**KAUNO TECHNOLOGIJOS UNIVERSITETAS**

**Diskrečiosios Struktūros**

Kursinio darbo ataskaita

Atliko: Tautvydas Petkus, IF-1/9

Vadovai: Radvilė Krušinskienė

**KAUNAS, 2012**

Turinys

[1. Uždavinio sąlyga ir jo analizė 3](#_Toc341642738)

[1.1. Uždavinio sąlyga 3](#_Toc341642739)

[1.2. Uždavinio analizė 3](#_Toc341642740)

[Dalinis grafas 3](#_Toc341642741)

[Dvipusis grafas 3](#_Toc341642742)

[Jungusis grafas 3](#_Toc341642743)

[Bendra užduotis 4](#_Toc341642744)

[2. Algoritmo aprašymas, iliustruotas pavyzdžiais 4](#_Toc341642745)

[2.1. Sprendimo algoritmas 4](#_Toc341642746)

[3. Programos kodas 4](#_Toc341642747)

[3.1. main.m 4](#_Toc341642748)

[AtlikimoFunkcija.m: 5](#_Toc341642749)

[1. Testai 7](#_Toc341642750)

[1.1. Pirmas pavyzdys: 7](#_Toc341642751)

[1.2. Antras pavyzdys: 8](#_Toc341642752)

[Išvados 8](#_Toc341642753)

[Literatūra: 8](#_Toc341642754)

# Uždavinio sąlyga ir jo analizė

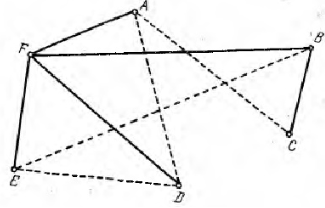
## Uždavinio sąlyga

37. Sudaryti algoritmą ir programą, kuri rastų visus dalinius dvipusius jungius grafus.

## Uždavinio analizė

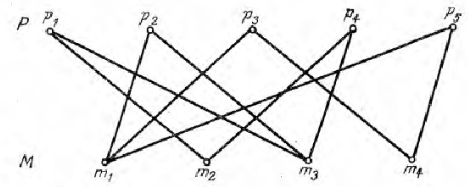
Užduotis yra surasti visus įmanomus bet kokio duoto grafo dalinius dvipusius jungius grafus.

Dalinis grafas – tai grafas, turintis visas orginalaus grafo Viršūnes, tačiau ne visas jo briaunas. Dalinis grafas ir pradinis grafas skiriasi tik briaunų skaičiumi, niekuo daugiau.



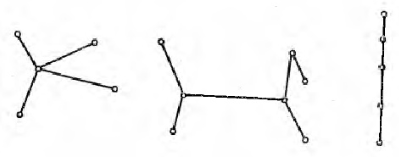
Aukščiau pateiktas dalinio grafo pavyzdys. Punktyrinės linijos – briaunos, kurių dalinis grafas neturi.

Dvipusis grafas – tai grafas, kurio viršūnės yra suskirstytos į dvi aibes. Šių dvejų aibių viršūnės aibės viduje tarpusavy nėra sujungtos jokiomis briaunomis, tačiau sujungtos briaunomis tarp dvejų aibių.



Aukščiau pateikta dvi aibės – P ir M. Jų viršūnės nėra sujungtos: p1 nesijungia su jokia kita savo aibės viršūne, p2 su jokia ir t.t. Tas pats ir su M aibe. Viršūnės gali jungtis tiktais su kitos aibės viršūnėmis.

Jungusis grafas – tai grafas, kuris sudaro vientisą grafą. Kitais žodžiais formuluojant, šį grafą galima nubrėžt ranka neatitraukus jos, brėžiant net ir kelis kartus per briaunas.



Aukščiau pavaizduotas grafas nėra jungusis, nes jį sudaro 3 atskiri pografiai, nesujungti tarpusavyje. Kad grafas būtų jungus, reikia sujungti visus pografius bent 1 briauna.

Bendra užduotis – transformuoti esamą grafą taip, jog iš jo padarytume jungųjį grafą. Taip galima bus padaryti atmetus nepageidautinas briaunas, kurios neleidžia grafui būti dviskilčiui. Taip mes gautume jungųjį dalinį grafą. Ir, galiausiai atmetus briaunas reikia įsitikinti, jog mūsų grafas yra vientisas ir jungus, jog atmetus briaunas ir padarius grafą dvidaliu, nepadalinom jo į 2 atskirus pografius.

