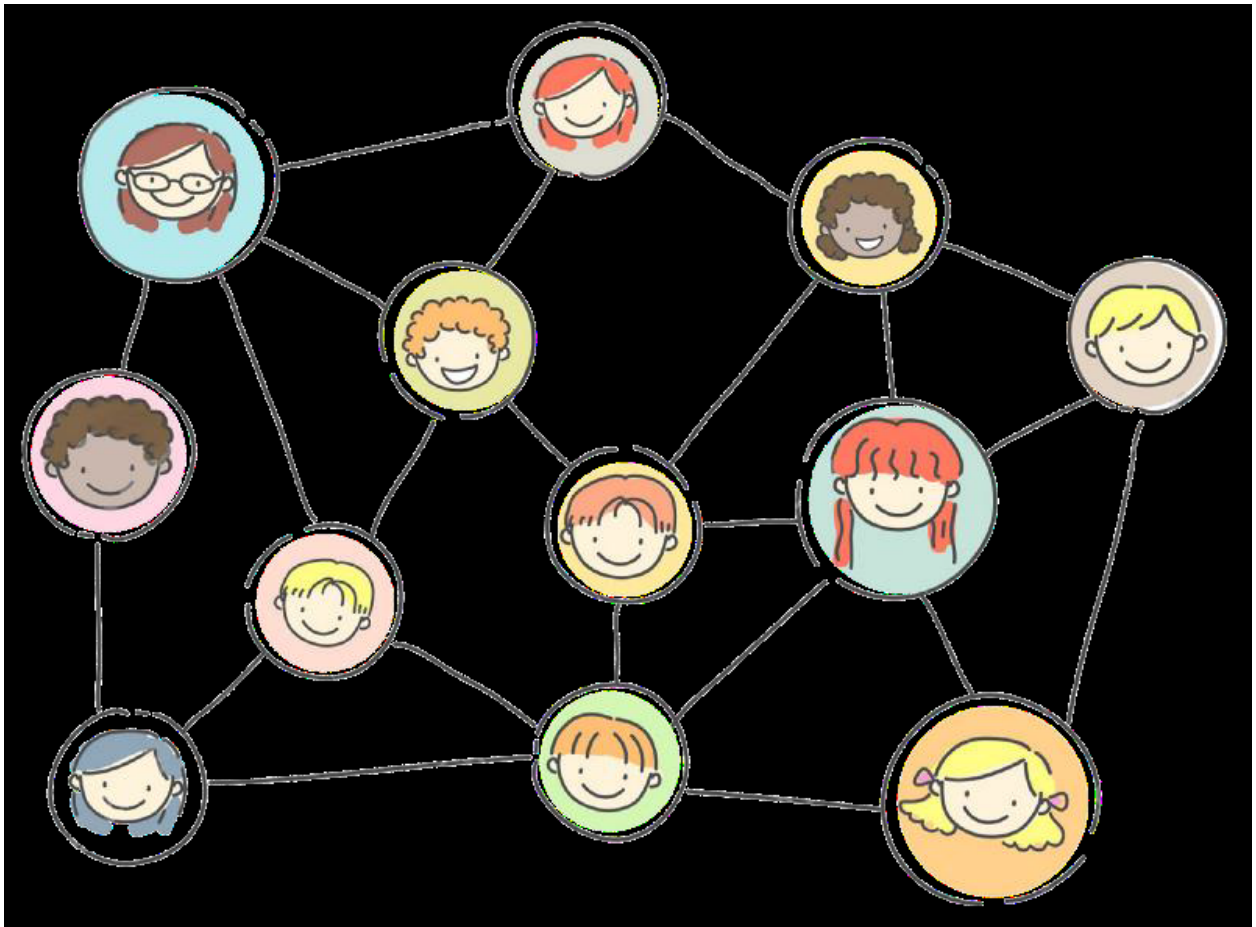
1

BAB I

PENDAHULUAN

1.1 Deskripsi Tugas

Bermain media sosial merupakan aktivitas yang sangat menyenangkan. Media sosial adalah sebuah media daring, dengan para penggunanya bisa dengan mudah berpartisipasi, berbagi, dan menciptakan isi blog, jejaring sosial, wiki, forum dan dunia virtual. Dari berbagai jenis media sosial, jejaring sosial merupakan jenis yang paling diminati oleh orang banyak. Popularitas jejaring sosial seperti facebook, twitter, Instagram, dan lainnya semakin meningkat dalam beberapa tahun terakhir. Pengguna media sosial pun berasal dari berbagai kalangan, mulai dari anak SD hingga orang tua berusia lanjut. Salah satu manfaat dari jejaring sosial adalah pengguna dapat berkomunikasi dengan orang-orang dari berbagai belahan dunia, baik orang yang sudah dikenal, maupun orang yang belum pernah dikenal sebelumnya.



Gambar 1. Ilustrasi graf pertemanan pada social network Facebook

(Sumber: <https://www.freecodecamp.org/news/deep-dive-into-graph-traversals-227a90c6a261/>)

Facebook sebagai salah satu pelopor jejaring sosial sejak tahun 2004 kini telah menembus angka 2,6 miliar pengguna. Fitur friend recommendation menjadi sangat penting karena banyaknya jumlah pengguna Facebook. Fitur bernama "People You May Know" ini dapat memberikan pengguna rekomendasi teman yang sebaiknya di-add, misalnya teman sekolah, teman kuliah, mantan pacar, atau orang yang kita kenal lewat suatu kegiatan tertentu.

Faktor utama saran pertemanan melalui fitur tersebut adalah berdasarkan mutual friend yang dimiliki oleh kedua akun pengguna. Misalnya pengguna A dan C belum berteman di facebook, tetapi keduanya berteman dengan pengguna B, berarti A dan C memiliki mutual friend yang sama, yaitu B. Semakin banyak mutual friend yang dimiliki antarkedua akun, maka semakin tinggi rekomendasi akun tersebut untuk di-add.

Selain itu, di setiap social media, termasuk Facebook, pengguna dapat mengeksplorasi akun-akun pengguna lainnya yang tidak memiliki mutual friends sama sekali. Akan tetapi, dengan menelusuri graf pertemanan antar akun sehingga kita dapat mengetahui 'jarak' antar akun agar bisa saling terhubung dan berteman.

