TUGAS 1 PRAKTIKUM

STABLE MATCHING PROBLEM

Disusun sebagai salah satu tugas

mata kuliah Analisis Algoritma



Muhammad Raihan Akbar

140810160013

PROGRAM STUDI S-1 TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

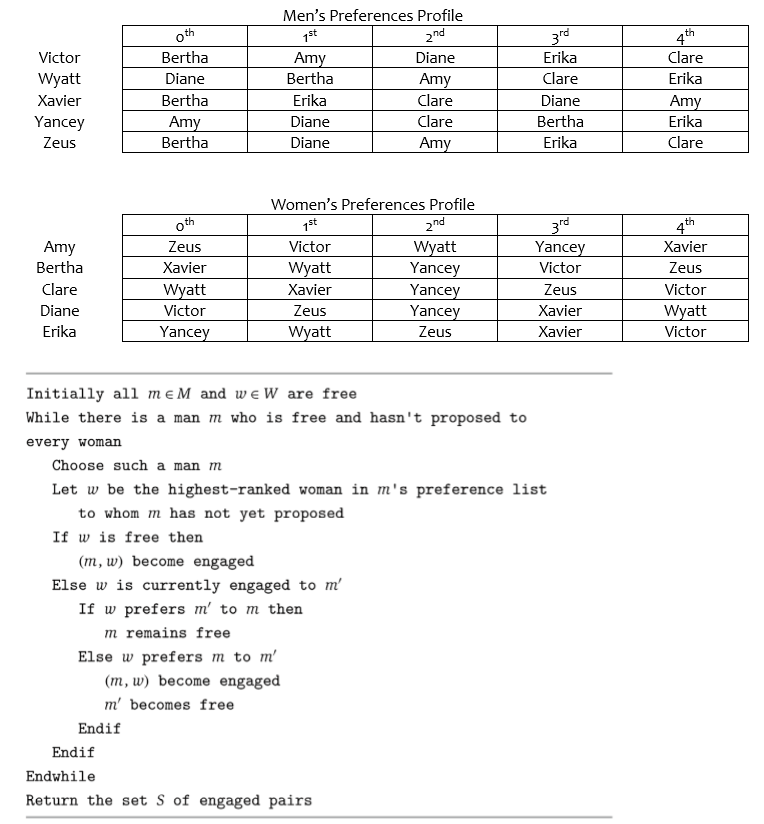
UNIVERSITAS PADJADJARAN

2019

Tugas Praktikum

Muhammad Raihan Akbar 140810160013

# Algoritam Gale-Shapley



Kode dalam bahasa C++

/\* Muhammad Raihan Akbar

Analgo - Worksheet 1 - Praktikum

Stable Marriage Problem

5-3-2019 F\*cking fresh from my head

===============================\*/

#include <iostream>

using namespace std;

int findIndex(string name){

int resultIndex;

if (name == "Victor" || name == "Amy")

resultIndex = 0;

else if (name == "Wyatt" || name == "Bertha")

resultIndex = 1;

else if (name == "Xavier" || name == "Clare")

resultIndex = 2;

else if (name == "Yancey" || name == "Diane")

resultIndex = 3;

else

resultIndex = 4;

return resultIndex;

}

int main()

{

string men[5][2] = {

{"Victor", "free"},

{"Wyatt", "free"},

{"Xavier", "free"},

{"Yancey", "free"},

{"Zeus", "free"},

};

string menPreference[5][5] = {

{"Bertha", "Amy", "Diane", "Erika", "Clare" }, // Victor

{"Diane", "Bertha", "Amy", "Clare", "Erika" }, // Wyatt

{"Bertha", "Erika", "Clare", "Diane", "Amy" }, // Xavier

{"Amy", "Diane", "Clare", "Bertha", "Erika" }, // Yancey

{"Bertha", "Diane", "Amy", "Erika", "Clare" } // Zeus

};

string women[5][2] = {

{"Amy", "free"},

{"Bertha", "free"},

{"Clare", "free"},

{"Diane", "free"},

{"Erika", "free"},

};

string womenPreference[5][5] = {

{"Zeus", "Victor", "Wyatt", "Yancey", "Xavier" }, // Amy

{"Xavier", "Wyatt", "Yancey", "Victor", "Zeus" }, // Bertha

{"Wyatt", "Xavier", "Yancey", "Zeus", "Victor" }, // Clare

{"Victor", "Zeus", "Yancey", "Xavier", "Wyatt" }, // Diane

{"Yancey", "Wyatt", "Zeus", "Xavier", "Victor" } // Erika

};

string engaged[5][2] = {

{ "Amy", "?"},

{ "Bertha", "?"},

{ "Clare", "?"},

{ "Diane", "?"},

{ "Erika", "?"}

};

cout<<"Storyline in getting true couple..." <<endl<<endl;

bool isSomeoneFree = true;

int tries = 0;

for (int i=0; i<5; i++){

for (int j=0; j<5; j++){

// Jika cowo itu jomblo

if (men[i][1] == "free"){

// Dan jika cewe yang pengen ditembak jomblo juga

if (women[ findIndex(menPreference[i][j]) ][1] == "free" ){

// Maka mereka jadian

men[i][1] = "engaged";

women[ findIndex(menPreference[i][j]) ][1] = "engaged";

engaged[ findIndex(menPreference[i][j]) ][1] = men[i][0];

cout<<engaged[ findIndex(menPreference[i][j]) ][1] <<" with "

<<engaged[ findIndex(menPreference[i][j]) ][0]

<<" engaged!" <<endl;

isSomeoneFree = false;

// Jangan lirik yang lain2

break;

}

// Namun jika cewe itu udah jadian

else {

// Dan ternyata si cowo itu lebih didamba

// daripada pasangannya sekarang..