# Algoritmo aprašymas, iliustruotas pavyzdžiais

## Sprendimo algoritmas

Pirmiausiai, mums reikia suskirstyti duotąjį grafą į visas įmanomas aibes. Pradedant nuo vienetinių aibių, ir baigiant ties viršūnių sumos puse, nes tada viršūnių aibės kartosis. Rekursiškai ieškoma visų galimų aibių. Svarbiausia, jog aibėse viršūnės nesikartotų, ir jog jos nesikartotų su ankstesnėmis aibėmis. Jų sąlyga yra ta, jog kiekviena aibės viršūnė bus eile didesnė negu ankstesnioji aibė. Pvz. {1 2}, {1 5}, {2 4 5} {4 5 6} ir t.t. Suradus visas aibes, mes turim ir likusią aibę, kurioje yra viršūnės, nesančios porinėje jų aibėje. Pvz. {2 3} ir. {1 4 5 6 7}. Taip padarydami, mes gaunam iš dalies dviskiltį grafą, kurio viršūnės yra suskirstytos į dvi aibes. Dabar reikia aibes sujungti galimomis pradinio grafo briaunomis, tačiau nejungiant viršūnės aibės viduje. Pvz., tikrinti, ar iš aibių {2 3} ir {1 4 5 6 7} viršūnė nr. 2 jungi su 1, 4, 5, 6, 7. Tą patį padaryti ir su viršūne Nr. 3. Sujungę aibes briaunomis, mes gauname jau dviskiltį dalinį grafą. Dabar reikia įsitikinti, ar grafas yra jungus. Pagrindinė jungumo sąlyga yra ta, jog briaunų skaičius turi būti ne mažesnis, nei viršūnių skaičius – 1. Praėjus šią sąlygą, reikia atmesti visus galimus grafus, kuriuose viršūnės laipsnis gali būti lygus 0. Grafas su viršūnėmis, kurių laipsniai 0, negali būti jungūs. Praėjus šį testą, darome jungiųjų komponenčių testą. Pasirenkame pirmąją viršūnę, ir su ja sjungiame kitas viršūnes. Darmome tai tol, kol nelieka viršūnių , kurias galime jungti. Tai atliekame kelis kartus, jei jungiųjų komponenčių yra daugiau. Jeigu jungiųjų komponenčių yra daugiau, tai grafas nėra jungusis, ir mūsų uždaviniui šis variantas atkernta. Belieka tik tie pografiai, kurie atitinka mūsų sąlygą.

Pseudokodas:

Ciklo ilgis = Virsuniu skaicius / 2

IF Ciklo ilgis > 1

FOR Dabartinis ciklas = 1 : Ciklo ilgis

IF Dabartinis ciklas = 1

FOR j = 1 : Virsuniu skaicius

Virsunes{} += j

IF Dabartinis Ciklas <> 1

FOR j1 = 1 : Virsuniu Skaicius

FOR j = 1 : Dabartinis Ciklas

FOR j2 = 1 : Virsuniu Skaicius

IF j2 > j1

Virsunes{} += j1 + j2

FOR Dabartine Aibe = 1 : length(Virsunes{})

FOR Dabartine Virsune = 1 : length(Virsunes{Dabartine Aibe})

FOR Orginali Virsune = 1 : Virsuniu Skaicius

FOR Dabartine Briauna = 1 : Briaunu Skaicius

IF Dabartine Virsune != Orginali Virsune &&   
 Dabartine Virsune = DabartineBriauna(1)

&& OrginaliBriauna = DabartineBriauna(2)

THEN Dalines Briaunos{} += Dabartine Briauna

FOR Dabartinis Poaibis = 1 : length(Poaibiai{})

IF length(Briaunos{Dabartinis Poaibis}) >= Virsuniu Skaicius – 1

FOR Dabartine Virsune = 1 : Poaibiai{DabartinisPoaibis}

FOR Dabartine Briauna = 1 : DalinesBriaunos{ DabartinisPoaibis}

IF Dabartine Virsune = Dabartine Briauna

Poaibio Laipsniai(Dabartine Virsune) ++

FOR Laipsnis = 1 : Laipsniai

IF DOES NOT CONTAIN 0

FOR Dabartinis Poaibis[1]

JungiuSk = 1

IF Komponentes1 !=

Komponentes 2

JungiuSk++;

IF JungiuSk = 1

Rezultatu Aibe += DalinesBriaunos{Dabartinis Poaibis}

# Programos kodas

## main.m

clc, close all, clear all ;

V = importdata('Virsunes.txt');

U = {};

T = importdata('Briaunos.txt');

for i=1:length(T),

U{i} = [T(i, 1) T(i, 2)];

end

[rezU] = AtlikimoFunkcija(V, U);

for i=1:length(rezU), %Isvedami musu rezultatai i konsoles langa

AA = rezU{i};

i

for j=1:length(AA),

AA1 = cell2mat(AA(j))

end

end

hold on; axis equal; axis([-1.1,1.1,-1.1,1.1]); grid on

arc=0; poz=0; Fontsize=10; lstor=1; spalva='b';

figure(1)

title('Duotasis grafas')

plotGraphVU1(V,U,0,0,[],1,10,3,'g');

title('Gautas grafas')

plotGraphVU1(V,rezU{1},0,0,[],1,10,3,'r'); %isvedam pati pirma rezultata grafiskai