Deskripsi tugas:

Dalam tugas besar ini, Anda akan diminta untuk membangun sebuah aplikasi GUI sederhana yang dapat memodelkan beberapa fitur dari People You May Know dalam jejaring sosial media (Social Network). Dengan memanfaatkan algoritma Breadth First Search (BFS) dan Depth First Search (DFS), Anda dapat menelusuri social network pada akun facebook untuk mendapatkan rekomendasi teman seperti pada fitur People You May Know. Selain untuk mendapatkan rekomendasi teman, Anda juga diminta untuk mengembangkan fitur lain agar dua akun yang belum berteman dan tidak memiliki mutual friends sama sekali bisa berkenalan melalui jalur tertentu.

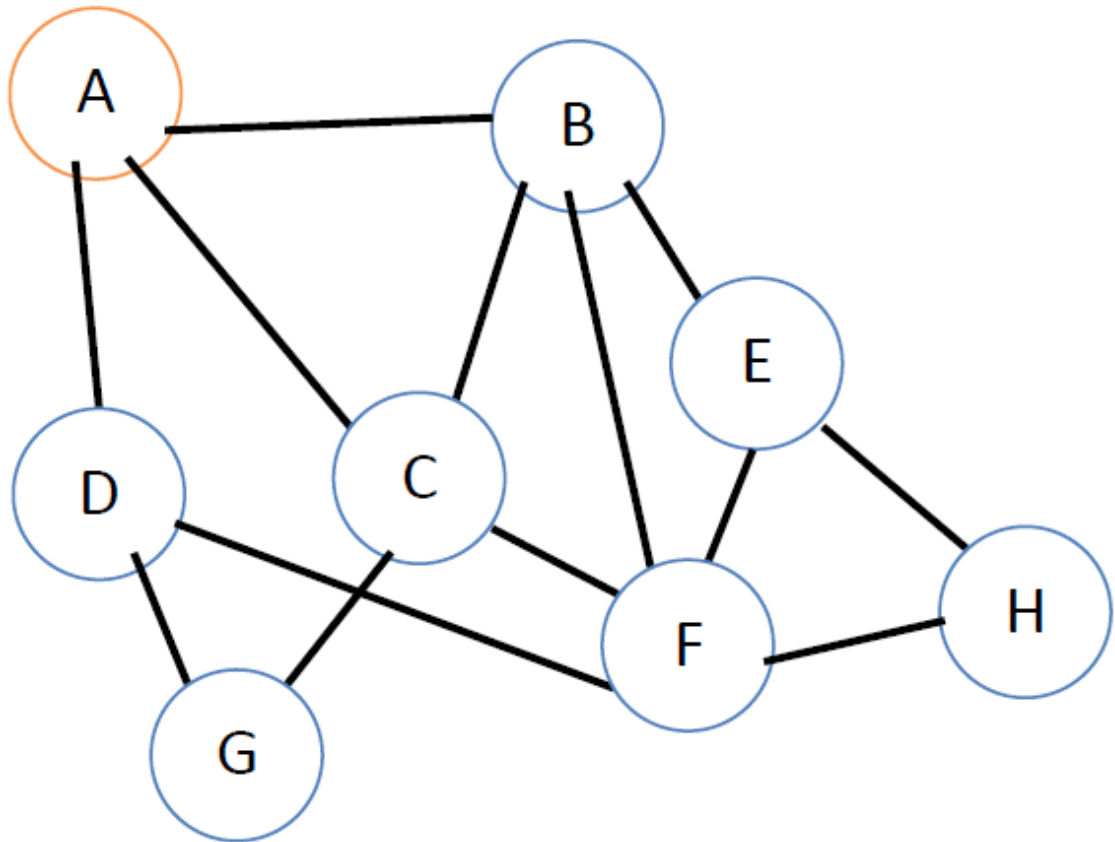
Contoh Input dan Output Program

Contoh berkas file eksternal:

```
13
AB
AC
AD
BC
BE
BF
CF
CG
DG
DF
EH
EF
FH
```

Gambar 2. Contoh Isi File Eksternal

Visualisasi graf pertemanan yang dihasilkan file eksternal:



Gambar 3. Contoh visualisasi graf pertemanan dari file eksternal

Untuk fitur friend recommendation, misalnya pengguna ingin mengetahui daftar rekomendasi teman untuk akun A. Maka output yang diharapkan sebagai berikut:

```
Daftar rekomendasi teman untuk akun A:  
Nama akun: F  
3 mutual friends:  
B  
C  
D  
  
Nama akun: G  
2 mutual friends:  
C  
D  
  
Nama akun: E  
1 mutual friend:  
B
```

Gambar 4. Hasil output yang diharapkan untuk rekomendasi akun A

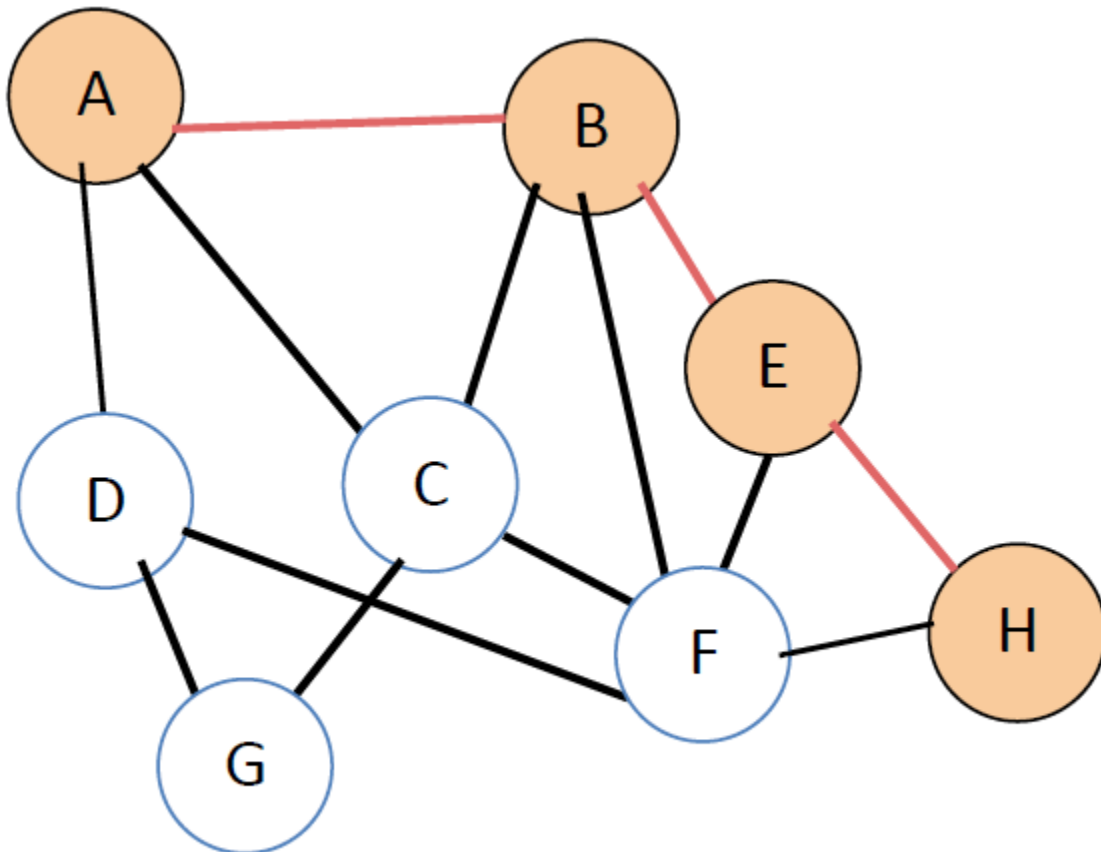
Untuk fitur explore friends, misalnya pengguna ingin mengetahui seberapa jauh jarak antara akun A dan H serta bagaimana jalur agar kedua akun bisa terhubung. Berikut output graf dengan penelusuran BFS yang dihasilkan.

