Univerza v Ljubljani
Fakulteta za matematiko in fiziko

Predstavitev 4-ciklov v snarkih

Domen Skornšek, Žan Luka Kolarič

Mentorja: Janoš Vidali, Riste Škrekovski

Predmet: Finančna matematika

Ljubljana, 2024

1 Uvodna beseda

Cilj najine naloge je bil ugotoviti, kako uvedba 4- ciklov vpliva na kromatični indeks snarkov. Uporabila sva dve metodi, ki sta v snark dodali 4-cikel (več o metodah v nadaljevanju). Več o algoritmih (programih) sva zapisala v nadaljevanju poročila.

2 Snarki, ki sva jih vključila v nalogo

- celmins swart snark 1 (26 vozlišč)
- celmins swart snark 2
- double star snark (30 vozlišč)
- flower snark j7 (28 vozlišč)
- goldberg snark 3 (24 vozlišč)
- goldberg snark 5 (40 vozlišč)
- loupekines snark 1 (22 vozlišč)
- loupekines snark 2 (22 vozlišč)
- snark_2760
- snark_3337
- snark_3363
- graf_25159

3 METODA 1

Pri metodi 1 sva vzela 2 robova, to sta ab in cd ter ju dvakrat razdelila (dodana vozlišča u1, u2, v1, v2), tako da sva dobila pot au1u2b in cv1v2d. Na koncu sva povezala vozlišča u1 z v1 ter u2 z v2. Tukaj naju predvsem zanima ali se kromatični indeks kdaj spremeni na 3 ter zakaj pride do tega, pa tudi ali se lahko še kdaj vrne nazaj na 4.
Za začetek sva napisala program, ki ponavlja zgoraj opisano metodo (50x ponovitev), na vsakem koraku pa nama izpisuje nov kromatični indeks ter povezavi, na kateri so se dodala nova vozlišča.

Drugi algoritem, ki sva ga zapisala, je algoritem, ki deluje zelo podobno, le da sva onemogočila dodajanje novih vozlišč na dve zaporedni povezavi. Več o tem v ugotovitvah.
Za konec sva tukaj napisala še algoritem, ki k-krat ponovi le prvo iteracijo. S tem sva želela ugotoviti, v kakšnem deležu se kromatični indeks ohrani.

4 UGOTOVITVE ZA METODO 1

Za preverjanje oz. implementacijo metode sva vzela več različnih snarkov, da bi bil vzorec čim bolj reprezentativen.
Funkcija $algoritem\_1\_iteracija$ omogoča iterativno spreminjanje grafa (snarka) z dodajanjem 4-ciklov in spremljanje sprememb kromatičnega indeksa grafa. Poleg tega za vsako iteracijo beleži uporabljene robove, ki so bili vključeni v modifikacijo grafa. Tu je glavna ugotovitev ta, da se kromatični indeks zmanjša iz 4 na 3 v primeru, ko se vozlišča dodajo na dve zaporedni povezavi. Česa drugega tukaj nisva opazila.
Kot že rečeno sva zato napisala modificiran algoritem pri katerem sva izključila možnosti, da se vozlišča dodajajo na dve zaporedni povezavi. Pri tem sva ugotovila, da se kromatični indeks lahko obdrži na 4 tudi v primeru ko se vozlišča ne dodajo na dva zaporedna robova, ampak nisva prepričana, zakaj bi do tega lahko prišlo. Sklepava, da ima tukaj glavno vlogo to, da se vozlišča dodajo na povezave, ki sestavljajo najkrajši možen cikel snarka oz. ena izmed povezav je vedno iz najkrajšega cikla.
Za konec metode 1 pa še, v koliko procentih se indeks ohrani pri 200 ponovitvah prve iteracije. Podatki kažejo, da se to pri nekaterih grafih zgodi z zelo majhno verjetnostjo, spet pri drugih pa dokaj pogosto. Povprečje je nekje pri 8%.