## AtlikimoFunkcija.m:

function [rezU] = AtlikimoFunkcija( V, U)

rezU = {};

ilgis = length(V);

ilgisCiklui = round(ilgis/2) - mod(ilgis, 2);

%Ciklo esme tokia: Sudarysime visas imanomas aibes. Taciau ju reikia

%Tik per pus maziau naudoti cikla, nes virsunes tada kartosis

%Naudojantis visu galimybiu formule, kai reiksmes negali kartotis,

%Apskaiciuosime, kiek kartu kiekviena cikla prasukus reikes skaiciuoti

%Pirmosios puses aibes. Is pradziu bus vienetine aibe, po to is

%dveju elementu ir t.t.

KompleksinisKelias = 0;

for i=1:ilgisCiklui,

KompleksinisKelias = KompleksinisKelias + RadimasC(i, ilgis);

end

%VK = []; %Nepanaudota funkcija

%if ilgisCiklui> 1

% for i=1:ilgisCiklui,

% for j=1:i,

% kelias = 0;

% for r=1:(ilgis-i+j),

% for t=1:RadimasC(i - j , ilgis - r),

% kelias = kelias + 1;

% poz = PozicijaC(i, ilgis, kelias, j, r);

% VK(poz , j) = V(r);

% end

% end

% end

% end

%end

%Sukiriam visus galimus poaibius

Poaibiai = {};

if ilgisCiklui >1,

for i=1:ilgisCiklui,

if (i == 1),

for i=1:ilgis,

Vir = V(i);

Poaibiai{length(Poaibiai) + 1} = Vir;

end

end

if (i == 2),

for j=1:(ilgis - i + 1),

for j2 = j:ilgis,

if (j2 > j),

Vir(1) = V(j);

Vir(2) = V(j2);

Poaibiai{length(Poaibiai) + 1} = Vir;

end

end

end

end

if (i == 3),

for j=1:(ilgis - i + 1),

for j1= (j + 1):(ilgis - i + 2),

for j2 = j1:ilgis,

if (j2 > j1 && j1 > j),

Vir(1) = V(j);

Vir(2) = V(j1);

Vir(3) = V(j2);

Poaibiai{length(Poaibiai) + 1} = Vir;

end

end

end

end

end

end

end

%Sujungiam poaibius su krastinem

Krastines = {};

for i=1:length(Poaibiai),

Virs1 = Poaibiai(i);

PoaibioIlgis = length(Poaibiai(i));

for j=1:length(Virs1),

Krast1 = {};

for j2=1:length(V),

reiksme1 = cell2mat(Virs1(j));

for t=1:length(reiksme1),

lygin = reiksme1(t);

reiksme2 = V(j2);

if (reiksme1 ~= reiksme2),

for j3=1:length(U),

AA = U(j3);

AA1 = cell2mat(AA(1));

if (AA1(1) == lygin && AA1(2) == reiksme2),

Krast1(length(Krast1)+1) = AA;

end

if (AA1(2) == lygin && AA1(1) == reiksme2),

Krast1(length(Krast1)+1) = AA;

end

end

end

end

end

end

Krastines{length(Krastines) + 1}= Krast1;

end

%Jungumas

for i=1:length(Poaibiai),

Keliai = (Krastines(i));

TikriKeliai = Keliai{1};

if (length(TikriKeliai) >= (ilgis - 1)),

Laipsniai = [];

Laipsniai(length(V)) = 0;

for j=1:length(TikriKeliai), %Suzinome visu virsuniu laipsnius

AA = TikriKeliai(j);

AA1 = cell2mat(AA(1));

for j2=1:length(V),

if ((AA1(1) == V(j2)) || (AA1(2) == V(j2))),

Laip = Laipsniai(j2);

Laip = Laip + 1;

Laipsniai(j2) = Laip;

end

end

end

patikrinimas = 0;

for j=1:ilgis,

if (Laipsniai(j) ~= 0),

patikrinimas = patikrinimas + 1;

end

end

if (patikrinimas == ilgis), %eliminuojami pografiai su 0 laipsniais

Jungios = 0;

Komponentes = zeros(1, ilgis);

while any(Komponentes == 0)

NJung = V(Komponentes == 0);

Pirmas = NJung(1);

Jungios = Jungios + 1;

Komponentes(Pirmas) = Jungios;

leidimas = true;

while leidimas

JungiosiosB = Komponentes;

for rr = 1:length(TikriKeliai),

AA = TikriKeliai(rr);

AA1 = cell2mat(AA(1));

if (Komponentes(AA1(1)) == Jungios)

Komponentes(AA1(2)) = Jungios;

end

if (Komponentes(AA1(2)) == Jungios)

Komponentes(AA1(1)) = Jungios;

end

end

leidimas = any(JungiosiosB ~= Komponentes);

end

end

if (Jungios == 1),

rezU{length(rezU) + 1} = TikriKeliai;

end

end

end

end

end

# Testai

Visų testų rezultatai yra išvedami per komandos langą. Norėdami pavaizduoti juos, reikia main.m faile pakeisti pografio numerį (Pagal nutylėjimą nustatytas yra 1as).

