Tukaj pišem svoj del, da bova kasneje združila

Žiga

10. december 2025

1 Preverjanje domneve za majhne n

Če smo domnevo želeli preveriti na vseh kubičnih grafih na n vozliščih, smo morali najprej te grafe generirati. Izkazalo se je, da število kubičnih grafov na n vozliščih eksponentno raste s številom vozlišč, zato smo domnevo preverili na vseh kubičnih grafih le za $n \leq 18$.

n	# kubičnih grafov
10	19
12	85
14	509
16	4.060
18	41.301
20	510.489
22	7.319.447

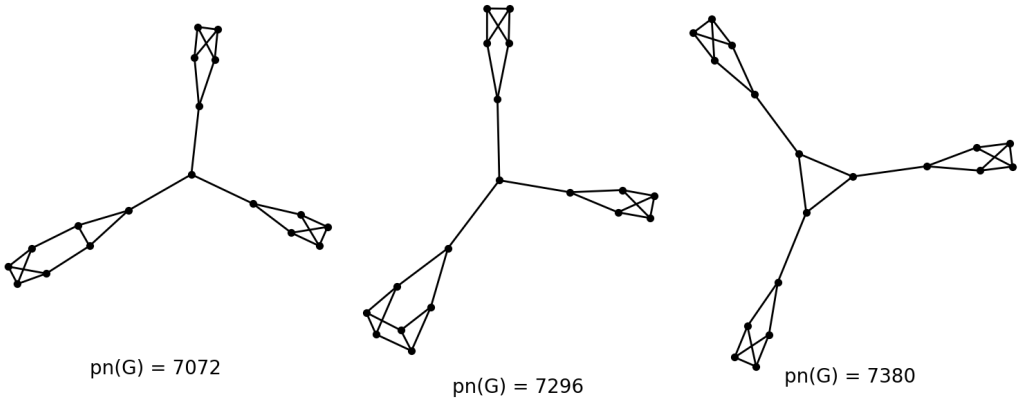
Tabela 1: Število neizomorfnih povezanih kubičnih grafov na n vozliščih.

Za $n \in \{10,\ 12,\ 14,\ 16,\ 18\}$ smo pridobili vse neizomorfne povezane kubične grafe na n vozliščih in za vsak graf G izračunali $pn(G)$. Za vsak n smo posebej primerjali število podpoti generiranega grafa $pn(G)$ s številom podpoti grafa L_n $pn(L_n)$. Rezultati so prikazani v spodnji tabeli.

n	število boljših grafov	minimalno število podpoti	$pn(L_n)$
10	0	1.276	1.276
12	0	3.076	3.076
14	0	5.504	5.504
16	6	3.640	12.744
18	23	7.072	22.532

Tabela 2: Število vseh grafov na n vozliščih z manjšim številom podpoti od grafa L_n.

Na podlagi rezultatov vidimo, da domneva velja za $n \in \{10,\ 12,\ 14\}$, medtem ko lahko domnevo za $n \in \{16,\ 18\}$ zavrnemo. Namreč v primeru ko je $n = 16$ smo našli 6 grafov z manjšim številom podpoti od grafa L_{16}, v primeru $n = 18$ pa že kar 23 grafov z manjšim številom podpoti od grafa L_{18}.



Slika 1: Grafi z najmanjšim številom podpoti za $n = 18$.

Ker je za $n \geq 20$ izračun *subpath number* vedno zahtevnejši, grafov pa vedno več, bomo za iskanje protiprimerov uporabili drugačno metodo. Ta metoda se mimenuje *simulated annealing*.

2 Simulated annealing

Za iskanje kubičnih grafov z manjšim številom podpoti od L_n grafa smo za $n \geq 20$ uporabili metahevristični algoritem *simulated annealing* (SA). Gre za probabilistično metodo globalne optimizacije, ki posnema proces fizikalnega ohlajanja kovin: sistem se sprva nahaja pri visoki temperaturi in lahko sprejema tudi poslabšanja, postopoma pa se temperatura znižuje, kar zmanjšuje verjetnost sprejemanja slabših rešitev. Pri dovolj počasnem ohlajanju se algoritem z visoko verjetnostjo približa globalnemu minimumu energetske funkcije, katera je v našem primeru *subpath number*.

