Diskretna matematika - Zadatak 10

Grupa 10

Februar 2025

1 Uvod

Problem sedam mostova Kenigsberga jedan je od najpoznatijih zadataka u istoriji matematike, a istovremeno je poslužio kao polazna tačka za razvoj teorije grafova. U ovom dokumentu dajemo kratak istorijski osvrt na problem i Ojlerov doprinos njegovom rešavanju.

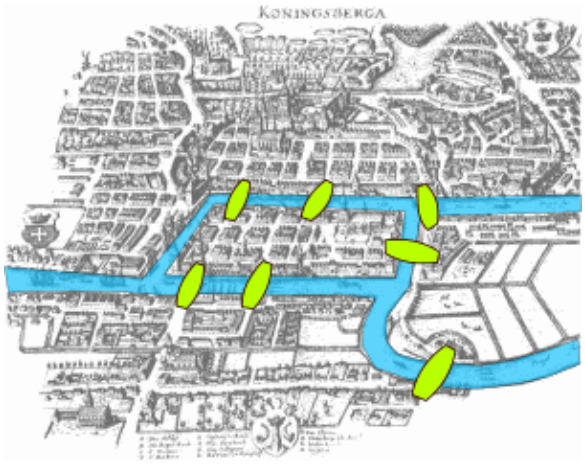
2 Istorijski kontekst

Grad Kenigsberg (danas Kalinjingrad, Rusija) bio je izgrađen na reci Pregel i posedovao je sedam mostova koji su povezivali različite delove grada. U 18. veku, stanovnici grada postavili su pitanje da li je moguće prošetati gradom tako da se svaki most pređe tačno jednom. Ovo pitanje je postalo intrigantno ne samo zbog praktične primene, već i zbog svoje apstraktne matematičke prirode.

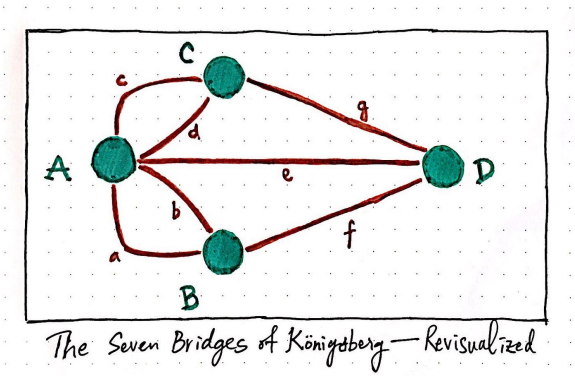
Leonhard Ojler je 1736. godine dao prvi matematički odgovor na ovaj problem. On je shvatio da se problem može preformulisati u apstraktnu teoriju, gde se kopna predstavljaju čvorovima, a mostovi linijama koje povezuju te čvorove. Ojler je dokazao da ne postoji putanja kojom se svaki most pređe tačno jednom, čime je postavio temelje moderne teorije grafova. Ovaj rad je značajan jer je prvi put uvedena ideja apstrakcije u matematičkom modeliranju realnih problema.

Problem sedam mostova Kenigsberga i Ojlerov pristup njegovom rešavanju predstavljaju ključnu prekretnicu u istoriji matematike. Ojlerov rad nije samo rešio konkretan problem, već je otvorio put razvoju novih matematičkih oblasti, među kojima je najvažnija teorija grafova. Danas, Ojlerov doprinos ostaje inspiracija u mnogim disciplinama, od matematike do računarstva i inženjeringa.

3 Prikaz problema i Ojlerovog grafa



Slika 1: Sedam mostova Kenigsberga



Slika 2: Ojlerov graf problema

4 Definicije

Definicija 1 *Neka je G graf bez petlji.* ***Ojlerov put*** *je put u grafu koji sadrži sve čvorove i grane tog grafa.* ***Ojlerova tura*** *je Ojlerov put čiji su početni i krajnji čvor jednaki.*

Formalno, za šetnju u grafu ćemo reći da je Ojlerov put ako zadovoljava sledeće uslove:

1. Šetnja sadrži sve čvorove grafa;
2. NE POSTOJE dve jednake grane u šetnji;
3. SVAKA grana grafa se pojavljuje u šetnji.

