Zadatak #7

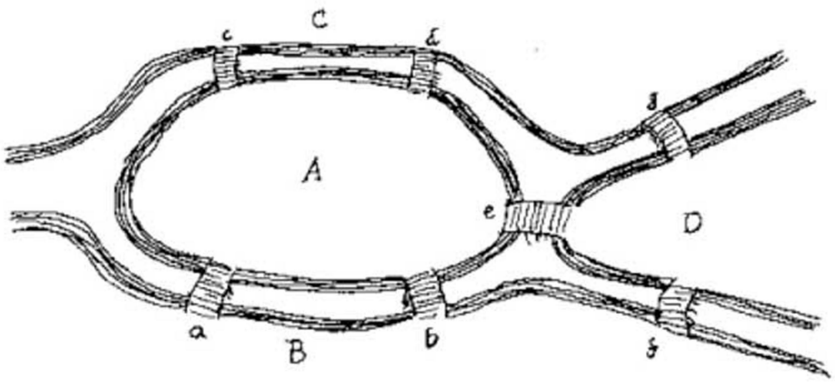
Istorija i primena teorije grafova

Uvod:

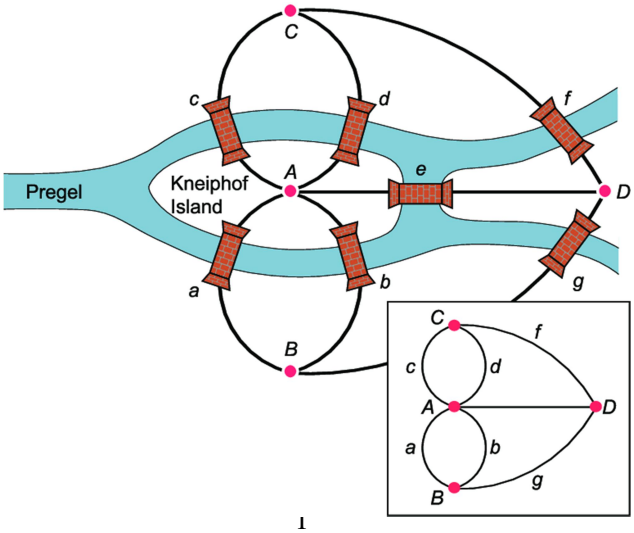
- Osnov za teoriju grafova postavio je 1736. švajcarski matematičar Leonard Ojler (1707-1783)
- Grafovi su važan matematički model za analizu problema u svakodnevnom životu i nalaze primenu u računarstvu, medicini, transportu, internet mreži i povezivanju, genetici itd..

Mostovi Kenigsberga:

- Königsberg tj. Kenigsberg je današnji Kalinjingrad u Rusiji. Kroz njega protiče reka Pregolja I deli ga na 4 dela. Stanovnici Kenigsberga su godinama pokušavali da nađu put koji prelazi preko svih 7 izgrađenih mostova tako da se nijedan ne ponovi dvaput. Ojler je bio zaintrigiran problemom i zaključio da je to do tada nepoznata grana matematike.
- Iako čvor i grane tada nisu bili definisani pojmovi, Ojler je uspeo da sastavi sledeću sliku:



- Gde su 4 dela grada označena sa {A, B, C, D} a mostovi sa {a, b, c, d, e, f, g}
- Delovi grada predstavljaju čvorove a mostovi grane, a to nam ova slika I pokazuje:



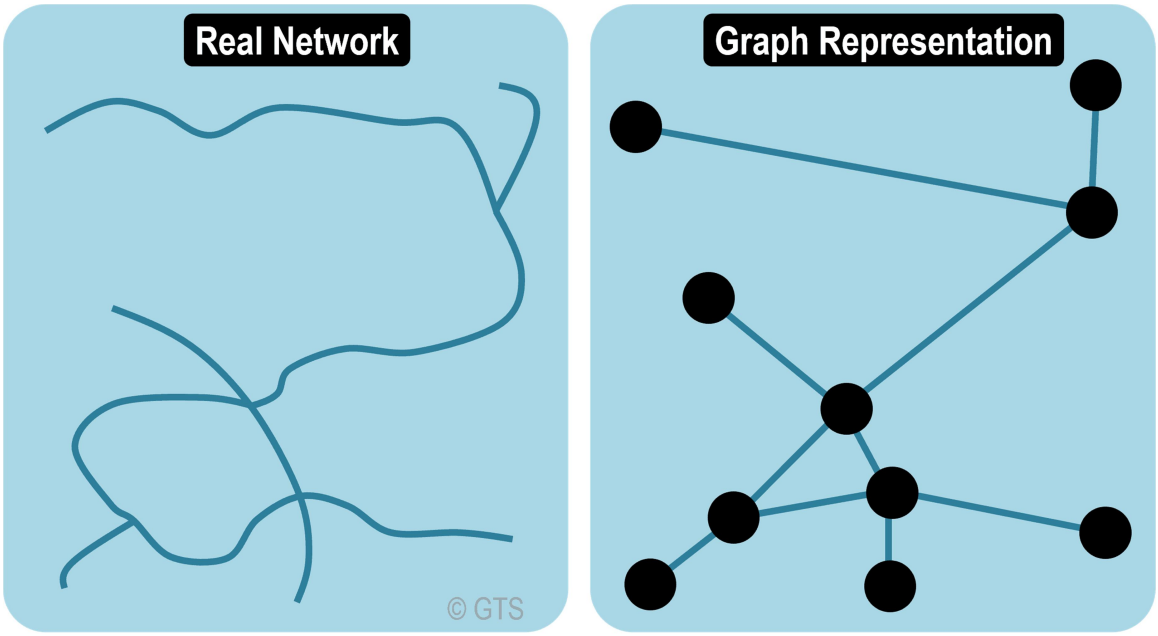
- Ojler je dokazao da ne postoji rešenje koje zadovoljava uslove.
- Osim na početku I na kraju šetnje, svaki put se mora ući I izaći sa jednog čvora preko dva različita mosta, što znači da broj mostova koji se spajaju u svakom čvoru mora biti paran.

Dalji razvoj:

- Nakon tog otkrića, ceo vek nije bilo napretka u polju grafova. Onda, 1878. britanac James Sylvester uvodi reč graf i problem „Sedam mostova Kenigsberga“ postaje popularan kod matematičara. Arthor Kayley izučava teoriju grafova i dolazi do važnih otkrića i rezultata. Kejlijeva formula nam govori o broju stabala na osnovu određenog broja čvorova.
- Još neki istorijski problemi u kojima su grafovi bili od pomoći su
    - Brojanje prostručnih stabala
    - Hipoteza o četiri boje
    - Teorema Kuratovskog
    - Problem trgovačkog putnika
    - Turanova teorema

Primena u drugim oblastima:

- U medicini moćemo koristiti grafove da bismo pokazali putanju širenja zaraznih bolesti i da bismo pronašli način da njih suzbijemo. Takođe, modelovanje molekularnih mreža, metaboloških puteva i genetskih regulacionih mreža koriste grafove za analizu bioloških procesa
- U transportnim mrežama, čvorovi predstavljaju lokacije, a grane puteve ili rute. Grafovi pomaću u planiranju najkraćih puteva, optimizaciji isporuke itd..



- U računarstvu grafovi predstavljaju topologiju računarskih mreža (kao što su LAN ili WAN). Čvorovi predstavljaju uređaje, dok su ivice veze između njih. Analizom grafova se optimizuje protok podataka i otkriva slabosti mreže.

Definicija usmerenog, neusmerenog i prostog grafa

Usmereni graf

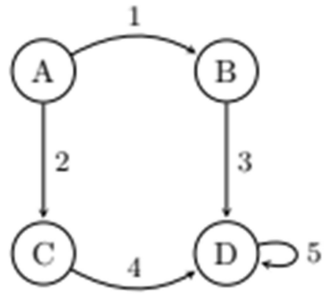
Usmeren graf je uredjeni par G = (V, E), gde je:

- V konacan skup cvorova (ili vrhova)

- $E \subseteq V \times V$ skup usmerenih grana (uredjenih parova cvorova)

Svaka grana (u, v) ima pocetni i krajnji cvor, gde $u \neq v$ osim u slucaju petlje.