2.1 Delovanje

V našem primeru je prostor iskanja sestavljen iz vseh *povezanih kubičnih grafov* na n vozliščih. Prehod med grafi definiramo s t. i. *dvojnim prevezovanjem robov* (ang. double-edge swap). Naj bo $G = (V, E)$ kubičen graf in naj bosta izbrani dve disjunktni povezavi $\{u, v\}, \{x, y\} \in E$. Zamenjava poteka tako, da se povezavi odstranita in nadomestita z novima paroma $\{u, x\}$ in $\{v, y\}$ ali s paroma $\{u, y\}$ in $\{v, x\}$. S takimi menjavami povezav ohranimo 3-regularnost grafa. Energijska funkcija, ki jo minimiziramo, pa je v našem primeru podana z $E(G) = \text{subpath\_number}(G)$, kjer funkcija $\text{subpath\_number}(\cdot)$ prešteje vse različne podpoti v grafu.

Proces začnemo z grafom, ki ga želimo izboljšati. V našem primeru je to graf $L_n =: G$. Pri vsakem koraku z double edge swap generiramo novega soseda G' in izračunamo $\Delta E = E(G') - E(G)$. Če je $\Delta E \leq 0$, rešitev sprejmemo. V nasprotnem primeru jo sprejmemo z verjetnostjo

$$p = \exp\left(-\frac{\Delta E}{T}\right),$$

kjer $T > 0$ predstavlja trenutni temperaturni parameter. S tem omogočimo kontrolirano sprejemanje slabših rešitev zgodaj v postopku, kar preprečuje prezgodnjo ujetost v lokalne minimume. Če rešitev sprejmemo nastavimo $G := G'$ in postopek ponavljamo.

V našem primeru smo naredili 20.000 ponovitev postopka, začetni parameter T_0 pa je bil nastavljen tako, da se je v zgodnjih ponovitvah slabša rešitev sprejela z verjetnostjo približno 40 %, proti koncu postopka pa skoraj nikoli.

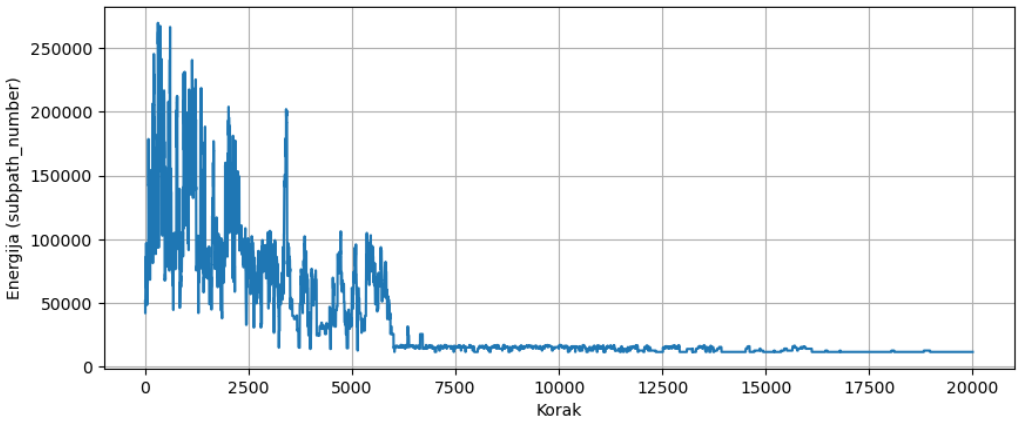
2.2 Rezultati

Z metodo *simulated annealing* smo za $n \in \{20,\ 22,\ \ldots,\ 30\}$ iskali grafe z manjšim številom podpoti od grafa L_n. Rezultati so v spodnji tabeli.

n	$subpath\_number(L_n)$	boljši $subpath\_number$	razlika
20	51.532	11.708	38.716
22	90.760	7.156	83.604
24	206.800	12.220	194.580
26	363.788	21.760	342.028
28	827.988	29.568	798.420
30	1.456.016	18.520	1.437.496

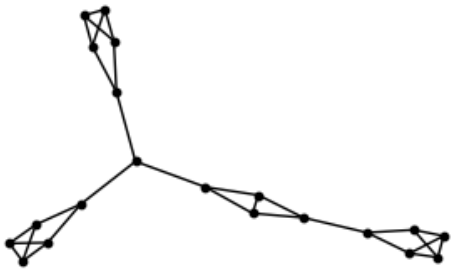
Tabela 3: Rezultati SA.