Definicija 2 *Graf je* ***Ojlerov*** *ako sadrži Ojlerovu turu. Graf je* ***polu-Ojlerov*** *ako sadrži Ojlerov put, ali ne i turu.*

5 Potreban i dovoljan uslov - Teoreme

Teorema 1 *Neka je G graf. Graf G je Ojlerov ako i samo ako je povezan i svaki čvor u G je parnog stepena.*

Dokaz.
($\Rightarrow$) Graf G je povezan po definiciji Ojlerovog grafa.
Neka je $v_1e_1v_2e_2\dots v_ne_nv_1$ Ojlerova tura u G. Posmatrajmo sada proizvoljan čvor $v \in V(G)$. Ako se čvor v pojavljuje l puta u konturi u slučaju kada je $v \neq u_1$, odnosno $l+1$ ako je $v = u_1$, onda je stepen tog čvora $d_G(v) = 2l$.
($\Leftarrow$) Posmatrajmo u G stazu najveće dužine:

$$u_1e_1u_2e_2\dots u_ne_nu_{n+1}$$

Pokazaćemo da su prvi i poslednji čvor isti, kao i da se svi čvorovi i grane grafa pojavljuju u toj stazi.

(i) $u_1 = u_{n+1}$: Ako pretpostavimo suprotno, da je $u_1 \neq u_{n+1}$, onda je u toj konturi neparan broj grana incidentan sa u_1 u toj stazi. Kako je stepen čvora u_1 paran, postoji grana $e \in E(G)$ koja nije sadržana u posmatranoj stazi. U tom slučaju bismo mogli kreirati dužu stazu od posmatrane, što dovodi do kontradikcije.

(ii) $\{u_1, \dots, u_n\} = V(G)$: Pretpostavimo suprotno, da postoje čvorovi koji nisu na posmatranoj stazi. Kako je G povezan, postoji grana $\{u_i, v\},\ i \in \{1, \dots, n\}$, sa osobinom $v \notin \{u_1, \dots, u_n\}$. U tom slučaju možemo konstruisati stazu veće dužine od posmatrane.

(iii) $\{e_1, \dots, e_n\} = E(G)$: Sada kada znamo da je posmatrana staza u stvari kontura koja sadrži sve čvorove grafa, pokazaćemo da su sve grane grafa na toj konturi. Ako pretpostavimo suprotno, da postoji grana koja nije na toj konturi, onda bismo ponovo mogli konstruisati dužu stazu od posmatrane.

Teorema 2 *Neka je $G = (V, E)$ graf koji nije Ojlerov. Graf G je polu-Ojlerov ako i samo ako je G povezan i ima tačno dva čvora neparnog stepena.*

Dokaz. ($\Rightarrow$) Sličnim rezonovanjem kao u prethodnom dokazu, za svaki čvor grafa koji nije na krajevima Ojlerovog puta, na tom putu se pojavljuje paran broj grana koje su incidentne sa tim čvorom. Za dva čvora na kraju puta imamo, osim eventualnog parnog broja grana unutar staze, još po jednu granu koja im je incidentna, odakle dobijamo da ta dva čvora imaju neparne stepene.

($\Leftarrow$) Neka su u i v jedini čvorovi grafa neparnog stepena. Posmatrajmo sada graf G' koji dobijamo

od grafa G dodavanjem nove grane e koja je incidentna sa čvorovima u i v (ona može biti paralelna nekim već postojećim granama). U grafu G' su sada svi čvorovi parnog stepena. Prema Teoremi 90, G' sadrži Ojlerovu turu. Po definiciji, ta tura sadrži granu e. Oduzimanjem grane e iz Ojlerove ture grafa G' dobijamo Ojlerov put u grafu G.

Šta je Pruferov kod?

Pruferov kod je način da se svako **označeno stablo** sa n čvorova kodira kao niz celih brojeva dužine $n-2$. Ovaj kod je uveo nemački matematičar *Heinz Prüfer* 1918. godine i koristi se za:

- Kodiranje i rekonstrukciju stabala
- Brojanje označenih stabala (Cayleyjeva formula)
- Optimizaciju memorije pri čuvanju grafa
- Takmičarsko programiranje i teoriju informacija

Broj označenih stabala sa n čvorova je n^{n-2}, i tačno toliko različitih Pruferovih kodova postoji.