5 METODA 2

Pri metodi 2 sva vzela povezavo ab iz grafa G in za vozlišci a in b poiskala njune sosede a_1, a_2 ter b_1, b_2. Odstranila sva vozlišči a, b in dodala nova vozlišča a'_1, a'_2, b'_1, b'_2. Nato sva dodala povezave $a_1a'_1, a_2a'_2, b_1b'_1, b_2b'_2$ in povezala vozlišča a'_1, a'_2, b'_1, b'_2 v naključni 4-cikel. Zanima naju ali se kdaj spremeni kromatičen indeks na 3 in zakaj pride do tega. Prav tako naju zanima, če se lahko indeks vrne nazaj na 4, ko je enkrat padel na 3.

Najprej sva napisala $algoritem\_2(snark)$, ki uvede 4-cikel na snarku z

uporabe metode 2 na naključni povezavi. Nato sva, da bi lažje preučila ohranjanje indeksa zapisala $ponavljaj\_algoritem\_2(snark, k)$, ki na snarku k-krat požene algoritem 2. Ker sva ugotovila, da se indeks ohrani v malo primerih sva napisala $ponavljaj\_algoritem\_2\_samo\_4(snark, k)$, ki izpiše samo povezave, kjer se indeks ohrani. Za konec sva napisala še iteracijsko kodo za $algoritem\_2(snark)$ in pa iteracijsko kodo, ki naključno izbira med 1. in 2. metodo.

6 UGOTOVITVE ZA METODO 2

Pri vpeljavi 4-cikla z uporabo metode 2, sva se najprej osredotočila na to, kdaj se kromatičen indeks ohrani v 1. vpeljavi cikla. Za to sva najprej na vseh snarkih uporabila algoritem $ponavljaj\_algoritem\_2\_samo\_4(snark, k)$. Ta algoritem nama je izpisal vse možne povezave, kjer se indeks ohrani v k-tih poskusih. Na vseh grafih sva ga pognala 200 krat in ugotovila, da se indeks lahko ohrani le pri goldberg snark 3, pri snarku 3363 in pri grafu 25159. Pri Goldberg snarku se ohrani pri izbiri treh povezav, pri snarku 3363 pri izbiri dveh povezav, pri grafu 25159 se pa ohrani ne glede na to katero povezavo izberemo.

Zdaj sva si za cilj zadala poiskat vzorec, kdaj se kromatičen indeks ohrni. Za goldberg snark 3 sva ugotovila, da se indeks ohrani v primeru, da izbereva povezavo iz njegovega najkrajšega cikla. Ker je najkrajši cikel tega snarka enak 3 sva takoj posumila, da mogoče pa pravilo za ohranjanje le ni tako lahko. Žal to pravilo pri snarku 3363 ne velja. Poskusila sva z iskanjem vzorca tako, da sva preverjala lastnosti originalnega grafa in grafa ko se vzorec spremeni oz. ne spremeni. Takšnega vzorca nisva našla. Nato sva poskusila z cikli, vendar tudi tukaj neuspešno. Žal pravila za obstoj kromatičnega indeksa v prvem koraku metode 2 nisva odkrila.

Naslednji cilj je bil, da ugotoviva, ali lahko indeks ostane 4 tudi v 2., 3., itd. koraku. Za preverjanje sva si izbrala snark 3363. Ugotovila sva, da bo, če izbereva katerokoli povezavo iz novo ustvarjenega 4-cikla ali neuporabljeno povezavo iz prejšnjega koraka, indeks ostal 4. Prav tako se v vsakem koraku št. povezav za ohranitev poveča za 3, kar je enako številu $novepovezave - 1$.

Zadnji cilj je pa bil odkriti, ali se lahko kromatični indeks kdaj vrne na 4, če je že padel na 3. Najprej sva v prvem koraku vzela povezavo snarka 3363, da se je indeks spremenil. Tukaj sva izvedla 300 ponovitev algoritma 2 in ugotovila, da se v nobenem primeru ne vrne na 4. Za konec sva zapisala še

iteracijski algoritem, s pomočjo katerega sva izvedla 100 iteracij algoritma 2 in ugotovila, da se indeks res nikoli ne vrne na 4.

Za konec pa sva naredila še algoritem, ki naključno izvaja 1. in 2. metodo. S pomočjo tega sva še dodatno potrdila, da se indeks ne vrne na 4 ne glede na to, kateri algoritem uporabiva.