## Pirmas pavyzdys:

Viršūnių aibė:

V = [1 2 3 4 5 6];

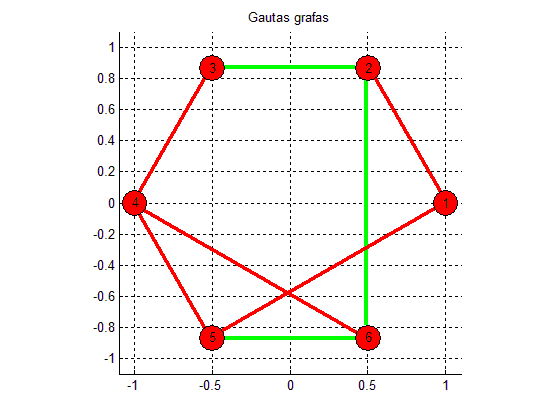
Briaunų aibė:

U = {[1 2], [1 5], [2 3], [2 6], [3 4], [4 5], [4 6], [5 6]};

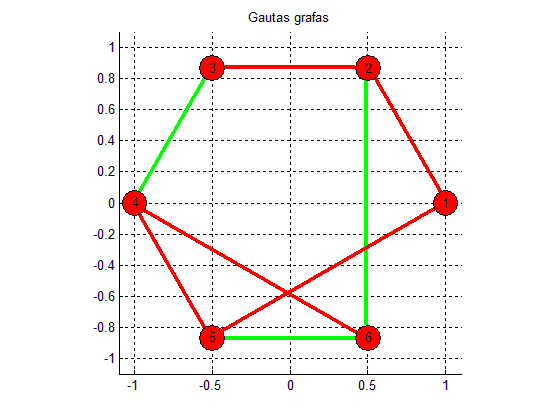
Gautas Grafas ir mūsų apskaičiuotas rezultatas (žalios briaunos kartu su raudonom – pradinis grafas, tik raudonos – gautas pografis):

Iš viso gauta 14 pografiai:

1as pografis:



13as pografis:



## Antras pavyzdys:

Viršūnių aibė:

V = [1 2 3 4 5];

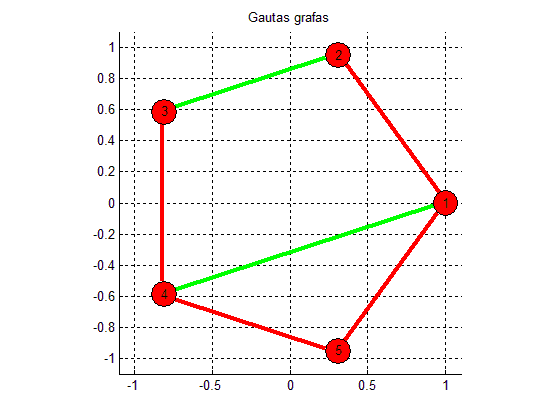
Briaunų aibė:

U = {[1 2], [1 4], [1 5], [2 3], [3 4], [4 5]};

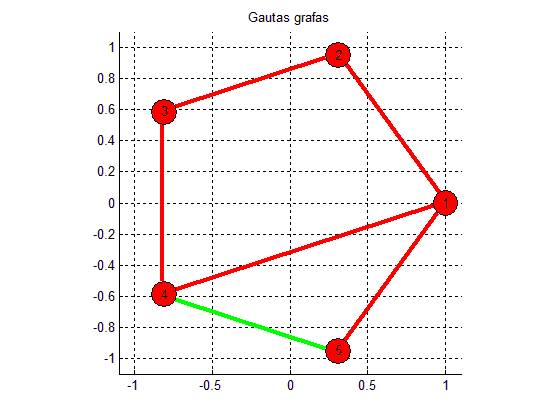
Gautas Grafas ir mūsų apskaičiuotas rezultatas (žalios briaunos kartu su raudonom – pradinis grafas, tik raudonos – gautas pografis):

Iš viso gauta 5 pografiai:

2as pografis:



1as pografis:



# Išvados

Programa veikia. Gavome algoritmo analizę, ir programa duoda mums reikiamus rezultatus.

# 6. Literatūra:

O. Ore „Grafai ir jų pritaikymas“ - 1973 „Mintis“, Vilnius