Subpath number v kubičnih grafih: minimizacija in protimeri za grafe L_n

Žiga Obradović, Lovro Levačić

6. december 2025

1 Uvod

V projektu obravnavamo problem minimizacije števila poti v razredu kubičnih grafov, torej grafov, kjer ima vsako vozlišče stopnjo 3. Naj bo $G = (V, E)$ povezan graf. *Subpath number* grafa G, označen s $pn(G)$, definiramo kot število vseh preprostih poti v grafu, vključno s trivialnimi potmi dolžine 0.

Preprosta pot dolžine ℓ je zaporedje vozlišč

$$(v_0, v_1, \ldots, v_\ell),$$

kjer se vozlišča ne ponavljajo ($v_i \neq v_j$ za vse $0 \leq i < j \leq \ell$), vsak par zaporednih vozlišč pa je povezan z robom grafa (za vsak $1 \leq i \leq \ell$ je $v_{i-1}v_i \in E$). Ker je graf neorientiran, poti, ki gredo v nasprotni smeri, štejemo ločeno.

V literaturi je bila postavljena domneva, da za vsako sodo število vozlišč $n \geq 10$ obstaja natanko en kubični graf L_n, ki minimizira subpath number med vsemi kubičnimi grafi na n vozliščih. Znano je, da domneva za dovolj velika n ne drži, ni pa znano, pri katerem najmanjšem n se pojavi prva protimera.

Cilji projekta so bili:

- izčrpno preveriti domnevo za manjša n, kjer je število kubičnih grafov še obvladljivo;
- za večja n uporabiti metahevristiko *simulated annealing* in poiskati grafe z manjšim $pn(G)$ od $pn(L_n)$;
- konstruirati nove družine grafov (z uporabo gradnikov iz definicije L_n) in primerjati njihov subpath number z $pn(L_n)$.

2 Teoretično ozadje

2.1 Subpath number

Za poljuben povezan graf $G = (V, E)$ definiramo

$$pn(G) = \#\{\text{preprostih poti v } G\},$$

pri čemer kot poti štejemo tudi trivialne poti dolžine 0 (vsako vozlišče samo zase). Subpath number je torej globalna mera, ki upošteva vse možne preproste poti, ne le najkrajših ali najdaljših.

Že za zelo majhne grafe se $pn(G)$ hitro poveča. Na primer: en sam rob ima $pn(G) = 4$ (dve trivialni poti in dve usmerjeni poti po robu), trikotnik ima $pn(G) = 15$, kvadrat $pn(G) = 28$, popoln graf K_4 pa $pn(G) = 64$. To nakazuje, da bo izračun $pn(G)$ v splošnem eksponenten v številu vozlišč in je zato primeren predvsem za grafe z zmernim n.

2.2 Družina grafov L_n in domneva

Grafi L_n so definirani za soda $n \geq 10$ in so zgrajeni iz več kopij grafa $K_4 - e$ (popoln graf na štirih vozliščih, iz katerega odstranimo en rob), ki so povezane v linearno verigo. Na koncih verige so dodani t. i. pendant bloki.

Natančneje:

- če je $n = 4q - 2$, je L_n sestavljen iz $(n - 10)/4$ kopij grafa $K_4 - e$, povezanih v verigo, na obeh koncih pa sta pendant bloka na 5 vozlišč;
- če je $n = 4q$, je L_n sestavljen iz $(n - 12)/4$ kopij grafa $K_4 - e$, na koncih pa sta pendant bloka na 5 in 7 vozlišč.

Geometrijsko imajo grafi L_n podobo verige blokov, ki se zaključuje z dvema malce večjima komponentama. Na zaslonu zvezka smo za ilustracijo narisali grafe L_n za $n = 10, 12, 14, 16, 18, 20, 22, 24$; pri majhnih n je veriga kratka, za $n = 24$ pa že vidimo izrazito "kačasto" strukturo.

Domneva, ki jo preverjamo, je:

Domneva 10. *V razredu kubičnih grafov na n vozliščih, kjer je n sodo, je graf L_n edini graf, ki minimizira subpath number.*

Ker je bilo teoretično dokazano, da domneva za dovolj velika n ne drži, je glavno odprto vprašanje, pri katerih najmanjših n se L_n neha obnašati kot minimizator.