Berikut output yang diharapkan untuk penelusuran menggunakan BFS.

Nama akun: A dan H
2nd-degree connection
A → B → E → H

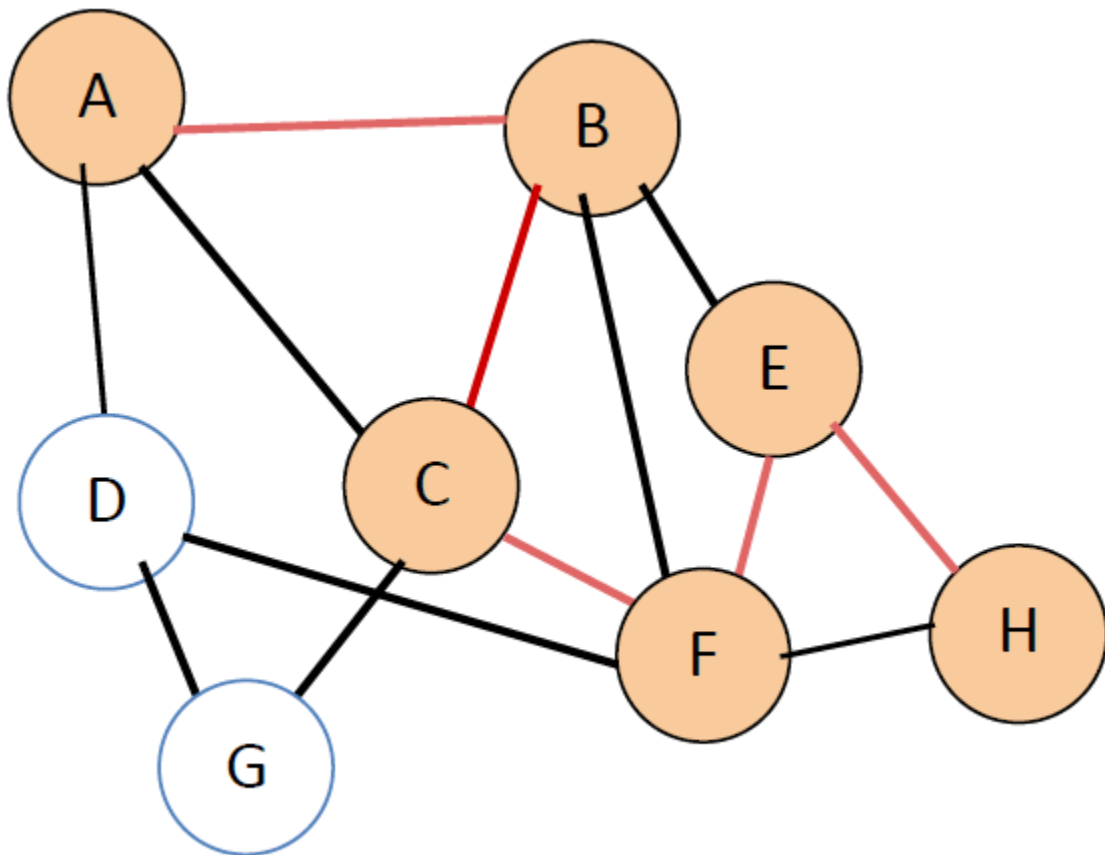
Gambar 5. Hasil output akun Nth degree connection akun A dan H menggunakan BFS

Perhatikan busur antara akun A dan H, terbentuk salah satu jalur koneksi sebagai berikut: A-B-E-H (ada beberapa jalur lainnya, seperti A-D-F-H, dll, urutan simpul untuk ekspan diprioritaskan berdasarkan abjad). Akun A dan H tidak memiliki mutual friend, tetapi kedua akun merupakan 2nd-degree connection karena di antara A dan H ada akun B dan E yang saling berteman. Sehingga akun H dapat terhubung sebagai teman dengan jalur melalui akun B dan akun E. Jalur koneksi dari A ke H menggunakan BFS digambarkan dalam bentuk graf sebagai berikut.



Gambar 6. Hasil visualisasi jalur koneksi menggunakan BFS

Sedangkan untuk penggunaan algoritma DFS, diperoleh jalur lainnya, yaitu A-B-C-F-E-H yang digambarkan dalam bentuk graf sebagai berikut



Gambar 7. Hasil visualisasi jalur koneksi menggunakan DFS

Pada fitur explore friends, apabila terdapat dua buah akun yang tidak bisa saling terhubung (tidak ada jalur koneksi), maka akan ditampilkan bahwa akun tersebut tidak bisa terhubung melalui jalur koneksi yang sudah dimilikinya sekarang sehingga orang tersebut memang benar-benar harus memulai koneksi baru dengan orang tersebut.

Misalnya terdapat dua orang baru, yaitu J dan I yang hanya terhubung antara J-I. Maka jalur koneksi yang dibentuk dari A ke J adalah.

Nama akun: A dan J
Tidak ada jalur koneksi yang tersedia
Anda harus memulai koneksi baru itu sendiri.

Gambar 8. Hasil output tidak ada jalur koneksi antara A dan J

Spesifikasi Program:

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

•

•

•

Gambar 9. Tampilan layout dari aplikasi desktop yang dibangun

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.