Primer usmerenog grafa



Neusmeren graf

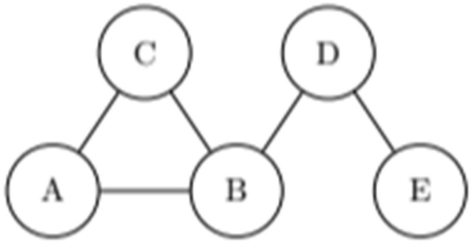
Neusmeren graf je uredjeni par $G = (V, E)$, gde je:

- V konacan skup cvorova (ili vrhova)

- $E \subseteq \{\{u, v\} \mid u, v \in V,\ u \neq v\}$ skup neusmerenih grana (neuredjenih parova cvorova)

Svaka grana $\{u, v\}$ povezuje cvorove u i v, pri cemu je $\{u, v\} = \{v, u\}$

Primer neusmerenog grafa



Definicija prostog grafa

Prost graf je neusmeren graf $G = (V, E)$,, gde je:

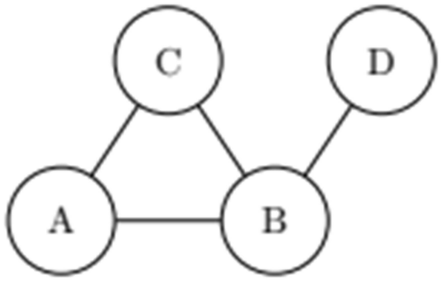
- V konacan skup cvorova (ili vrhova)

- $E \subseteq \{\{u, v\} \mid u, v \in V,\ u \neq v\}$ skup neuredjenih parova cvorova

- Svaka grana povezuje dva razlicita cvora $(u \neq v)$ - Ne postoje paralelene grane izmedju istih parova cvorova;

Prost graf nema petlje niti paralelne grane.

Primer prostog grafa



Definicija jednakosti i izomorfizma grafova

Jednakost grafova:

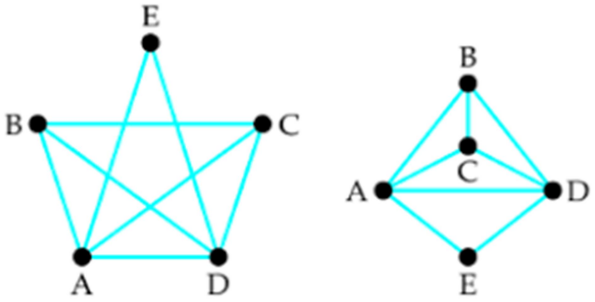
Def. Dva grafa su jednaka ako i samo ako imaju identične skupove čvorova i grana.

$$G_1 = G_2 \Leftrightarrow V_1 = V_2 \wedge E_1 = E_2$$

$$G_i - graf$$

$$V_i - skup\ čvorova\ grafa\ G_i$$

$$E_i - skup\ grana\ grafa\ G_i$$



Izomorfni grafovi:

Def. Dva grafa su izomorfna ako i samo ako postoji bijekcija između njihovih čvorova.

$$G_1 \cong G_2 \Leftrightarrow \exists f: V_1 \leftrightarrow V_2 \wedge (uv \in E_1 \Leftrightarrow f(u)f(v) \in E_2)$$

$$G_i - graf$$

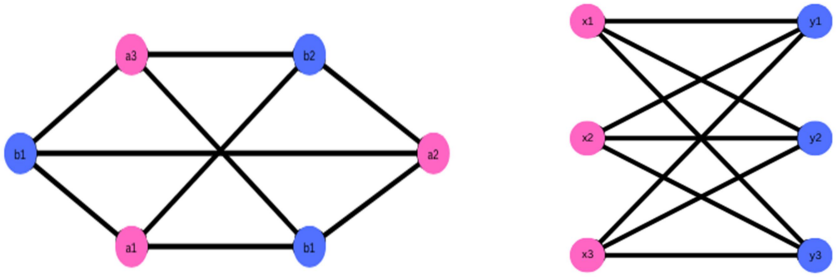
$$V_i - skup\ čvorova\ grafa\ G_i$$

$$E_i - skup\ grana\ grafa\ G_i$$

Potrebni uslovi izomorfizma su:

- jednak broj čvorova
- jednak broj grana
- očuvan stepen čvorova

Primer: Posmatrajmo ova dva grafa



- Oba grafa imaju 6 čvorova, 9 grana i 3-regularni su
- Takođe, po načinu na koji su obojeni čvorovi, možemo reći da su grafovi bipartitni
- Na osnovu toga, možemo reći da su grafovi izomorfni

Definišimo bijektivno preslikavanje:

$$f: \{a_i \to x_i \ \ \wedge \ b_i \to y_i\}$$

Uslovi za izomorfizam : Definicije i primeri

Neophodni uslovi da dva grafa budu izomorfna:

Izomorfizam grafova podrazumeva postojanje bijekcije između skupova čvorova dva grafa tako da se očuva povezanost čvorova. Da bi dva grafa bila izomorfna, moraju ispunjavati sledeće neophodne uslove:

1. Jednak broj čvorova: Oba grafa moraju imati isti broj čvorova.

2. Jednak broj grana: Oba grafa moraju imati isti broj grana.

3. Multiskup stepena čvorova: Za svaki čvor u jednom grafu mora postojati čvor u drugom grafu sa istim stepenom. Drugim rečima, multiskup stepena svih čvorova mora biti identičan.

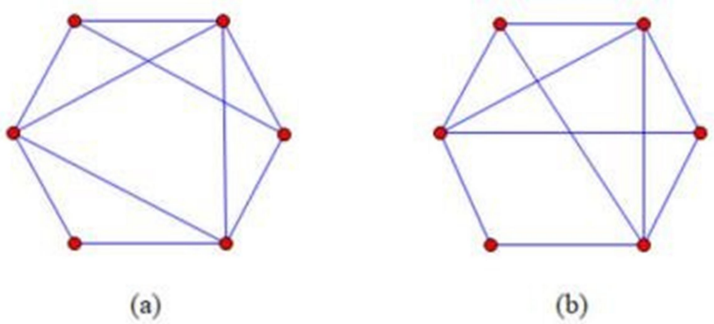
4. Povezanost: Oba grafa moraju biti povezana ili nepovezana na isti način.

Ovi uslovi su **neophodni**, ali i ako ih grafovi ispunjavaju, to još ne garantuje izomorfizam. Na primer, dva grafa mogu imati isti broj čvorova, grana, stepena čvorova i ciklusa, ali ipak ne biti izomorfni.

Da bi se dokazao izomorfizam, potrebni su efikasniji algoritmi poput:

- Algoritma za testiranje grafova na izomorfizam, poput Babai-Luks algoritma (koji radi u polinomski ograničenom vremenu za specifične vrste grafova).
- Praktičnih heuristika, poput označavanja čvorova, upoređivanja matrica incidencije ili adijacencije, ili korišćenja specijalizovanih softverskih alata (npr. nauty, Traces).

Primer dva neizomorfna grafa koji zadovoljavaju uslove:



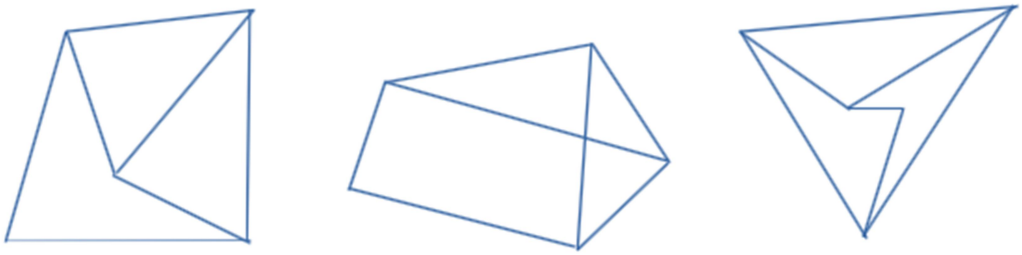
- U grafu (a), način na koji su čvorovi povezani (tj. raspored ivica) stvara drugačiju globalnu strukturu nego u grafu (b).
- Različite topološke karakteristike (planarnost) i strukturalne osobine (način povezivanja čvorova) pokazuju da ovi grafovi nisu izomorfni, čak i ako imaju isti stepen čvorova.

Dokaz izomorfnosti i ekvivalencije

Pojam izomorfnosti

Izomorfizam grafova se definiše kao relacija koja preslikava skup čvorova i grana jednog grafa u skup čvorova i grana drugog, tako da održava osobinu susjednosti čvorova. To znači da, ako su dva čvora jednog grafa povezana granom, onda su granom povezani i odgovarajući čvorovi drugog grafa. Ako zamislimo čvorove i grane jednog grafa u prostoru, pomjeranjem čvorova i deformisanjem odgovarajućih grana, možemo ga dovesti do poklapanja sa njegovim izomorfnim oblikom. Važno je da pri ovoj transformaciji ne narušimo susjednost čvorova.

