Projektna naloga

PROBLEM NA NAPAKE ODPORNE METRIčNE DIMENZIJE

Anamarija Potokar, Hana Samsa

Mentorja: doc. dr. Janoš Vidali,
prof. dr. Riste Škrekovski

Fakulteta za matematiko in fiziko

December 2024

1 Na napake odporna metrična dimenzija

Množica $S \in V$ v grafu G je razrešljiva, če za vsak par vozlišč $x, y \in V(G)$ ostaja vozlišče $s \in S$, da velja $d(x, s) \neq d(y, s)$. Rečemo, da sta x in y razrešeni z vozliščem s. Množica S je odporna na napake, če je $S \setminus \{v\}$ prav tako razrešljiva za vsak $v \in S$.

Metrična dimenzija neusmerjenega in povezanega grafa $G = (V, E)$ je moč najmanjše podmnožice nabora vozlišč $S \subset V$ z lastnostjo, da so vsa vozlišča v V enolično določena z njihovimi razdaljami do vozlišč podmnožice S.

Na napake odporna metrična dimenzija grafa G, je velikost najmanjše razčlenujoče množice S, odporne na napake in jo označimo z $ftdim(G)$.

Naloga projektne naloge je bila, da s pomočjo celoštevilskega linearnega programa poiščemo grafe z $dim(G) = 2$ in $ftdim(G) = 5, 6, 7$ ali več. Pri tem se za manjše grafe, torej grafe z malo vozlišči, uporablja sistematično iskanje (ang. *systematic search*), za večje grafe pa metahevristični pristop (ang. *simulated annealing search*).

2 Celoštevilski linearni program

Imamo povezan in neusmerjen graf $G = (V, E)$, kjer je $V = \{1, 2, \ldots, n\}$ množica vozlišč in $\mid E \mid = m$. Naj bo $d(u, v)$ najkrajša pot med vozliščema $u, v \in V$.

V naslednjem celoštevilskem lineearnem programu naj velja $1 \leq u < v \leq n$ in $1 \leq i < j \leq n$. Najprej definirajmo matriko koeficientov A. Spremenljivka x_i, definirana s predpisom (2), nam pove, ali vozlišče i pripada množici S.

$$A_{(u,v),i} = \begin{cases} 1, d(u,i) \neq d(v,i) \\ 0, d(u,i) = d(v,i) \end{cases} \tag{1}$$

$$x_i = \begin{cases} 1, i \in S \\ 0, i \notin S \end{cases} \tag{2}$$

$$\min \sum_{i=1}^{n} x_i \tag{3}$$

$$\sum_{i=1}^{n} A_{(u,v),i} \cdot x_i \geq 2, \ 1 \leq u < v \leq n \tag{4}$$

$$x_i \in \{0, 1\}, \ 1 \leq i \leq n \tag{5}$$

Enačba (3) predstavlja najmanjšo podmnožico S. Enačba (4) pa nam da pogoj, da je poodmnožica S razrešljiva množica, odporna na napake. To pomeni, da če za vozlišči i velja $d(u,i) \neq d(v,i)$ in je hkrati $i \in S$, bo potem vsota enaka 2, kar pa je ravno to, kar iščemo.

3 Potek dela

V prvi fazi projekta sva identificirali grafe z določenimi lastnostmi glede na njihovo metrično dimenzijo in na napake odporno metrično dimenzijo za različna števila vozlišč. V najini kodi sva na začetku napisali funkciji, ki za povezan graf G izračunata njegovi dim in $ftdim$. Funkciji delujeta tako, da za graf G preštejeta število vozlišč in izračunata medsebojne razdalje za poljuben par, sestavita koeficientno matriko, rešita ustrezen CLP in vrneta moč razrešljive množice. Dodatno sva definirali še funkcijo, ki za željeni dim in $ftdim$ izračuna minimalno potrebno število vozlišč grafa, ki izpolnjuje ta pogoja. V teoriji je sicer najmanjše potrebno število vozlišč enako kar $ftdim$, vendar pa je v praksi to število ponavadi (dosti) večje. Funkciji sva nato uporabili za iskanje ustreznih grafov, izpisali njihovo število pri dani dim, $ftdim$ in številu vozlišč ter jih izrisali.