Ker smo našli grafe z manjšim številom podpoti, smo torej domnevo, da imajo grafi L_n najmanjše število podpoti, zavrnili tudi za $n \in \{20,\ 22,\ \ldots,\ 30\}$.



Slika 2: Potek energije med simulated annealing za $n = 20$.

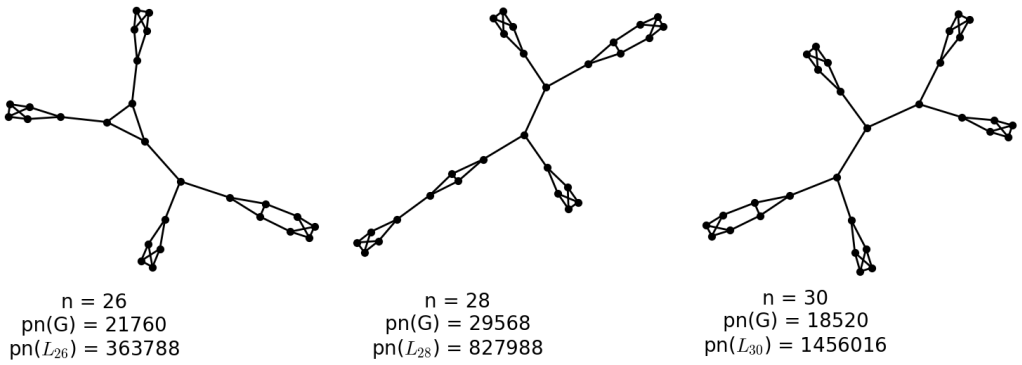
Na sliki lahko vidimo potek energije (števila podpoti) med procesom SA za $n = 20$. Kar lahko razberemo je, da so se med postopkom sprejele tudi slabše rešitve, saj energija (število podpoti) ni strogo padajoča. Program je uspešno sprejemal tudi sabše rešitve in se s tem izognil morebitnim lokalnim minimumom. Vselej pa smo na koncu našli graf s kar 5-krat manjšim številom podpoti ob grafa L_{20}. Čeprav samo na podlagi rezultatov SA še ne moremo trditi, da je 11.706 najmanjše število podpoti za $n = 20$, bomo kasneje dokazali, da je to pravzaprav res in da smo s postopkom SA našli graf, ki minimizira število podpoti za $n = 20$.



Slika 3: Graf dobljen s SA za $n = 20$.

Vredno je poudariti, da v splošnem z najdenimi grafi še zdaleč ne minimiziramo števila

podpoti za inzbrani n, čeprav so za nekatere n najdeni grafi že precej dobri. Našli smo le primere grafov z manjšim številom podpoti od opazovanega L_n grafa. To se dobro vidi, ko primerjamo dobljeno minimalno število podpoti za $n = 28$ in $n = 30$. Dobljeni graf za $n = 30$ ima namreč kar za tretjino manj podpoti od dobljenega grafa za $n = 28$. Ta podatek nam za graf z $n = 30$ sicer ne pove veliko, medtem ko smo lahko skoraj prepričani, da za $n = 28$ obstaja graf z manjšim številom podpoti.



Slika 4: Grafi pridobljeni s SA za $n = 26$, 28 in $n = 18$.

