

# Število podpoti pri grafih z danim ciklomatskim številom

*Kratko poročilo projektne naloge*  
*Skupina 18: Maja Košmrlj, Jan Maradin*

## Uvod

V projektu obravnavamo *povezane* grafe  $G = (V, E)$ , za katere je definirano *ciklomatsko število*:

$$\mu(G) = |E| - |V| + 1.$$

To število pove, koliko neodvisnih ciklov vsebuje graf oziroma koliko povezav bi morali odstraniti, da bi dobili gozd.

Cilj projektne naloge je raziskati, kako se obnaša *število podpoti in geodetskih podpoti grafa*, torej število vseh enostavnih poti (vključno s potmi dolžine 0), pri grafih z danim ciklomatičnim številom. Zanima nas, kateri grafi imajo za dano število vozlišč  $|V| = n$  in dano ciklomatsko število  $\mu(G) = k$  najmanjše oziroma največje število podpoti.

## Teoretično ozadje

**Definicija 1** (Podpot). Naj bo  $G = (V, E)$  povezan graf in  $P = v_1v_2 \dots v_k$  pot v grafu  $G$ . *Podpot* grafa  $G$  je vsaka zaporedna podzaporedje oglišč te poti, torej vsaka pot oblike

$$v_i v_{i+1} \dots v_j, \quad \text{kjer } 1 \leq i < j \leq k.$$

Število podpoti  $p_n(G)$  skupno število vseh enostavnih poti v grafu, pri čemer se šteje tudi vsako vozlišče kot pot dolžine 0.

**Definicija 2** (Geodetsko število). Naj bo  $G = (V, E)$  povezan graf. *Geodetsko število podpoti*  $gp_n(G)$ , definiramo kot skupno število vseh geodetskih enostavnih poti v grafu. Za vsaka dva vozlišča  $u, v \in V(G)$  označimo z  $d(u, v)$  razdaljo med njima, z  $\sigma^*(u, v)$  pa število vseh enostavnih  $u-v$  poti dolžine  $d(u, v)$ . Tedaj je

$$gp_n(G) = \sum_{u, v \in V(G)} \sigma^*(u, v).$$

Vračunane so tudi poti dolžine 0, torej  $u = v$ . Geodetsko število podpoti tako meri, koliko najkrajših enostavnih poti vsebuje graf.

**Kaktusni graf** je graf, v katerem se poljubna dva cikla stikata v največ enem vozlišču. To pomeni, da se cikli lahko dotikajo, vendar se ne prekrivajo.

**Graf PTC(n,k)** (angl. *Pseudo Triangle Chain*), oziroma psevdotrikotna veriga, je poseben primer kaktusnega grafa z  $n$  vozliščci in  $k$  cikli. Vsi cikli so trikotniki, povezani v

verigo in zato graf dosega največje možno število podpoti med vsemi takšnimi grafi. To je razlog, da jih v projektu pogosto uporabimo kot osnovni primer (tj. zgornja meja) za primerjavo z drugimi grafi, ki imajo enako ciklomatično število  $\mu(G)$ .

**Izrek** (Knor, 2025) Graf  $PTC(n, k)$  enolično maksimizira število podpoti med vsemi kaktusnimi grafi z  $n$  vozlišči in  $k \geq 2$  cikli.

## Plan dela

Projektno nalogo bova ločila na dva dela.