Program yang dibuat harus memenuhi spesifikasi wajib sebagai berikut:

- 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 kaskas yang tersedia. Sebagai referensi, salah satu kaskas 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/1XhFSpHU028Gaf7YxkmdbluLkQqVI3MY6gt1t-PL30LA/edit?usp=sharing>
- 4) Terdapat dua fitur utama, yaitu:
 - a. **Fitur friend recommendation:** 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; 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:** 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); Program menerima pilihan dua akun yang belum berteman; Program akan menampilkan nilai N-th degree connection antar kedua akun dan memberikan jalur melalui akun mana saja sampai kedua akun bisa terhubung; 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.; 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.

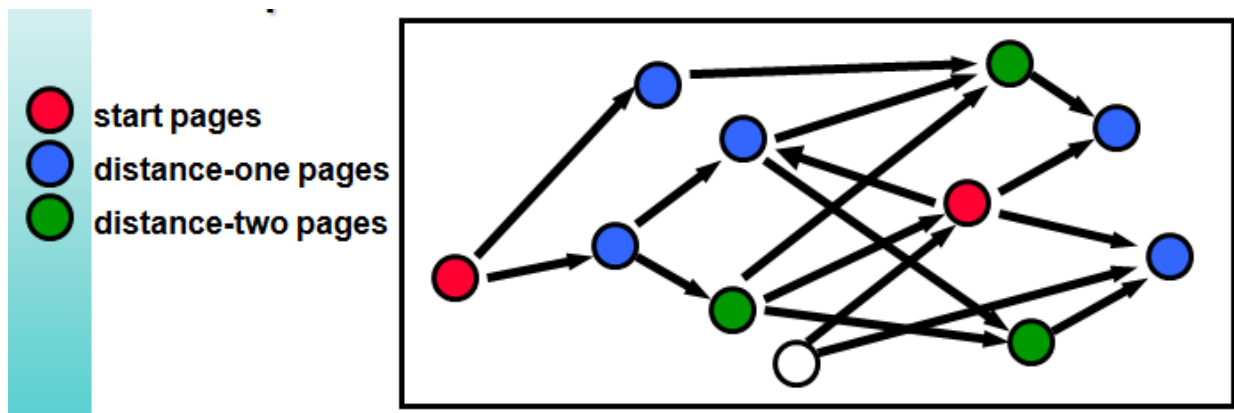
BAB II LANDASAN TEORI

2.1 Graf Traversal

Secara istilah, graf traversal merupakan topik dalam ilmu komputasi yang membahas graf untuk proses kunjungan ke semua simpulnya. Proses kunjungan ini berjalan secara sistematis, baik dengan pencarian melebar/*Breadth-First Search*(BFS) maupun dengan pencarian mendalam/*Depth-First Search*(DFS). Berikut adalah contoh graf traversal:



Gambar 10. Graf Social Network



Gambar 11. Graf Web Page Network

2.2 BFS(Breadth-First Search)

Pencarian melebar/BFS adalah salah satu metode traversal yang dikenal dalam ilmu komputasi. Secara sederhana, algoritmanya adalah sebagai berikut:

1. Mulai dari suatu simpul s
2. Kunjungi semua simpul yang bertetangga dengan simpul s terlebih dahulu
3. Kunjungi simpul yang belum dikunjungi dan bertetangga dengan simpul-simpul yang sebelumnya dikunjungi, demikian seterusnya

2.3 DFS(Depth-First Search)

DFS juga termasuk salah satu metode traversal dalam ilmu komputasi. Secara sederhana, algoritmanya adalah sebagai berikut:

1. Mulai dari suatu simpul s
2. Kunjungi simpul t yang bertetangga dengan simpul s
3. Ulangi DFS untuk simpul t
4. Ketika mencapai simpul u sedemikian sehingga semua simpul yang bertetangga dengannya telah dikunjungi, lakukan *backtrack* ke simpul terakhir yang dikunjungi sebelumnya dan mempunyai simpul t yang belum dikunjungi.
5. Pencarian berakhir apabila tak ada lagi simpul yang dapat dikunjungi dari simpul yang telah dikunjungi

2.4 C# Desktop Application Development

Desktop app ini ditulis dengan bahasa C# (C-Sharp) di dalam aplikasi visual studio, dengan menggunakan windows Form. Windows Form atau sering disebut dengan WinForm adalah Class library buatan microsoft yang bersifat GUI atau Graphical User Interface. Windows Form tergabung dalam Framework .Net. Tujuan diciptakan Win Form adalah untuk mempermudah developer dalam membuat suatu aplikasi berbasis desktop. Banyak sekali fitur yang disediakan oleh winform, seperti label, panel dan beberapa fitur lainnya yang menunjang developer dalam UI/UX aplikasi.

BAB III

ANALISIS PEMECAHAN MASALAH

3.1 Langkah Pemecahan Masalah

Dalam proses mendesain solusi dari permasalahan ini, langkah pertama adalah dengan membagi permasalahan menjadi permasalahan yang lebih kecil. Permasalahan utama dibagi menjadi permasalahan mencari rekomendasi teman terdekat dan permasalahan mencari rekomendasi teman tertentu.

Secara umum langkah pertama penyelesaian masalah adalah dengan Mapping elemen persoalan dengan elemen graf. Pada langkah selanjutnya penyelesaian akan berbeda tergantung dengan permasalahannya. Pada permasalahan mencari rekomendasi teman terdekat, penyelesaian dapat dibagi menjadi 3 yaitu mencari teman, mencari temannya teman, dan mengeliminasi teman dari temannya teman. Pada permasalahan mencari teman tertentu, terdapat dua solusi yang perlu dibuat, yaitu solusi menggunakan algoritma DFS dan solusi menggunakan algoritma BFS. dikarenakan elemen persoalan telah dimapping menjadi elemen graf, maka algoritma BFS dan DFS dapat langsung diterapkan pada graf yang telah terbentuk.

3.2 Mapping Elemen Persoalan

Algoritma DFS dan BFS merupakan algoritma pencarian pada suatu graf. Oleh karena itu elemen yang dimiliki kedua algoritma tersebut adalah simpul dan sisi. Pada persoalan kali ini sebuah simpul merupakan suatu pengguna dari sosial media yang diamati. Sisi yang menghubungkan dua simpul merupakan pertemanan antara pengguna yang diwakilkan kedua simpul tersebut.