3 Metodologija

3.1 Konstrukcija družin grafov

Pri implementaciji ne uporabljamo formalne definicije L_n neposredno, temveč ga zgradimo iz treh gradnikov, ki izhajajo iz grafa $K_4 - e$:

- **levi gradnik** je graf $K_4 - e$, na katerega dodamo novo vozlišče, povezano z obema vozliščema stopnje 2; dobimo pendant blok na 5 vozliščih z naravno "priklopno" točko;
- **srednji gradnik** je graf $K_4 - e$ z označenima dvema vozliščema stopnje 2, ki služita kot levi in desni priklopni točki v verigi;
- **desni gradnik** je razširitev grafa $K_4 - e$ z dvema dodatnima vozliščema in zgornjim vozliščem, kar skupaj tvori pendant blok na 7 vozliščih.

Graf L_n dobimo tako, da:

1. začnemo z levim gradnikom;
2. dodamo ustrezno število srednjih gradnikov in jih povezujemo v verigo prek označenih vozlišč stopnje 2;
3. na koncu verigo zaključimo bodisi z levim bodisi z desnim gradnikom, tako da ima graf natanko n vozlišč in ostane kubičen.

Podobno smo definirali dve novi družini grafov, namenjeni iskanju protimerov:

star1 (funkcija *build_star*) grafe zgradimo tako, da na kratko "hrbtenico" vozlišč stopnje 3 pripenjamo več levi h gradnikov. Dobljena struktura ima eno osrednje vozlišče in več "krakov", na katerih so pendant bloki. Vizualno so ti grafi podobni zvezdi z več kraki.

star2 (funkcija *build_star2*) uporabi podobno shemo, vendar hrbtenico po potrebi podaljša še z vmesnim (srednjim) gradnikom. Ta družina je bolje prilagojena določenim vrednostim n (modularno po 6).

V zvezku smo narisali grafe obeh družin za npr. $n = 10, 12, \ldots, 32$. Pri L_n se vidi skoraj čista veriga, pri zvezdastih grafih pa izrazit osrednji vozlišči in trije ali več krakov, kar intuitivno zmanjšuje število možnih dolgih poti.

3.2 Izračun subpath number

Za izračun $pn(G)$ potrebujemo način, kako sistematično našteti vse preproste poti v grafu, ne da bi kakšno izpustili ali prešteli dvakrat.

Uporabimo naslednjo idejo:

1. Vsakemu vozlišču grafa priredimo indeks od 0 do $n-1$, da lahko množico obiskanih vozlišč predstavimo z bitno masko.
2. Za vsako vozlišče v zaženemo globinsko iskanje (DFS) po grafu, pri čemer maska označuje, katera vozlišča so že na trenutni poti.
3. Ob vsakem rekurzivnem klicu štejemo trenutno pot kot eno (tudi pot dolžine 0, kjer je na poti samo začetno vozlišče).
4. Iz vozlišča lahko nadaljujemo le na soseda, ki še ni v maski. Tako se nobeno vozlišče na isti poti ne ponovi in dobimo preproste poti.

Ker za vsako vozlišče v sprožimo DFS, algoritem res obišče vse preproste poti, ki se začnejo v v, in vsako pot prešteje natanko enkrat — tisti, ki se konča v zadnjem rekurzivnem klicu. Uporaba bitnih mask nam omogoča, da o obiskanih vozliščih ne vodimo počasnejših seznamov, pač pa preprosto testiramo posamezne bite.

Časovna zahtevnost je eksponentna v n, kar smo potrdili tudi eksperimentalno: čas izvajanja hitro naraste, ko presežemo približno $n = 20$ vozlišč. Zato je ta funkcija primerna predvsem za:

- grafe z največ približno 20–22 vozlišči, kjer jo lahko uporabimo tudi za izčrpen pregled;
- evalvacijo posameznih kandidatov v metahevristiki, kjer število ocen (korakov) omejimo.

3.3 Generiranje vseh kubičnih grafov

Za manjše n želimo domnevo preveriti izčrpno, zato potrebujemo generator vseh neizomorfnih povezanih kubičnih grafov na n vozliščih. Uporabimo vgrajeno funkcijo v SageMath-u (nauty_geng), ki temelji na programu *geng* iz paketa *nauty*. S parametri tej funkciji predpišemo, da:

- ima vsak graf natanko n vozlišč,
- je minimalna in maksimalna stopnja vozlišč enaka 3 (graf je kubičen),
- je graf povezan,
- dobimo po en predstavnik iz vsakega izomorfnega razreda.