### 1. Analiza za majhne grafe.

V okolju SageMath bova s pomočjo funkcije *graphs.nauty\_geng* generirala vse povezane grafe z števili vozlišč  $n \leq 8$  in za vsak graf izračunali:  $|V|$ ,  $|E|$ ,  $\mu(G)$ ,  $gp_n(G)$  in  $p_n(G)$ . Pri majhnih grafih bova  $p_n(G)$  izračunali z iskanjem vseh enostavnih podpoti med pari vozlišč. Ker SageMath ne vsebuje vgrajene funkcije za štetje vseh enostavnih podpoti, sva uporabila rekurzivni pristop. Funkcija `all_simple_paths(G, start, end)` poišče vse poti med vozliščema `start` in `end`, tako da postopno obiskuje sosednja vozlišča, pri čemer se izogiba že obiskanim vozliščem. Za vsak par vozlišč  $(u, v)$  funkcija prešteje vse enostavne podpoti, nato pa se vse dobljene vrednosti seštejejo v končno vrednost  $p_n(G)$ . Na podlagi rezultatov bova določila grafe, ki imajo minimalni in maksimalni  $p_n(G)$  pri enakem  $\mu(G)$ .

### 2. Analiza za večje grafe.

Za grafe z več kot osmimi vozlišči bova uporabila metodo simuliranega ohlajanja, saj popolna generacija vseh neizomorfnih grafov pri tej velikosti postane računsko prezahtevna. Za dani vrednosti  $|V| = n$  in  $\mu(G) = k$  bova najprej ustvarila povezani graf z  $n$  vozlišči in točno  $m = n - 1 + k$  povezavami, za katerega si mislimo da je podoben optimalnemu. Ta graf bo začetno stanje algoritma. V vsakem koraku bova izvedla majhno lokalno spremembo grafa (t.i. *edge swap*), pri kateri odstranimo en obstoječi rob in ga nadomestimo z novim, tako da ohranimo število povezav in s tem tudi ciklomatsko število. Novo stanje sprejmemo vedno, kadar se vrednost  $p_n(G)$  izboljša. Če se vrednost poslabša, takšno stanje sprejmemo le z verjetnostjo  $e^{(\Delta/T)}$ , kjer je  $\Delta$  razlika med novim in trenutnim stanjem,  $T$  pa trenutna temperatura. V algoritmu bova začela pri višjih temperaturah, da bova lahko z večjo verjetnostjo našla globalni ekstrem in ne samo lokalnega, glede na graf, ki ga bova podala kot začetno stanje. Ko temperaturo postopoma znižujemo po pravilu  $T_{k+1} = \alpha T_k$ , postane algoritem vedno bolj "konservativen" in sprejema skoraj izključno korake, ki izboljšajo rezultat. Po koncu iteracij bova kot končno rešitev vzela tisti graf, ki je med izvajanjem dosegel največjo (oziroma najmanjšo) vrednost  $p_n(G)$ , ter ga primerjala z znanimi ekstremnimi kaktusnimi grafi za iste vrednosti  $n$  in  $\mu(G)$ .

## Pričakovani rezultati in hipoteze

Pričakujeva, da bo vrednost  $p_n(G)$  najnižja pri drevesih in bo naraščala z večanjem ciklomatskega števila  $\mu(G)$ , saj dodatni cikli omogočajo več različnih enostavnih poti med

pari vozlišč. Na podlagi znane literature o kaktusnih grafih je najina domneva, da bo minimalni  $p_n(G)$  dosežen pri zaporedno povezanih ciklih, kjer so cikli razporejeni v verigo in tako le lokalno povečujejo število poti. Največji  $p_n(G)$  pa pričakujeva pri strukturi, kjer imajo vsi cikli skupno vozlišče, saj takšna razporeditev ustvarja veliko število alternativnih poti preko centralnega vozlišča.

Pri večjih grafih pričakujeva pojav lokalnih ekstremov, saj dodatne povezave ne povečajo vedno števila enostavnih poti. Na podlagi literature o kaktusnih grafih domnevava, da bodo strukture, pri katerih cikli delijo vozlišča ali so med seboj preblizu, ustvarile le majhen prirastek  $p_n(G)$ , medtem ko bodo jasno ločeni cikli še naprej povečevali število podpoti. Zato pričakujeva, da se bodo lokalni maksimumi in minimumi pri večjih  $n$  pojavljali predvsem v grafih, kjer je razporeditev ciklov manj učinkovita pri ustvarjanju novih poti, kar bova preverila s simuliranim ohlajanjem.