3 Rezultati boljših grafov

Število podpoti na novo konstruiranih grafov smo sedaj primerjali neposredno z grafi L_n. Rezultati izračunov za $n \in \{16, 18, \ldots, 40\}$ so predstavljeni v spodnji tabeli.

n	pn(L_n)	pn(Tree)	pn(Cat1)	pn(Cat2)
16	12.744	3.640	3.640	3.640
18	22.532	7.072	7.072	7.072
20	51.532	12.816	12.816	11.708
22	90.760	7.156	7.156	7.156
24	206.800	12.220	12.220	12.220
26	363.788	19.596	19.596	19.064
28	827.988	11.824	11.824	11.824
30	1.456.016	18.520	18.520	18.520
32	3.312.856	27.528	27.528	27.572
34	5.825.044	17.644	17.644	17.644
36	13.252.444	25.972	25.972	25.972
38	23.301.272	36.612	36.612	37.232
40	53.010.912	24.616	24.616	24.616

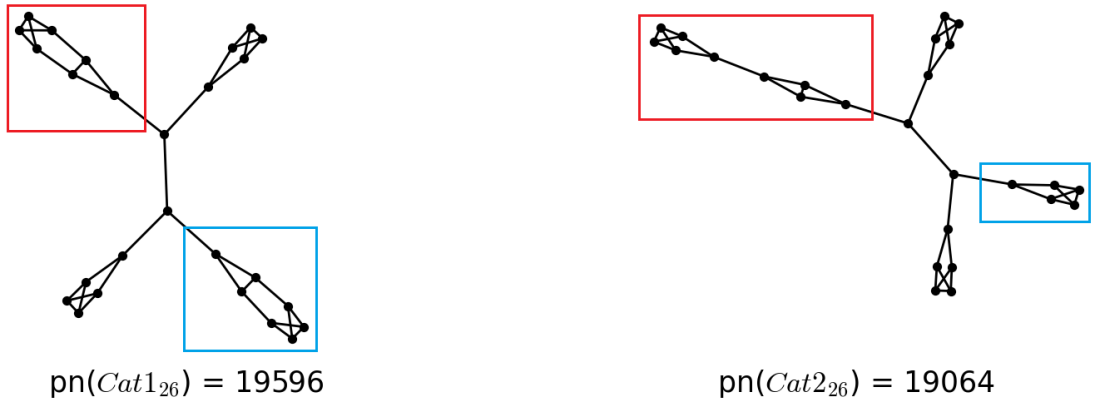
Tabela 4: Primerjava pn(L_n) s tremi različnimi konstrukcijami.

Hitro opazimo, da imajo vse tri konstrukcije veliko manjše število podpoti kot graf L_n. V primeru $n = 40$ imajo naše konstrukcije kar 2.153-krat manjše število podpoti kot graf L_{40}. Opazimo lahko tudi, da imata grafa *Cat1* in *Tree* isto število podpoti. Njun subpath number se namreč ujema vse do $n = 300$ (Testirali smo samo do $n = 300$). To glede na njuno konstrukcijo ni nič presenetljivega, saj ju sestavlja isto število glavnih gradnikov in tudi število povezav med posameznimi gradniki je enako.

VERJETNO POPRAVEK: GLEDE NA RAZDELEK O KONSTRUKCIJAH

Poleg vsega opazimo tudi razliko v številu podpoti med konstrukcijama *Cat1* in *Cat2*. Razlike se pojavijo pri $n = 6k + 2$ za $k = 2,\ 3,\ 4, \ldots$, kar bi lahko pričakovali, saj sta grafa različno

konstruirana ravno pri teh n. Kot vidimo na spodnji sliki, se konstrukciji pri $n = 26$ razlikujeta v treh ključnih gradnikih. Če izhajamo iz konstrukcije *Cat1* oz. *cat2* na 22 vozliščih (konstrukciji sta za $n = 22$ enaki), opazimo, da pri konstrukciji *Cat1* 4 nova vozlišča vpeljemo tako, da dva skrajna **pendant bloka?** zamenjamo s **podaljšanima?** blokoma. Pri konstrukciji *cat2* pa 4 nova vozlišča vpeljemo tako, da med pendant bloke vrninemo **srednji gradnik?**.



Slika 5: Konstrukciji *Cat1* (levo) in *Cat2* (desno) za $n = 26$.