Dobimo generator, ki enega za drugim vrača predstavnike izomorfnih razredov vseh povezanih 3-regularnih grafov na n vozliščih. S tem lahko za vsak graf izračunamo $pn(G)$ in določimo globalni minimum.

Eksperimentalno smo izračunali število takih grafov za $n = 4, 6, \ldots, 18$, kar je povzeto v Tabeli 1.

Rast je zelo hitra: že pri $n = 18$ imamo več kot 40 000 grafov, pri $n = 20$ pa bi jih bilo že več kot pol milijona. To upravičuje prehod na metahevristične metode za $n \geq 20$.

n	# kubičnih grafov
4	1
6	2
8	5
10	19
12	85
14	509
16	4060
18	41301
20	510489
22	7319447

Tabela 1: Število neizomorfnih povezanih kubičnih grafov na n vozliščih.

3.4 Metahevristika *simulated annealing*

Za večja n se izčrpen pregled ne izplača, zato uporabimo metahevristiko *simulated annealing*. Ta metoda posnema fizični proces ohlajanja snovi: na začetku dopušča tudi poslabšanja, kasneje pa vedno bolj preferira izboljšave.

Potrebujemo dve komponenti:

Sosednji graf. Iz danega kubičnega grafa G generiramo soseda z eno 2-robno zamenjavo: izberemo dva roba $\{u, v\}$ in $\{x, y\}$ s štirimi paroma različnimi vozlišči in ju preklopimo v eno od alternativnih paritev $\{u, x\}, \{v, y\}$ ali $\{u, y\}, \{v, x\}$. Takšno operacijo imenujemo *double-edge swap*. Zanima nas samo rezultat, ki:

- nima zank ali večrobov,
- je še vedno 3-regularen,
- je povezan.

Če taka zamenjava v omejenem številu poskusov ne obstaja, soseda ne zamenjamo.

Pravilo sprejemanja. Začetni graf je običajno L_n. V vsakem koraku izračunamo subpath number trenutnega grafa G (energija E) in kandidata G' (energija E'). Kandidata vedno sprejmemo, če $E' \leq E$. Če je $E' > E$, ga sprejmemo z verjetnostjo

$$\exp\left(-\frac{E' - E}{T}\right),$$

kjer je T trenutna temperatura. Temperaturo po vsakem koraku pomnožimo z faktorjem $0 < \alpha < 1$ (na primer $\alpha = 0{,}999$). S tem je na začetku sprejemanje poslabšanj pogostejše, proti koncu pa redko, kar omogoča izogib lokalnim minimumom.

Algoritem vrača najboljši graf, ki ga je videl (najmanjši $pn(G)$), ter potek energij skozi korake, kar smo uporabili za grafično analizo.

4 Eksperimentalni rezultati

4.1 Preverjanje domneve za manjša n

Za manjša n smo uporabili kombinacijo izčrpnega pregleda in usmerjenega iskanja. Za $n = 16$ in $n = 18$ smo za vse kubične grafe izračunali $pn(G)$ in shranili grafe, pri katerih je $pn(G) < pn(L_n)$. Lastnosti teh rezultatov so povzete v Tabeli 2.

n	$pn(L_n)$	minimalni $pn(G)$	pn(star)	komentar
16	12744	3640	3640	zvezdasti graf doseže globalni minimum
18	22532	7072	7072	zvezdasti graf doseže globalni minimum

Tabela 2: Primerjava subpath number za L_n, zvezdaste grafe in globalni minimum pri $n = 16$ in $n = 18$.

Vidimo, da je pri obeh teh velikostih graf L_n daleč od minimizatorja: pri $n = 16$ ima skoraj štirikrat več poti kot optimalni graf, pri $n = 18$ pa več kot trikrat več. V obeh primerih globalni minimum doseže ena izmed zvezdastih konstrukcij, ki ima vizualno izrazito osrednje vozlišče in tri krake s pendant bloki.

To pomeni, da domneva 10 že za $n = 16$ in $n = 18$ ne drži: ne samo da L_n ni edini minimizator, sploh ni minimizator.

4.2 Rezultati metahevristike

Metahevristiko simulated annealing smo zagnali za $n = 10, 12, 14, 16, 18, 20, 22$, pri čemer je bil začetni graf vedno L_n. Število korakov je bilo 1000, začetna temperatura $T_0 = 1$, faktor ohlajanja $\alpha = 0{,}999$.