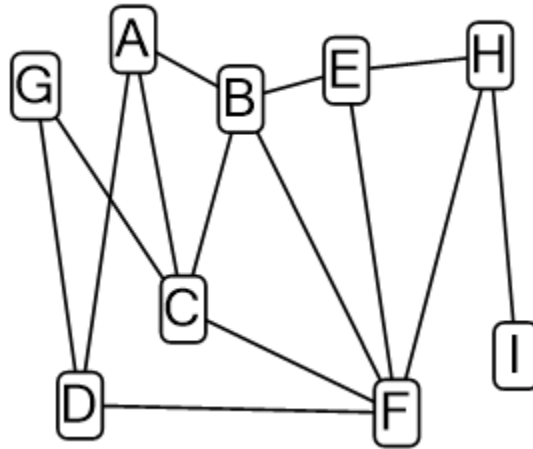
3.3 Contoh Mapping Elemen Persoalan

Misal terdapat 9 pengguna yaitu pengguna A, B, C, D, E, F, G, H, dan I. diantara 9 pengguna tersebut terdapat pertemanan sebagai berikut:

AB
AC
AD
BC
BE
BF
CF
CG
DG
DF
EH
EF

F H
H I

Dalam proses Mapping persoalan akan terbentuk suatu graf yang memuat 9 simpul, dimana setiap simpul mewakili seorang pengguna. Setiap pengguna yang berteman akan memiliki sisi yang mewakili pertemannya. Hasil dari proses mapping tersebut akan membuat graf yang divisualisasikan menjadi,



BAB IV IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Program

A. Rekomendasi Teman

```
FriendRec (Vertex v1) -> Dictionary of (vertex, int)
KAMUS LOKAL
    v: List of Vertex
    vr : Dictionary of (vertex, int)

ALGORITMA
    {untuk semua vertex yang terhubung dengan v1}
    for (Vertex ve in v1.Edges) do
        {tambahkan vertex yang terhubung dengan ve jika belum ada di v}
        v <- v.Union(ve.Edges)
    endfor
    {hapus vertex yang terhubung dengan v1}
    v <- v.Except(v1.Edges)
    {hapus v1 dari v}
    v.Remove(v1)

    for(vertex ve in v) do {untuk setiap vertex di v}
        {tambahkan pasangan nama vertex dan jumlah mutual friend dengan v1}
        vr.Add(ve, MutualFriend(ve, v1).count)
    endfor
    {urutkan berdasarkan jumlah mutual friend}
    vr <- vr.OrderByDescending(x => x.value)

    -> vr
```

B. Cari Teman (BFS)

```
DFS(Vertex v, Vertex v1) -> List of Vertex
KAMUS LOKAL
    s : Stack of (vertex, list of Vertex)
    v2, v3, v4, tempv : list of vertex
    se : (vertex, list of vertex)

ALGORITMA
    s.push((v, v4))
    se <- s.pop()
    while (se[0] != v1) do
        v3.Add(se[0])
        v2 <- se[0].Edges.Except(v3)
        v2.Reverse()
        for (vertex ve in v2)
            tempv <- se[1]
            tempv.Add(se[0])
            s.push(ve, tempv)
        endfor
```

```

        se <- s.pop()
    endwhile
    -> se[1]

```

C. Cari Teman (DFS)

BFS(Vertex v, Vertex v1) -> list of Vertex

KAMUS LOKAL

```

    s : Queue of (Vertex, List of Vertex)
    v2, v3, v4, tempv : List of Vertex
    se : (vertex, list of vertex)

```

ALGORITMA

```

    s.Enqueue((v, v4))
    se <- s.Dequeue()

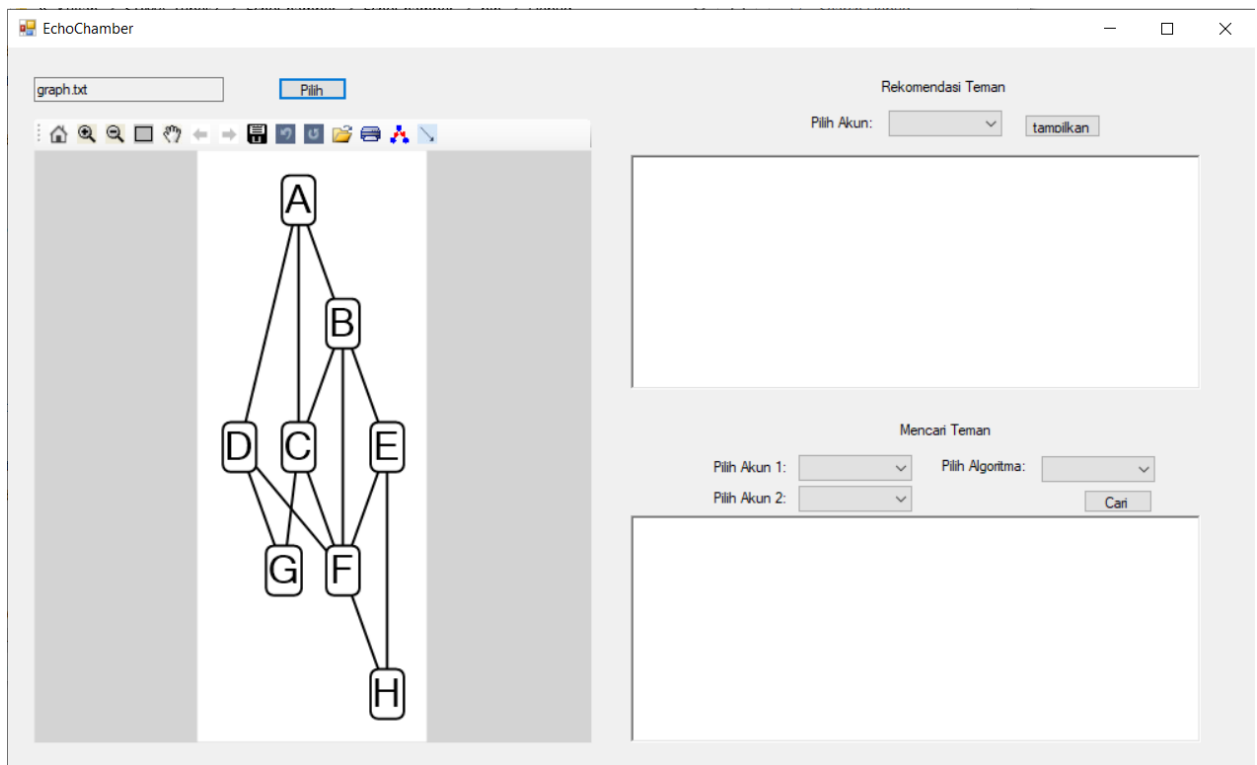
    while(se[0] != v1) do
        v3.Add(se[0])
        v2 <- se[0].Edges.Except(v3)
        for (Vertex ve in v2)
            tempv <- se[1]
            tempv.Add(se[0])
            s.Enqueue((ve, tempv))
        endfor
        se <- s.Dequeue()
    endwhile
    -> se[1]

```

4.2 Struktur Data

Struktur data yang digunakan dalam program adalah graf yang diimplementasikan dalam bentuk adjacency list. Kelas Graph direpresentasikan sebagai list of vertex dan Kelas vertex memiliki atribut nama dan list of vertex yang merepresentasikan edge.