Sistematično iskanje je bilo zaradi časovne zahtevnosti za večje grafe in višje $ftdim$ zelo omejeno, zato sva v drugi fazi za iskanje grafov uporabili metahevristični algoritem *simulated annealing* oziroma *simulirano ohlajanje*, preizkusili pa sva tudi *hill climbing*. To sta vrsti algoritma v stohastični optimizaciji, ki uporabljata naključnost za iskanje čim boljših rešitev danega problema v razumnem času, še posebej, kadar je popolna rešitev zaradi kompleksnosti problema težko dosegljiva.

Za definicijo funkcije simuliranega hlajenja sva najprej implementirali funkcijo, ki grafu naključno odstrani ali doda povezavo. To sva nato uporabili za primerjavo z izhodiščnim grafom, in če je bil prilagojen graf glede na njegovi dim in $ftdim$ bližje iskani rešitvi, sva ga sprejeli, sicer pa se vrnili na izhodiščni graf. Ko je zanka pretekla največje število iteracij, določeno z argumentom funkcije, se je prekinila, funkcija pa je izrisala najboljši najden graf ter njegovi dim in $ftdim$.

4 Koda

Komentirana koda je dostopna na povezavi.

5 Sistematično iskanje

V prvi fazi sva se iskanja ustreznih grafov z lastnostima dim(G)= 2 in ftdim(G) = 5, 6, 7, ... lotili tako, da sva za konstantno vrednost dim postopoma povečevali željeno ftdim in število vozlišč, za katerega iščemo ustrezne grafe. Najprej sva kodo preizkusili za vrednosti dim = 2 in ftdim = 4: V prvi fazi sva se iskanja ustreznih grafov z lastnostima $dim(G) = 2$ in $ftdim(G) = 5, 6, 7, ...$ lotili tako, da sva za konstantno vrednost dim postopoma povečevali željeno ftdim in število vozlišč, za katerega iščemo ustrezne grafe. Najprej sva kodo preizkusili za vrednosti $dim = 2$ in $ftdim = 4$:

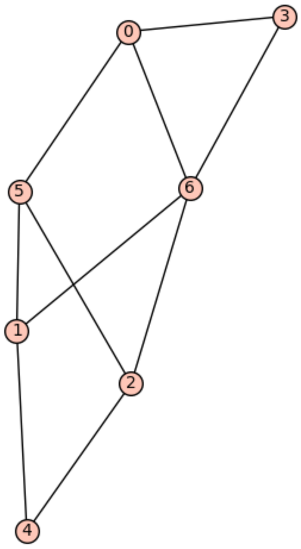
- na 4 vozliščih obstajata 2 taka grafa, čas izvajanja kode je 0.04 sekunde,
- na 5 vozliščih obstaja 8 takih grafov, čas izvajanja kode je 0.11 sekunde,
- na 6 vozliščih obstaja 46 takih grafov, čas izvajanja kode je 0.69 sekunde,
- na 7 vozliščih obstaja 232 takih grafov, čas izvajanja kode je 7.19 sekund,
- na 8 vozliščih obstaja 1525 takih grafov, čas izvajanja kode je 2 minuti in 1 sekunda.

Rezultate za $ftdim = 4$ ponazorimo še s tabelo:

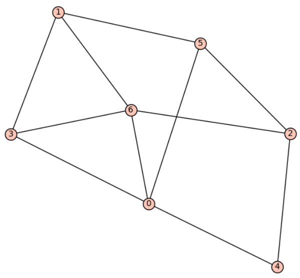
št. vozlišč	**št. grafov**	**čas izvajanja**
4	2	0.04s
5	8	0.11s
6	46	0.69s
7	232	7.19s
8	1525	2min1s

Tabela 1: Rezultati za $ftdim = 4$

Nato sva se lotili iskanja odgovora na vprašanje naloge: za $ftdim(G) = 5$ sva najprej ugotovili, da za manj kot 7 vozlišč tak graf sploh ne obstaja. Na 7 vozliščih obstajata dva ustrezna grafa, katera je algoritem našel in izrisal v 5.96 sekundah. Ta sta prikazana na slikah 1 in 2.