Težina provjere izomorfnosti grafova je u tome što jedan graf može imati veoma veliki broj međusobno izomorfnih oblika. Kako se u primjenama grafovi često sastoje od velikog broja čvorova i grana, problem postaje još teži. Priroda složenosti problema ilustrovana je slikom 1, sa tri međusobno izomorfna grafa. Čak ni u ovom jednostavnom slučaju, nije potpuno očigledno da se radi o istom grafu.



Slika 1.
Međusobno izomorfni grafovi

Dokaz izomorfnosti grafova sa slike 1:

Da bismo dokazali da su grafovi sa slike međusobno izomorfni, primenjujemo osnovne kriterijume za izomorfizam grafova:

1. Broj čvorova:

Neka je |V|| broj čvorova u svakom grafu.

Posmatrajući sva 3 grafa sa slike, uočavamo da svaki ima isti broj čvorova:

$$|V_1| = |V_2| = |V_3| = 5$$

2. Broj grana:

Neka je |E| broj grana u svakom grafu.

Posmatrajući sva 3 grafa sa slike, uočavamo da svaki ima isti broj grana:

$$|E_1|=|E_2| = |E_3|= 6$$

3. Stepenska sekvenca:

Stepenska sekvenca je niz stepena svih čvorova sortiranih po veličini.

Posmatrajmo svaki graf pojedinačno:
Svaki graf ima:

- jedan čvor sa stepenom 4,
- i četiri čvora sa stepenom 2.

Dakle, stepenska sekvenca za sve grafove je:

$$(4,2,2,2,2)$$

Pošto je stepenska sekvenca identična, struktura povezanosti je očuvana.

4. Očuvanje susjednosti čvorova (bijektivno preslikavanje):

Možemo definisati preslikavanje između čvorova prvog grafa u čvorove drugog i trećeg grafa tako da se susjedstvo čvorova očuva:
Ako je u ~ v u jednom grafu, tada je f(u) ~ f(v) u drugom grafu, gdje f predstavlja bijekciju između skupova čvorova.

Na osnovu slike:

- Veze između čvorova mogu se drugačije crtati, ali očuvanje susjednosti i povezanosti potvrđuje izomorfizam.

Zaključak:

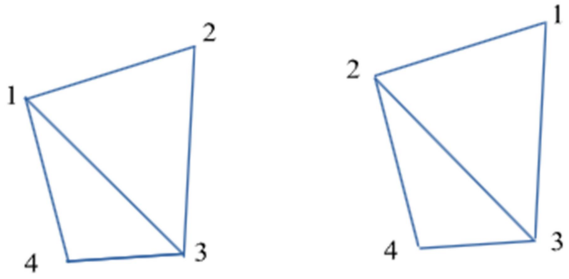
S obzirom na to da:

1. Grafovi imaju isti broj čvorova,
2. Imaju isti broj grana,
3. Imaju identične stepenske sekvence,
4. Postoji preslikavanje koje čuva susjednost čvorova,

dokazujemo da su grafovi prikazani na Slici 1 međusobno izomorfni.

U cilju objašnjenja osnovne ideje kojom se vodi algoritam, podsjetimo se jedne klase grafova, označenih grafova, koju karakteriše specifična, znatno uprošćena provjera izomorfnosti. Dva označena grafa se smatraju izomorfnim samo ako se oznake svakog čvora i grane jednog, tačno poklapaju sa oznakama drugog grafa. Sa tako definisanom relacijom, dva grafa prikazana na slici 2 bi bila neizomorfna, iako se očigledno radi o grafovima identične morfologije.

Istina, ovako definisana relacija izomorfnosti izuzetno pojednostavljuje rad sa grafovima, ali istovremeno, klasa problema koji se mogu tretirati ovom vrstom grafova je znatno umanjena, kao i njena praktična primjenljivost. Najbolje bi bilo kada bi smo mogli ispitivati izomorfnost grafova sa onom lakoćom koja se sreće kod označenih grafova, a da u isto vrijeme ne budemo ograničeni samo na tu klasu grafova.



Slika 2.