4.3 Cara Penggunaan Program

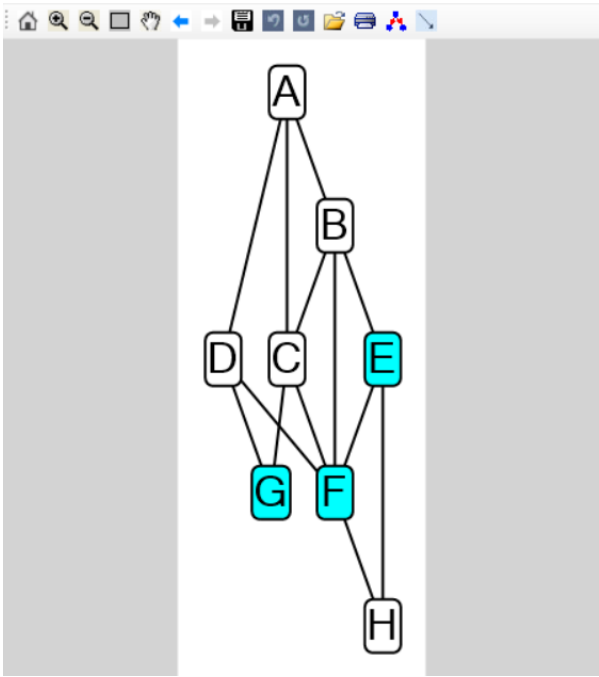
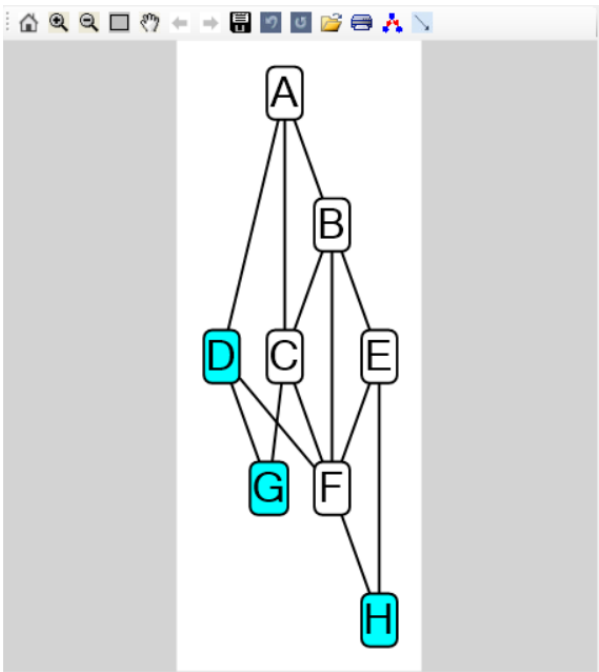


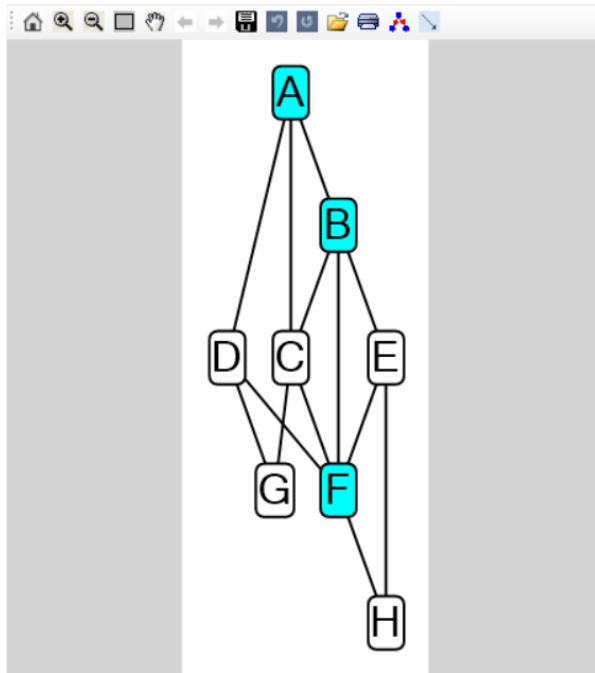
Gambar 12. Interface Program yang Dibuat

Setelah program dijalankan, untuk membaca graf dari file txt tekan tombol pilih. Kemudian pilih file yang akan dibaca pada window file explorer yang terbuka. File txt yang akan dibaca harus sesuai dengan format Gambar 2. Setelah itu visualisasi dari graf akan muncul dalam interface program.

Terdapat dua fitur yang ada dalam program yaitu rekomendasi teman dan mencari teman. Untuk menjalankan fitur rekomendasi teman pilih akun yang akan dicari rekomendasinya lalu tekan tombol tampilkan. Program akan menampilkan akun-akun rekomendasi teman beserta jumlah dan nama akun mutual friend. Akun yang direkomendasikan juga akan berubah warnanya menjadi cyan pada visualisasi graf. Untuk menjalankan fitur mencari teman pilih akun 1 dan juga akun 2. Kemudian pilih algoritma yang akan digunakan yaitu DFS atau BFS lalu tekan cari. Program kemudian akan menampilkan proses pencarian akun teman dengan algoritma yang dipilih, jalur pencarian yang ditemukan, dan juga level koneksi yang terjadi. Jalur pencarian juga akan di visualisasi graf dengan warna ungu.