for (int k=0; k<5; k++){

// Selama si cowo yang nembak lebih didamba daripada pasangannya sekarang...

if ( womenPreference[ findIndex(menPreference[i][j]) ][k] ==

engaged[ findIndex(menPreference[i][j]) ][1] )

break;

if ( womenPreference[ findIndex(menPreference[i][j]) ][k] == men[i][0]

&& k < 5 ){

// Pasangan itu putus...

cout<<engaged[ findIndex(menPreference[i][j]) ][1] <<" with "

<<engaged[ findIndex(menPreference[i][j]) ][0]

<<" break apart! then ";

men[ findIndex(engaged[ findIndex(menPreference[i][j]) ][1]) ][1] = "free";

// Lalu jadian sama cowo yang baru...

men[i][1] = "engaged";

women[ findIndex(menPreference[i][j]) ][1] = "engaged";

engaged[ findIndex(menPreference[i][j]) ][1] = men[i][0];

cout<<engaged[ findIndex(menPreference[i][j]) ][1] <<" with "

<<engaged[ findIndex(menPreference[i][j]) ][0]

<<" engaged!" <<endl;

isSomeoneFree = true;

}

}

}

}

else

break;

}

// Maksimal kesempatan TIAP cowo nembak 5 kali.

tries++; // This condition needs further evaluation

// Kalo masih ada yang jomblo...

if (isSomeoneFree == true && i == 5 - 1 )

i = -1; // Restart

}

// Lalu diakhir cerita, tampilkan pasangan2 itu

if (isSomeoneFree == false){

cout<<endl <<endl

<<"Stable couple are : "<<endl<<endl;

for(int i=0; i<5; i++){

cout<<engaged[i][1] <<" engaged with "

<<engaged[i][0] <<endl;

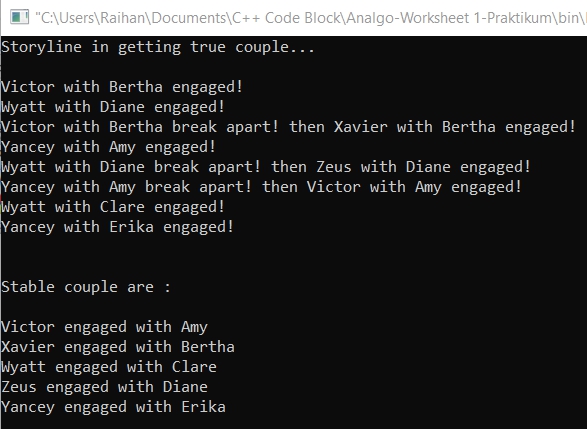
}

}

return 0;

}

Screenshot kode saat running :



# Pembuktian Algoritam G-S

Anda diminta untuk membuktikan algoritma G-S benar dengan menjawab pertanyaan berikut:

1. **Fakta (1.1):** Seorang wanita tetap bertunangan dari titik di mana dia menerima proposal pertamanya; dan urutan mitra yang bertunangan dengannya menjadi lebih baik dan lebih baik lagi (hal ini sesuai dengan daftar preferensi wanita). **->** tidak perlu dipertanyakan

Benar. Wanita langsung bertunangan dari titik dimana dia menerima proposal pertamanya.

1. **Fakta (1.2):** Urutan wanita yang dilamar pria lebih buruk dan lebih buruk lagi (hal ini sesuai dengan daftar preferensi pria). **->** tidak perlu dipertanyakan

Benar. Daftar preferensi pilihan tiap pria sudah berurut secara *descending*.

1. **Teorema (1.3):** Algoritma G-S berakhir setelah paling banyak n2 iterasi menggunakan While Loop. Buktikan!

Paling banyak n2 iterasi didapat dari hasil perkalian antara jumlah pria dan wanita yang ada didalam himpunan. Aturan yang menyebutkan setiap pria akan mengajukan proposal ke, maksimal, seluruh wanita yang ada memberikan pencerahan bahwa jumlah maksimal proposal yang diajukan adalah jumlah pria dikali jumlah wanita.

1. **Teorema (1.4):** Jika seorang pria bebas di beberapa titik dalam eksekusi algoritma, maka ada seorang wanita yang belum dia ajak bertunangan. Buktikan!

Dapat dibuktikan dengan kontradiksi. Misal ada waktu tertentu dalam pelaksanaan algoritma ketika seorang pria single, namun telah mengusulkan kepada setiap wanita. Ini berarti saat ini, setiap wanita telah dipasangkan setidaknya satu kali. Ada n wanita yang bertunangan dan ada n pria yang bertunangan, yang berarti bahwa tidak mungkin ada orang yang masih single jika setiap n sudah berpasangan.

1. **Teorema (1.5):** Himpunan S yang dikembalikan saat terminasi adalah perfect matching Buktikan!

Pria pasti hanya akan melamar apabila belum berpasangan atau pasangan sebelumnya tidak cocok. Sedangkan wanita akan selalu memilih pria dengan preferensi teratas untuk bertunangan dengannya. Baik pria dan wanita, prinsip mereka adalah melamar sesuai urut preferensi. Dengan itu Himpunan S adalah perfect matching dikarenakan teori tersebut.

1. **Teorema (1.6):** Sebuah eksekusi algoritma G-S mengembalikan satu set pasangan S. Set S adalah pasangan yang stabil. Buktikan!

Tidak ada pria yang bisa ditolak oleh semua wanita. Wanita pun harus mendapatkan satu pria dan ia hanya dapat menolak lamaran ketika ada pria yang lebih tinggi preferensinya dibandingkan pria sebelumnya. Setiap iterasi dari loop sementara melibatkan tepat satu proposal dan pria tidak akan melamar wanita yang sama dua kali.