Neizomorfni označeni grafovi

1. Metode za dokazivanje izomorfnosti grafova

Prilikom dokazivanja izomorfnosti grafova, koriste se sljedeće metode:

- Poređenje broja čvorova i grana: Ako dva grafa nemaju isti broj čvorova ili isti broj grana, oni ne mogu biti izomorfni.
- Stepeni čvorova: Stepen svakog čvora (broj grana koje izlaze iz čvora) mora biti jednak u oba grafa.
- Poređenje susjednih čvorova: Posmatra se struktura susjednih čvorova i njihove veze.
- Potreba za algoritmom: Kada je graf složen, koriste se algoritmi poput:
    - Brute-force pristupa: Razmatraju se sva moguća preslikavanja čvorova.
    - Algoritmi optimizacije: Algoritmi poput VF2, Ullmannovog algoritma i Nauty.

2. Praktični značaj izomorfizma grafova

Izomorfizam grafova ima široku primjenu u realnim problemima, uključujući:

- Hemiju: Identifikacija strukturalno identičnih molekula.
- Računarske nauke: Optimizacija mreža, prepoznavanje obrazaca i analiza podataka.
- Dizajn elektronskih kola: Poređenje različitih struktura elektronskih šema.

3. Algoritamska složenost problema

- Problem izomorfizma grafova pripada klasi NP problema, što znači da trenutno ne postoji poznat algoritam koji ga rješava u polinomsko vrijeme za sve grafove.
- S obzirom na složenost, naučnici razvijaju specijalizovane algoritme za određene klase grafova (npr. drveća, ciklusi, planarni grafovi) gdje se izomorfizam može dokazati efikasnije.

Pojam ekvivalencije

Ekvivalencija kod grafova je važan pojam u teoriji grafova koji se koristi za grupisanje čvorova prema određenim pravilima. Jedan od načina na koji se ovo koristi jeste u definisanju povezanih komponenti neusmjerenog grafa.

U neusmjerenom grafu, dva čvora su povezana ako postoji put između njih. Ova povezanost se može posmatrati kao relacija ekvivalencije. Relacija ekvivalencije je matematički koncept koji zadovoljava tri osobine: refleksivnost, simetričnost i tranzitivnost.

Refleksivnost znači da je svaki čvor povezan sa samim sobom. U grafu, to znači da postoji trivijalan put od jednog čvora do njega samog. Ovaj put ima dužinu nula, jer ne zahtijeva prelazak preko drugih čvorova.

Simetričnost podrazumijeva da ako je čvor u povezan sa čvorom v, onda je i v povezan sa u. U grafu, ako postoji grana između dva čvora, ona može biti posmatrana u oba smjera, jer je graf neusmjeren.

Tranzitivnost označava da, ako je čvor u povezan sa čvorom v, a v sa w, onda je u povezan sa w. U grafu, to znači da se putevi između čvorova mogu kombinovati kako bi se formirao novi put.

Kada ova relacija ekvivalencije bude definisana, čvorovi se grupišu u klase ekvivalencije. Svaka klasa ekvivalencije predstavlja skup čvorova koji su međusobno povezani. Ove grupe čvorova nazivaju se povezane komponente grafa. Povezane komponente mogu se predstaviti kao indukovani podgrafi. Indukovani podgraf je podgraf koji uključuje sve čvorove iz jedne klase ekvivalencije, zajedno sa svim granama koje ih povezuju. Na ovaj način, povezane komponente pomažu u razumijevanju strukture grafa i njegovih dijelova koji su međusobno povezani.

Primjer 1. Za primer ekvivalencije kod grafova, možemo uzeti jednostavan neusmjereni graf sa četiri čvora A, B, C, i D, gdje su grane sljedeće:

- *A je povezan sa B,*
- B je povezan sa C,
- C je povezan sa D.

A – B – C – D

Dokaz ekvivalencije:

Definišemo relaciju dostizanja R(u, v) između čvorova u i v. Pokazujemo da je ova relacija ekvivalencija:

1. Refleksivnost: Svaki čvor je dostupan samom sebi. Na primjer, R(A,A), R(B,B), R(C,C) i R(D,D) su tačne.

2. Simetričnost: Ako je u povezan sa v, onda je v povezan sa u. Na primjer, ako R(A,B) važi (postoji grana između njih), onda važi i R(B,A).

3. Tranzitivnost: Ako je u povezan sa v, a v sa w, onda je u povezan sa w. Na primjer, R(A,B) i R(B,C) impliciraju R(A,C).

Zaključak:

Relacija dostizanja dijeli čvorove u jednu povezanu komponentu, jer su svi čvorovi međusobno povezani putem. Dakle, cijela mreža čini jednu klasu ekvivalencije.