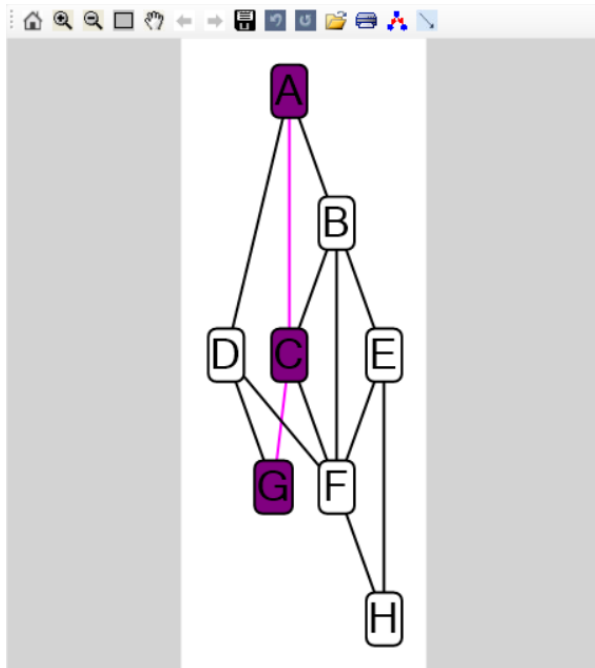
4.4 Hasil Pengujian

Hasil	Analisis
 <p>Rekomendasi teman untuk A</p>	<p>Hasil yang didapat sudah tepat. Rekomendasi teman untuk A adalah sebagai berikut. F dengan 3 mutual friends yaitu B, C, dan D. G dengan 2 mutual friends yaitu C dan D. E dengan 1 mutual friend yaitu B.</p>
 <p>Rekomendasi Teman untuk B</p>	<p>Hasil yang didapat sudah tepat. Rekomendasi teman untuk B adalah sebagai berikut. D dengan 2 mutual friends yaitu A dan F. H dengan 2 mutual friends yaitu E dan F. G dengan 1 mutual friend yaitu C.</p>

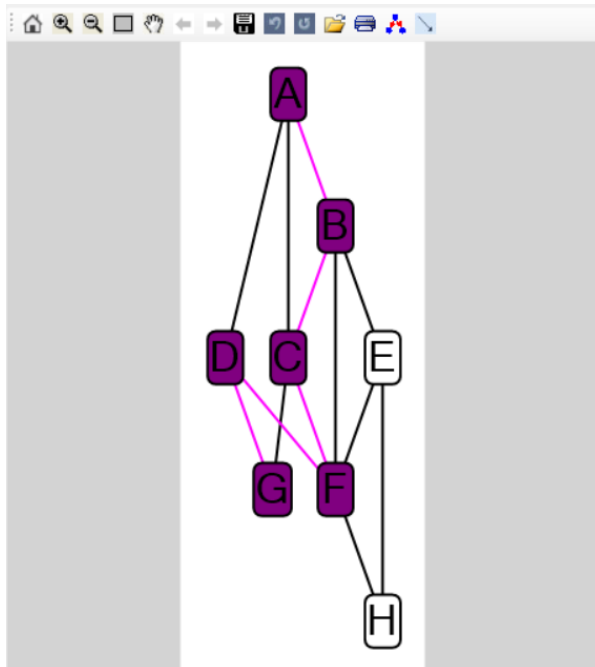


Rekomendasi Teman untuk G

Hasil yang didapat sudah tepat.
Rekomendasi teman untuk G adalah sebagai berikut. A dengan 2 mutual friends yaitu C dan D. F dengan 2 mutual friends yaitu C dan D. B dengan 1 mutual friend yaitu C.

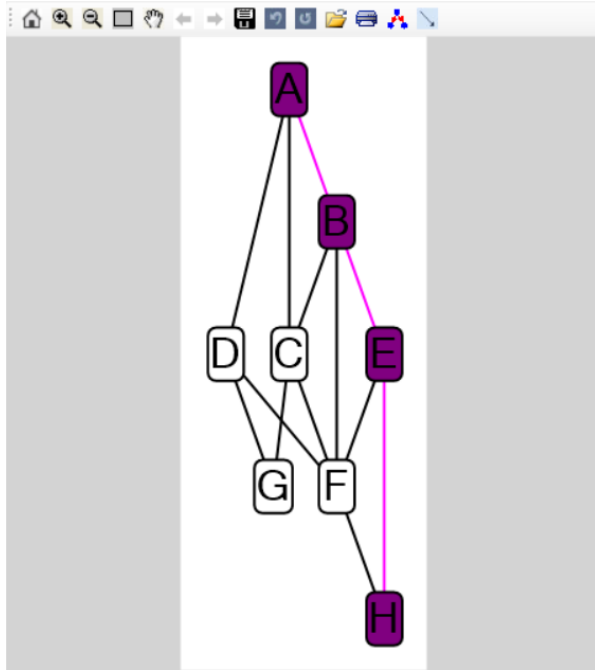


Mencari Teman dari akun A ke G dengan BFS



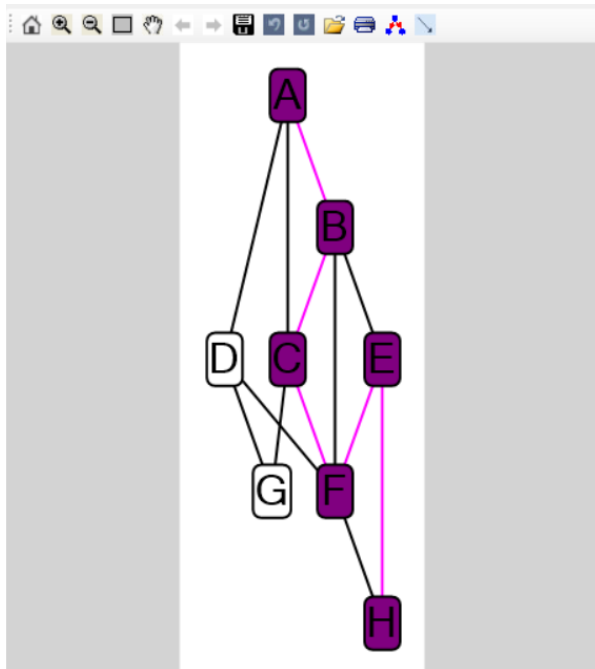
Mencari Teman dari akun A ke G dengan DFS

Pencarian dengan kedua algoritma sudah mendapatkan hasil yang tepat. Pencarian dengan BFS menghasilkan jalur A-C-G (koneksi level 2) sedangkan pencarian dengan DFS menghasilkan jalur A-B-C-F-D-G (koneksi level 5). Pada kasus ini strategi BFS lebih baik untuk digunakan karena menghasilkan jalur yang lebih optimal (level koneksi lebih rendah). Hal ini dapat terjadi karena BFS melakukan pencarian ke samping sehingga simpul yang dicari pasti akan ditemukan dalam tingkat yang paling minimum. Berbeda dengan DFS yang melakukan pencarian ke dalam sehingga besar peluang simpul ditemukan namun dengan tingkat koneksi yang lebih tinggi karena prioritas penelusuran mengikuti alfabet. Namun dalam proses pencarian solusi seringkali algoritma DFS dapat memberikan hasil dengan mengevaluasi jumlah simpul yang lebih sedikit dibandingkan algoritma DFS (walaupun solusi yang dihasilkan belum tentu optimal).