Najboljše dosežene vrednosti $pn(G)$ (energije) so v Tabeli 3.

n	najboljši $pn(G)$ z SA	čas izvajanja
10	1276	$\approx$ 1,5 s
12	3076	$\approx$ 3,6 s
14	5504	$\approx$ 5,3 s
16	8248	$\approx$ 9,8 s
18	7072	$\approx$ 16,0 s
20	14212	$\approx$ 30,0 s
22	20580	$\approx$ 44,4 s

Tabela 3: Najboljše vrednosti subpath number, ki jih je našel simulated annealing.

Za $n = 16$ in $n = 18$ lahko te rezultate primerjamo z znanimi globalnimi minimumi:

- pri $n = 16$ je globalni minimum 3640, simulated annealing pa je v enem zagonu našel graf z $pn(G) = 8248$, torej precej boljši od L_{16}, vendar še ne optimalen;
- pri $n = 18$ je globalni minimum 7072, enako pa znaša tudi najboljši rezultat simulated annealinga, kar pomeni, da je metahevristika v tem primeru dosegla globalni minimum.

Potek energije skozi korake (narisan v zvezku) lepo pokaže značilno obnašanje: na začetku večja nihanja in občasna poslabšanja, proti koncu pa stabilizacija okoli lokalnega oziroma globalnega minimuma.

4.3 Primerjava z zvezdastimi družinami

Za različna soda n smo izračunali $pn(G)$ za zvezdasti družini star1 in star2 in jih primerjali med seboj. Rezultati za $n = 14, 20, 26, 32, \ldots, 98$ so povzeti v Tabeli 4.

Vidimo, da je za manjša n (14, 20, 26) boljša konstrukcija star2, od $n = 32$ naprej pa sistematično zmaguje star1. V obeh primerih se vrednosti $pn(G)$ rastejo približno kvadratno z n, pri čemer je star1 asimptotsko boljši (ima počasnejšo rast).

n	pn(star1)	pn(star2)	boljši graf
14	7188	5504	star2
20	12816	11708	star2
26	19596	19064	star2
32	27528	27572	star1
38	36612	37232	star1
44	46848	48044	star1
50	58236	60008	star1
56	70776	73124	star1
62	84468	87392	star1
68	99312	102812	star1
74	115308	119384	star1
80	132456	137108	star1
86	150756	155984	star1
92	170208	176012	star1
98	190812	197192	star1

Tabela 4: Subpath number za dve zvezdasti družini grafov.

V kombinaciji z analizo za $n = 16$ in $n = 18$ lahko sklepamo, da so zvezdaste konstrukcije zelo močni kandidati za minimizatorje subpath number v razredu kubičnih grafov pri večjih n, medtem ko grafi L_n že pri zmerno velikih n močno zaostajajo.

5 Zaključek in nadaljnje delo

V poročilu smo opisali problem minimizacije subpath number v razredu kubičnih grafov, družino grafov L_n in prvotno domnevo o njihovi optimalnosti. Implementirali smo:

- konstrukcijo grafov L_n in dveh zvezdastih družin,
- natančen, a eksponenten algoritem za izračun $pn(G)$,
- generiranje vseh neizomorfnih povezanih kubičnih grafov za majhna n,
- metahevristični algoritem simulated annealing s 2-robno zamenjavo kot sosednjo operacijo.

Eksperimentalno smo pokazali:

- da se število kubičnih grafov zelo hitro povečuje (pri $n = 18$ jih je že več kot 40 000),
- da grafi L_n pri $n = 16$ in $n = 18$ nikakor niso minimizatorji subpath number; zvezdasti grafi imajo tam več kot trikrat manj poti,
- da metahevristika simulated annealing uspešno najde grafe z bistveno manjšim $pn(G)$ od $pn(L_n)$, v primeru $n = 18$ pa celo globalni minimum,
- da zvezdaste konstrukcije star1 in star2 za večja n sistematično dajejo manjše subpath number in so zato resni kandidati za "prave" minimalne strukture.

Domneva 10 je torej že za relativno majhna n napačna. Zvezdaste konstrukcije nakazujejo novo možno strukturo minimalnih kubičnih grafov, kjer večina vozlišč leži v pendant blokih, medtem ko je hrbtenica grafa kratka.