Algoritam za ispitivanje izomorfnosti

```
using System;
using System.Collections.Generic;

public class Graph{
    private int[,] adjMatrix;
    private int numNodes;

    public Graph(int numNodes){
        this.numNodes = numNodes;
        this.adjMatrix = new int[numNodes, numNodes];
    }

    public void AddEdge(int u, int v){
        adjMatrix[u, v] = 1;
        adjMatrix[v, u] = 1;
    }

    public bool IsIsomorphic(Graph other){
        if (numNodes != other.numNodes){
            return false;
        }
        var permutations = GeneratePermutations(numNodes);

        foreach (var perm in permutations){
            if (AreMatricesEquivalent(perm, other.adjMatrix))
                return true;
        }
        return false;
    }

    private List<int[]> GeneratePermutations(int n){
        var result = new List<int[]>();
        Permute(new List<int>(), new HashSet<int>(), n, result);
        return result;
    }
```

```
    private void Permute(List<int> current, HashSet<int> used, int n, List<int[]> result){
        if (current.Count == n){
            result.Add(current.ToArray());
            return;
        }

        for (int i = 0; i < n; i++){
            if (!used.Contains(i)){
                current.Add(i);
                used.Add(i);
                Permute(current, used, n, result);
                current.RemoveAt(current.Count - 1);
                used.Remove(i);
            }
        }
    }

    private bool AreMatricesEquivalent(int[] perm, int[,] otherMatrix){
        for (int i = 0; i < numNodes; i++){
            for (int j = 0; j < numNodes; j++){
                if (adjMatrix[perm[i], perm[j]] != otherMatrix[i, j]){
                    return false;
                }
            }
        }
        return true;
    }
```

```
class Program{
    static void Main(string[] args){
        Graph G1 = new Graph(3);
        G1.AddEdge(0, 1);
        G1.AddEdge(1, 2);
        G1.AddEdge(2, 0);

        Graph G2 = new Graph(3);
        G2.AddEdge(0, 2);
        G2.AddEdge(2, 1);
        G2.AddEdge(1, 0);

        if (G1.IsIsomorphic(G2)){
            Console.WriteLine("Grafovi su izomorfni!");
        }
        else{
            Console.WriteLine("Grafovi nisu izomorfni!");
        }
    }
}
```

- Algoritmu se ručno moraju zadati grafovi.
- U ovom primeru grafovi su
    - G1={(0,1), (1,2), (2,0)}
    - G2={(0,2), (2,1), (1,0)}

  I oni su izomorfni.

Podgraf, pokrivajući podgraf i indukovan podgraf

- Podgraf je pojam iz teorije grafova i označava graf G′=(V′, E′), koji je deo nekog većeg grafa G=(V, E).
- Formalno, podgraf G′G'G′ se definiše kao graf čiji su čvorovi (V′) podskup čvorova grafa G, a čije su grane (E′) podskup grana grafa G koje povezuju čvorove iz skupa V′.

Vrste podgrafa:

- Cvorovi i grane kao podskupovi
- Inducirani podgraf
- Razapeti (pokrivajući) podgraf

Čvorovi i grane kao podskupovi:

- $V' \subseteq V$
- $E' \subseteq E$
- Svaka grana u E′ mora povezivati čvorove iz V′

Inducirani podgraf:

- Ako je skup E′ definisan na osnovu skupa V′, tj E′ sadrži sve grane iz G koje povezuju čvorove iz V′, onda je G′ inducirani podgraf
- $E' = \{(u,v) \in E : u \in V',\ v \in V'\}$

Razapeti (Pokrivajući) podgraf:

- Podgraf G′ je razapeti podgraf ako V′ = V, tj sadrži sve čvorove grafa G ali ne mora sadržati sve grane grafa G.

Primeri:

- Osnovni podgraf:

  Graf G=(V,E) gde:

  - V={A,B,C,D}
  - E={{A,B},{B,C},{C,D}}

  Podgraf G′ može biti:

  - V′={A,B,C}
  - E′={{A,B},{B,C}}

- Inducirani podgraf:

  Za V′={A,C}, inducirani podgraf bi imao: E′={}, jer ne postoji grana koja direktno povezuje A i C u G.

- Razapeti podgraf:

  Graf G′ sadrži:

  - V′={A,B,C,D}
  - Podskup grana, npr. E′={{A,B},{C,D}}