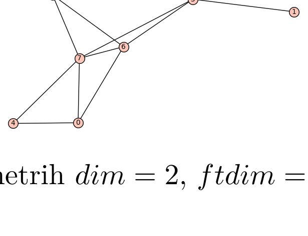


Slika 1: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 7.

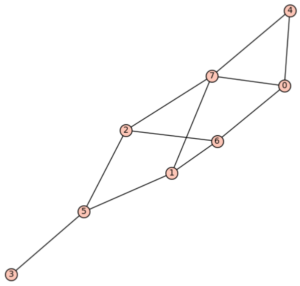


Slika 2: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 7.

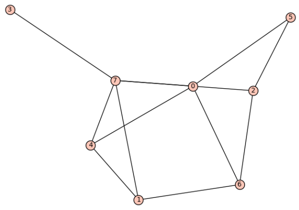
Na 8 vozliščih obstaja že bistveno več grafov, za katere velja $dim(G) = 2$ in $ftdim(G) = 5$, in sicer 65, tudi koda pa v primerjavi z grafi na 7 vozliščih rabi veliko dlje časa, da se izvede; ustrezne grafe je poiskala in izrisala v 1 minuti in 58 sekundah. Nekateri izmed teh so prikazani v slikah, ki sledijo.



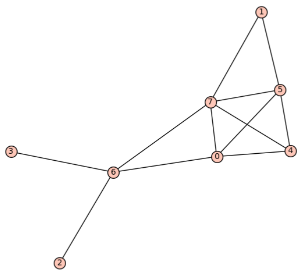
Slika 3: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 8.



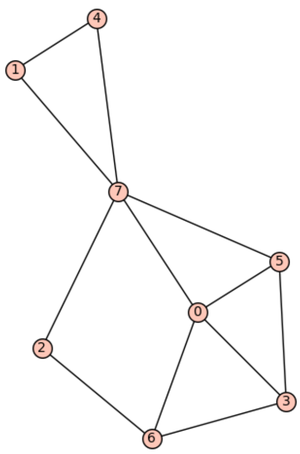
Slika 4: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 8.



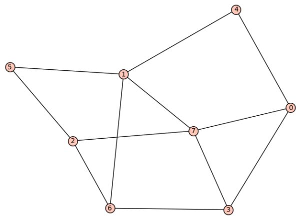
Slika 5: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 8.



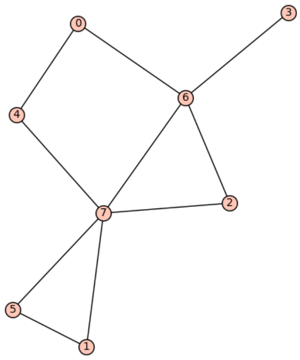
Slika 6: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 8.



Slika 7: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 8.

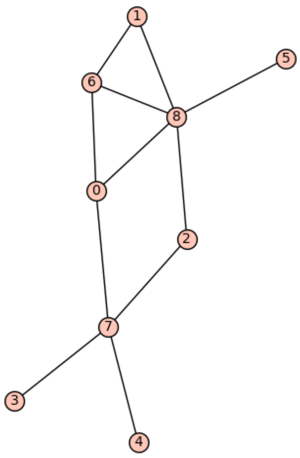


Slika 8: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 8.

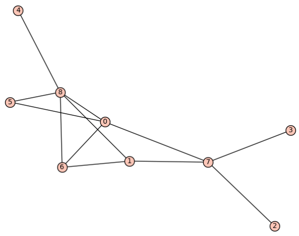


Slika 9: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 8.

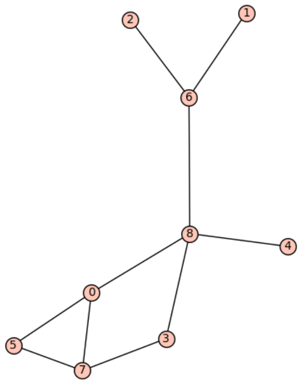
Pri iskanju in risanju takih grafov na 9 vozliščih se je koda po pol ure dela prenehala izvajati in ni našla vseh ustreznih grafov. Trije izmed teh so prikazani na slikah 10, 11 in 12.