Algoritam za kodiranje (stablo u Pruferov kod)

Za dato označeno stablo:

1. Dok stablo ima više od 2 čvora:
   - Pronađi list (čvor stepena 1) sa najmanjom oznakom
   - U Pruferov kod upiši suseda tog lista
   - Ukloni taj list iz stabla
2. Kada ostanu samo dva čvora, proces se završava

Algoritam za dekodiranje (kod u stablo)

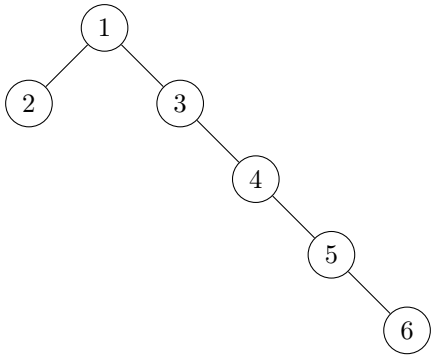
Za dati Pruferov kod dužine $n-2$:

1. Inicijalizuj niz frekvencija čvorova (od 1 do n)
2. Svaki broj iz koda povećava frekvenciju svog pojavljivanja
3. Za svaki broj iz koda:
   - Pronađi najmanji čvor sa frekvencijom 0 (to je list)
   - Poveži ga sa trenutnim čvorom iz koda
   - Ažuriraj frekvencije
4. Na kraju ostaju dva čvora koji se međusobno povezuju

Primer

Razmotrimo stablo sa 6 čvorova i sledećim granama:

$$(1,2),\ (1,3),\ (3,4),\ (4,5),\ (5,6)$$



Koraci kodiranja

1. Listovi: 2, 6 → najmanji je 2 → sused je 1 → dodaj 1 → ukloni 2
2. Listovi: 1, 6 → najmanji je 1 → sused je 3 → dodaj 3 → ukloni 1
3. Listovi: 3, 6 → najmanji je 3 → sused je 4 → dodaj 4 → ukloni 3
4. Listovi: 4, 6 → najmanji je 4 → sused je 5 → dodaj 5 → ukloni 4
5. Ostaju čvorovi 5 i 6 → kraj

Pruferov kod: $\boxed{1, 3, 4, 5}$

Rekonstrukcija iz koda [1, 3, 4, 5]

1. $n = 6$ → čvorovi su 1 do 6
2. Broj pojavljivanja:
   - 1: 1 put, 3: 1, 4: 1, 5: 1
   - 2 i 6: 0 → to su listovi
3. Povezivanje:
   - 1 → najmanji bez pojavljivanja je 2 → grana (1,2)
   - 3 → najmanji je 6 → (3,6)
   - 4 → najmanji je 1 → (4,1)
   - 5 → najmanji je 3 → (5,3)
   - Ostali: 4 i 5 → (4,5)

Dobijeno stablo je identično početnom.