Zanimivo je, da ni ena izmed konstrukcij zmeraj boljša. Za $n \in \{14,\ 20,\ 26\}$ ima namreč najmanjše število podpoti *Cat2*, medtem ko je za $n \in \{32,\ 38,\ 44, \dots\}$ boljši graf *Cat1*.

n	pn(Cat1)	pn(Cat2)
14	7.188	5.504
20	12.816	11.708
26	19.596	19.064
32	27.528	27.572
38	36.612	37.232
44	46.848	48.044
50	58.236	60.008
56	70.776	73.124
62	84.468	87.392
68	99.312	102.812
74	115.308	119.384
80	132.456	137.108

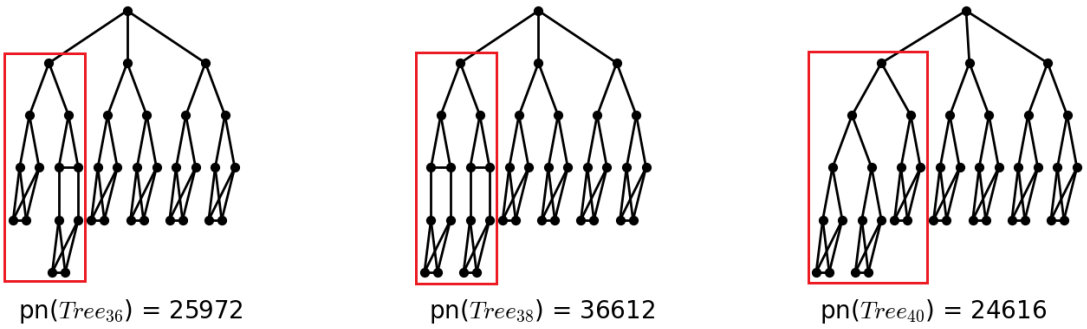
Tabela 5: Primerjava števila podpoti med strukturama Cat1 in Cat2.

Če podrobno pogledamo rezultate primerjave števila podpoti grafov L_n z našimi konstrukcijami opazimo, da prav za $n = 6k + 2$ za $k = 2,\ 3,\ 4, \dots$ število podpoti v naših konstrukcijah doživi velik narast in ima višjo vrednost kot njegov naslednik $n + 2$ ali celo $n + 4$. Ker se nam to ni zdelo smiselno, smo za $n = 20$ vrednost pn(Cat2), katera je za ta n med našimi konstrukcijami najnižja, primerjali z vsemi kubičnimi grafi na 20 vozliščih. Ugotovili smo, da graf *Cat2* med kubičnimi grafi na 20 vozliščih minimizira število podpoti. To pomeni, da glede tega skoka ne moremo narediti nič. To pomeni, da število podpoti ni povsem odvisno od števila vozlišč, pomembni so le posamezni gradniki znotraj grafa, katerih za $n = 6k+2$ ne moremo konstruirati.

n	pn(L_n)	pn(Tree)	pn(Cat1)	pn(Cat2)
16	12.744	3.640	3.640	3.640
18	22.532	7.072	7.072	7.072
20	51.532	12.816	12.816	11.708
22	90.760	7.156	7.156	7.156
24	206.800	12.220	12.220	12.220
26	363.788	19.596	19.596	19.064
28	827.988	11.824	11.824	11.824
30	1.456.016	18.520	18.520	18.520
32	3.312.856	27.528	27.528	27.572
34	5.825.044	17.644	17.644	17.644
36	13.252.444	25.972	25.972	25.972
38	23.301.272	36.612	36.612	37.232
40	53.010.912	24.616	24.616	24.616

Tabela 6: Skok števila podpoti za $n = 6k + 2$.

Za skok števila podpoti so zagotovo krivi **podaljšani** gradniki. V konstrukcijah *cat1* in *Tree*, ko iz $n = 6k$ preidemo v $n = 6k + 2$, dve dodatni vozlišči vpeljemo tako, da enemu podaljšanemu gradniku dodamo še drugega namesto enega **pendant bloka**. Tu se v smislu števila podpoti naredi največji skok. Ko pa n povečamo še za 2, torej $n = 6k + 4$, ta dva podlajšana gradnika zamenjamo s tremi pendant bloki, kar nam število podpoti ponovno zmanjša. Vse tri konstrukcije in njihove razlike so predstavljene na spodnji sliki.