Slika 10: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 9.



Slika 11: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 9.



Slika 12: Graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 9.

Za več grafov lahko na povezavi poženete datoteko z imenom *Exaustive search.ipynb*. Rezultate za $ftdim = 5$ ponazorimo še s tabelo:

št. vozlišč	**št. grafov**	**čas izvajanja**
7	2	5.96s
8	65	1min58s

Tabela 2: Rezultati za $ftdim = 5$

Očitno se torej že pri $ftdim = 5$ in 9 vozliščih, kar se zdi dokaj malo, koda ne izvede v sprejemljivem času. Preverili sva še, kako časovno zahtevno je iskanje grafov z $dim = 2$ in $ftdim = 6$, za kar je potrebnih najmanj 12 vozlišč; tudi ta koda se ni izvedla v normalnem času, zato sva za primere pri $ftdim = 5$ in več kot 8 vozlišč ter $ftdim = 6, 7, ...$ grafe iskali s pomočjo metahevrističnih algoritmov.

6 Metahevristično iskanje

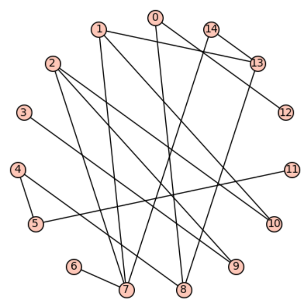
6.1 Simulirano ohlajanje

V naslednji fazi sva želeli narediti kodo, ki bi poiskala tudi večje grafe. To sva dosegli s metahevrističnim pristopom oziroma s simuliranim ohlajanjem. Tako kot v prvi fazi sva kodo najprej preizkusili za $dim = 2$ in $ftdim = 4$. Pri tem sva za grafe vzeli že večje grafe s številom vozlišč od 9 do 15, poskusili pa sva tudi za grafe z 20 vozlišči. Potem sva postopoma povečevali ciljno na napake odporno metrično dimenzijo.

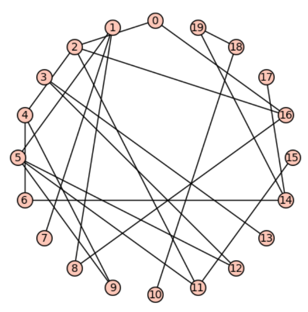
Pri simuliranem ohlajanju se je pojavila težava pri kodah metricna_dolzina in na_napake_odporna_metricna_dolzina, zato sta ti kodi drugačni kot pri sistematičnem iskanju.

Ta pristop je definiran tako, da vrne najboljši približek grafa pri danih dim in $ftdim$. S tem pristopom lahko mnogo hitreje poiščemo tudi grafe na veliko vozliščih in večjimi na napake odpornimi metričnimi dimenzijami.

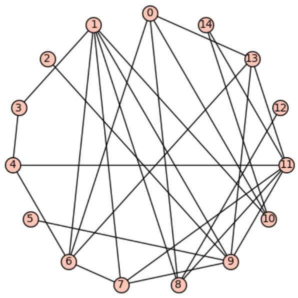
Nekateri izmed poiskanih grafov so prikazani v nadaljevanju.



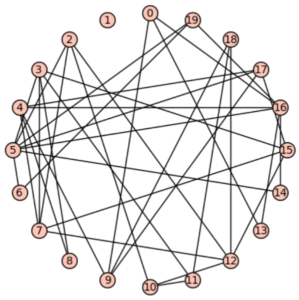
Slika 13: Najboljši graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 15.



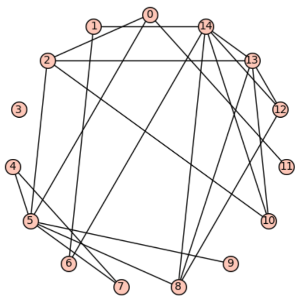
Slika 14: Najboljši graf pri parametrih $dim = 2$, $ftdim = 5$, in število vozlišč je 20.