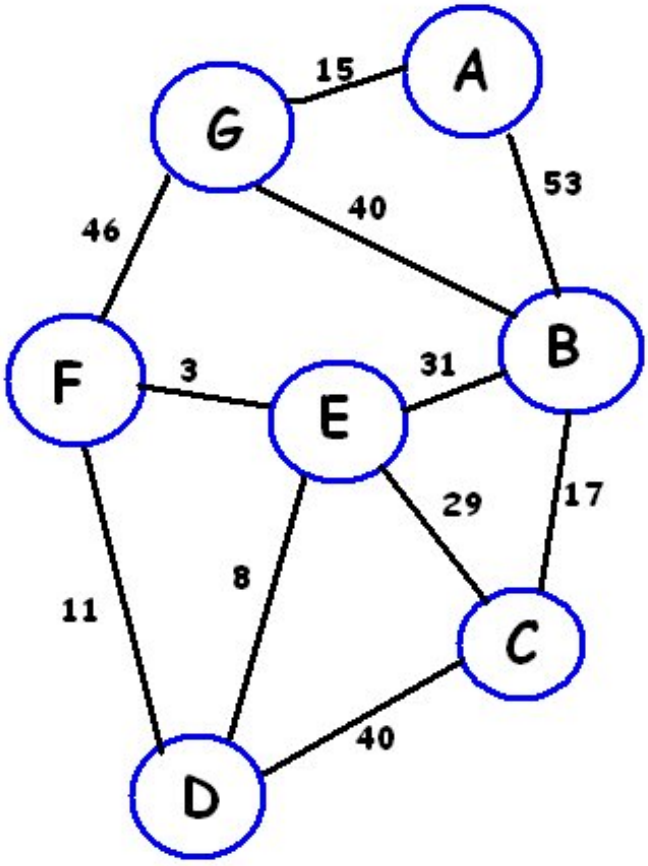
Zaključak

Pruferov kod omogućava efikasno kodiranje i rekonstrukciju označenih stabala, posebno koristan u kombinatorici i algoritamskim problemima. Svako stablo može biti predstavljen kodom dužine $n - 2$ i obrnuto.

Težinski graf

Definicija: Neka je $G = (V, E)$ graf, gde je V konačan skup čvorova, a E konačan skup grana (uređenih ili neuređenih parova čvorova). Ako je svakoj grani $e \in E$ pridružena stvarna vrednost (tzv. težina), tada se kaže da je G **težinski graf**, a funkcija težine je funkcija $w : E \to \mathbb{R}$ koja svakoj grani dodeljuje njenu težinu.

Dakle, težinski graf je uređena trojka (V, E, w), gde w označava funkciju težine.



Slika 3: Primer težinskog grafa

Hierholzerov algoritam za određivanje Ojlerove ture

Algoritam koristi DFS strategiju za generisanje Ojlerove ture u neusmerenom grafu. Svaka grana se koristi tačno jednom. Početni čvor može biti bilo koji sa stepenom većim od nule.

```
// G je graf: G[v] sadrzi indekse grana koje izlaze iz v
vector<vector<int>> G;
vector<pair<int, int>> edges; // grane kao parovi (a, b)
vector<bool> used; // da li je grana vec iskoriscena
vector<int> euler; // rezultat: redosled cvorova

void dfs(int v) {
    while (!G[v].empty()) {
        int id = G[v].back(); G[v].pop_back();
        if (used[id]) continue;
        used[id] = true;
        int u = edges[id].first ^ edges[id].second ^ v; // drugi kraj grane
        dfs(u);
    }
    euler.push_back(v);
}
```

Grana između čvorova a i b čuva se kao par (a, b) u nizu `edges`, a čvorovi u `G[a]` i `G[b]` sadrže indeks te grane. Funkcija koristi XOR trik da odredi suprotan kraj grane bez dodatne logike.

Objašnjenje rada algoritma

Hierholzerov algoritam pronalazi Ojlerovu turu tako što rekurzivno gradi konturu i postepeno je proširuje sve dok ne iskoristi sve grane grafa.

1. Kreće se iz proizvoljnog čvora v (poželjno iz čvora koji ima nenulti stepen).
2. Svaki put kada naiđe na još neiskorišćenu granu, bira je, označava kao iskorišćenu, i prelazi na drugi kraj te grane.
3. Proces se nastavlja dok se iz trenutnog čvora ne iscrpe sve grane (tj. lista susedstva postane prazna).
4. Kada to nastupi, algoritam se "vraća nazad"— tj. ubacuje taj čvor u niz `euler`.
5. Pošto se koristi rekurzivna strategija dubine (DFS), čvorovi se u `euler` ubacuju tek kada su sve grane koje iz njih izlaze već iskorišćene.
6. Na kraju se niz `euler` obrne ako je potrebno, jer on sadrži čvorove u obrnutom redosledu posećivanja.

Ono što čini ovaj algoritam efikasnim je to što se svaka grana prolazi tačno jednom i koristi se struktura liste susedstva radi brzog pristupa. Funkcija koristi tzv. XOR trik da identifikuje drugi kraj grane bez potrebe da se dodatno proverava da li je čvor trenutni ili susedni:

$$u = a \oplus b \oplus v$$

gde su (a, b) krajevi grane, a v je trenutni čvor. Na taj način se uvek dobija "onaj drugi"čvor.
Vremenska složenost: $O(E)$, gde je E broj grana u grafu. Svaka grana se poseti tačno jednom.