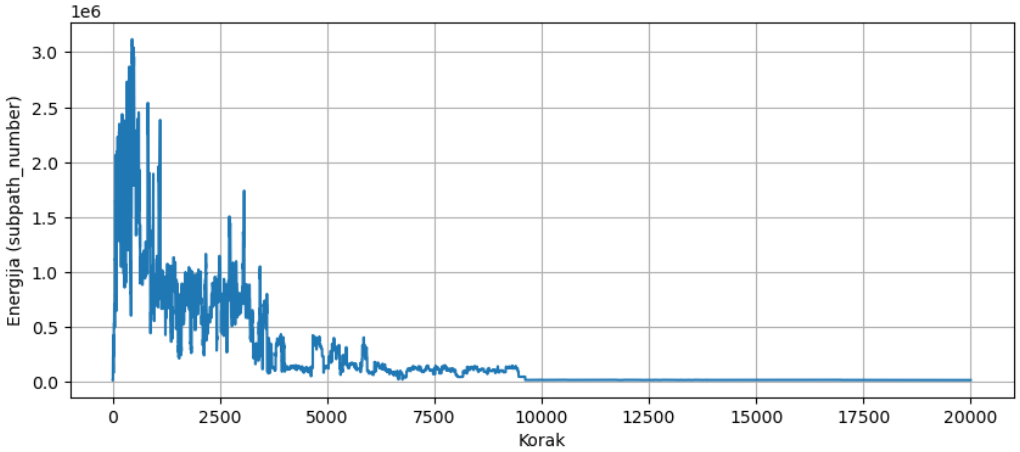


Slika 6: Razlike v konstrukcijah *tree* za $n = 36$ (levo), $n = 38$ (srednji) in $n = 40$ (desno).

3.1 Simulated annealing

Želimo vedeti, ali naše konstrukcije minimizirajo število podpoti za kubične grafe na n vozliščih. Ker smo za $n \in \{10,\ 12,\ 14,\ 16,\ 18,\ 20\}$ pregledali vse kubične grafe, lahko z gotovostjo trdimo, da konstrukcija *Cat2* minimizira subpath number za kubične grafe na n vozliščih za $n \in \{10,\ 12,\ 14,\ 16,\ 18,\ 20\}$.

Sedaj trdimo, da so naše konstrukcije tiste, ki minimizirajo število podpoti na kubičnih grafih. Podobno kot smo to domnevo raziskovali za L_n grafe, smo tudi za te grafe protiprimere iskali s *simulated annealing* metodo. Metodo smo preizkusili za $n \in \{22,\ 24,\ 26\}$, kjer smo prvič za začetni približek uporabili konstrukcijo *Cat1*, drugič pa še konstrukcijo *Tree*. V obeh primerih nam je metoda za $n = 22$ in $n = 24$ vrnila kar pn(Cat1) oziroma pn(Tree), prav tako pa je v primeru $n = 26$ našla nižji subpath number, kateri se ujema s pn(Cat1). Vse vrednosti so nam bile tako že znane, kar pomeni, da z metodo *simulated annealing* nismo našli kubičnih grafov na n vozliščih, z nižjim številom podpoti od naših konstrukcij za $n \in \{22,\ 24,\ 26\}$.



Slika 7: Potek energije med simulated annealing za $n = 26$ z začetnim približkom *Cat1*.

Kar bi nas pri tej ugotovitvi lahko skrbelo je, da se je algoritem SA zataknil v nekem lokalnem minimumu. Vendar iz grafa poteka energije na sliki lepo vidimo, da je algoritem obravnaval tudi druge grafe s precej višjim številom podpoti. V prvih 2.000 korakih je namreč sprejel približno 35 % vseh slabših rešitev, kasneje pa vedno manj. Vsekakor pa to ne pomeni, da lahko z gotovostjo trdimo, da naše konstrukcije minimizirajo število podpoti med kubičnimi grafi, saj bi morali za to obravnavati čisto vse grafe, kot smo jih za $n \leq 20$.