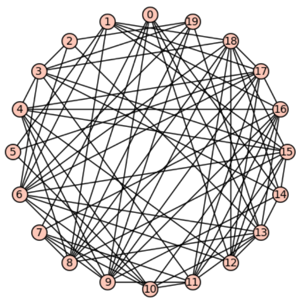
Slika 15: Najboljši graf pri parametrih $dim = 2$, $ftdim = 6$, in število vozlišč je 15.



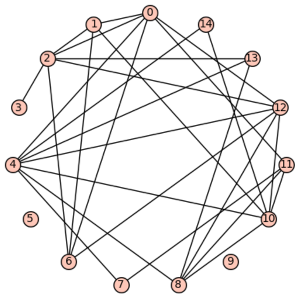
Slika 16: Najboljši graf pri parametrih $dim = 2$, $ftdim = 6$, in število vozlišč je 20.



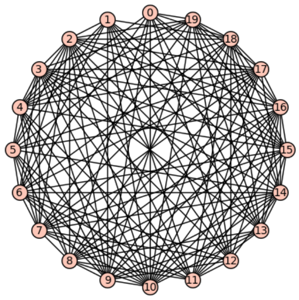
Slika 17: Najboljši graf pri parametrih $dim = 2$, $ftdim = 7$, in število vozlišč je 15.



Slika 18: Najboljši graf pri parametrih $dim = 2$, $ftdim = 7$, in število vozlišč je 20.



Slika 19: Najboljši graf pri parametrih $dim = 2$, $ftdim = 8$, in število vozlišč je 15.



Slika 20: Najboljši graf pri parametrih $dim = 2$, $ftdim = 8$, in število vozlišč je 20.

6.2 Hill climbing

Kot dodatek sva poizkusili večje grafe z določeno na napake odporno metrično dimenzijo poiskati tudi z algoritmom Hill climbing. To je algoritem, ki postopoma izboljšuje graf tako, da iz trenutne rešitve ustvari novo, nekoliko spremenjeno različico (sosednjo rešitev oziroma graf). Če je nova rešitev boljša, jo sprejme in nadaljuje postopek, dokler ne doseže cilja ali največjega števila iteracij.

Razlika med hill climbing in simuliranim ohlajanjem je v odločanju, kdaj zamenjati začetnega kandidata za rešitev s prilagojenim grafom. Namreč, če je prilagojen graf boljši kot začetni graf, bomo vedno zamenjali začetnega s prilagojenim grafom. Če pa je prilagojeni slabši od začetnega, pa ju bomo pri simuliranem ohlajanju morda še vedno zamenjali z neko določeno verjetnostjo, medtem ko pri hill climbingu tega ne bomo storili.

7 Ugotovitve

Najina pričakovanja so bila, da grafov z na napako odporne metrično dimenzijo ni veliko, prav tako pa, da jih je težko poiskati. Izkazalo se je, da to ni čisto res. Namreč, ko sva večali število vozlišč, je s tem število ustreznih grafov zelo hitro naraščalo, hkrati pa je hitro naraščala tudi časovna zahtevnost najdbe vseh takih grafov. S sistematičnim iskanjem je to res nekoliko počasneje, vendar je tudi zelo učinkovito, saj preveri vse možne grafe in vrne točne rešitve.

S simuliranim ohlajanjem je postopek iskanja hitrejši in omogoča preiskovanje večjih grafov in višjih na napake odpornih metričnih dimenzij, ampak skoraj nikoli ne pridemo do optimalne rešitve, saj koda vrne najboljši približek grafu.

Domnevali sva, da je lažje poiskati grafe, če je razlika med metrično dimenzijo in na napake odporno metrično dimenzijo manjša. Izkazalo se je, da to drži, kar je razvidno iz samega časa delovanja kode. Prav tako pa je čas delovanja kode krajši, če gre za manjše grafe oziroma grafe z manj vozlišči ali pa z nižjo zahtevano na napake odporno metrično dimenzijo.

8 Literatura in repozitorji

A. Simić, M. Bogdanović, Z. Maksimović, J. Milošević. Fault-tolerant metric dimension problem: A new integer linear programming formulation and exact formula for grid graphs. Kragujevac Journal of Mathematics, 2018.

Repozitorjji je dostopen na:
`https://github.com/HanaSamsa/Problem-metricnih-dimenzij-odpornih-na-napake/tree/main`