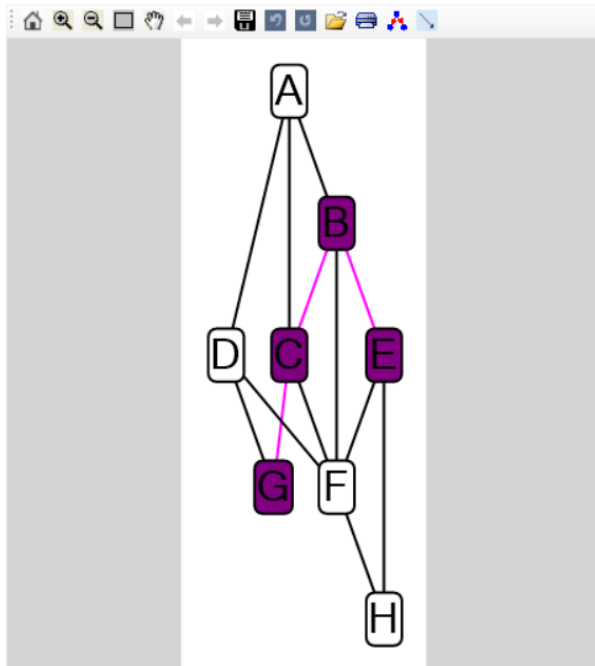


Mencari Teman dari akun A ke H dengan BFS

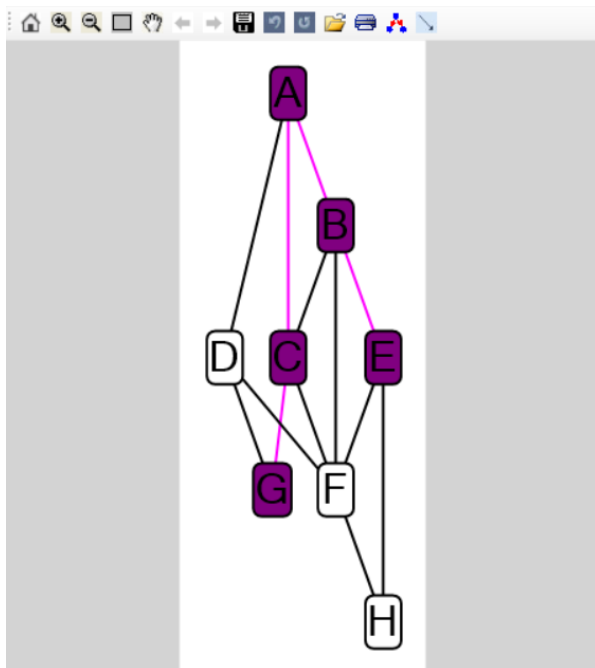
Pencarian dengan kedua algoritma sudah mendapatkan hasil yang tepat. Pencarian dengan BFS menghasilkan jalur A-B-E-H (koneksi level 3) sedangkan pencarian dengan DFS menghasilkan jalur A-B-C-F-E-H (koneksi level 5). Pada kasus ini strategi BFS lebih baik untuk digunakan karena menghasilkan jalur yang lebih optimal (level koneksi lebih rendah). Hal ini dapat terjadi karena BFS melakukan pencarian ke samping sehingga simpul yang dicari pasti akan ditemukan dalam tingkat yang paling minimum. Berbeda dengan DFS yang melakukan pencarian ke dalam sehingga besar peluang simpul ditemukan namun dengan tingkat koneksi yang lebih tinggi karena prioritas penelusuran mengikuti alfabet. Namun dalam proses pencarian solusi seringkali algoritma DFS dapat memberikan hasil dengan mengevaluasi jumlah simpul yang lebih sedikit dibandingkan algoritma DFS (walaupun solusi yang dihasilkan belum tentu optimal).



Mencari Teman dari akun A ke H dengan DFS



Mencari Teman dari akun G ke E dengan BFS



Mencari Teman dari akun G ke E dengan DFS

Pencarian dengan kedua algoritma sudah mendapatkan hasil yang tepat. Pencarian dengan BFS menghasilkan jalur G-C-B-E (koneksi level 3) sedangkan pencarian dengan DFS menghasilkan jalur G-C-A-B-E (koneksi level 4). Pada kasus ini strategi BFS lebih baik untuk digunakan karena menghasilkan jalur yang lebih optimal (level koneksi lebih rendah). Hal ini dapat terjadi karena BFS melakukan pencarian ke samping sehingga simpul yang dicari pasti akan ditemukan dalam tingkat yang paling minimum. Berbeda dengan DFS yang melakukan pencarian ke dalam sehingga besar peluang simpul ditemukan namun dengan tingkat koneksi yang lebih tinggi karena prioritas penelusuran mengikuti alfabet. Namun dalam proses pencarian solusi seringkali algoritma DFS dapat memberikan hasil dengan mengevaluasi jumlah simpul yang lebih sedikit dibandingkan algoritma DFS (walaupun solusi yang dihasilkan belum tentu optimal).

BAB V

SIMPULAN DAN SARAN

5.1. Kesimpulan

Algoritma pencarian graf DFS dan BFS merupakan algoritma yang sangat cocok digunakan untuk mencari hubungan pertemanan pada akun sosial media. Penulis mendapatkan bahwa dalam skala kecil pencarian dengan metode BFS umumnya bersifat lebih panjang, tetapi akan menemukan hasil yang paling maksimum, sedangkan pencarian dengan metode DFS bersifat lebih simpel, tetapi hasil yang ditemukan belum tentu yang terbaik. Pada graf dengan skala besar (seperti sosial media pada umumnya), metode pencarian BFS jauh mengungguli metode pencarian DFS dari segala aspek. Dalam kasus mencari *mutual friend* penulis menemukan metode DFS dan BFS akan menghasilkan hasil yang sama dengan langkah pengerjaan yang memiliki tingkat kompleksitas sama.

Melalui tugas besar 2 Strategi Algoritma ini, penulis juga mendapat kesempatan menggunakan *software* baru untuk *desktop app development* dan juga bahasa pemrograman baru.

5.2. Saran

Penulis merasa jika terdapat algoritma yang bisa memperkirakan relationship degree suatu pertemanan, metode pencarian DFS dapat menjadi lebih cepat/efektif untuk dilakukan dibandingkan dengan metode pencarian BFS. Penulis merasa topik penelitian tersebut cukup berharga untuk diteliti lebih jauh.

Daftar Pustaka

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag1.pdf>

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag2.pdf>

<http://csharp.net-informations.com/>