Univerza v Ljubljani
Fakulteta za matematiko in fiziko

Krepka liha barvanja grafov

Živa Artnak, Tilen Šlibar in Živa Kurež

Januar 2025

1 Uvod

Tema našega projekta so krepka liha barvanja grafov. Projektna naloga bo izvedena v programu SageMath, za osnovno literaturo pa bomo uporabili članek On strong odd colorings of graphs [1].

2 Osnovne definicije

Definicija 2.1. Graf je k-***obarvljiv***, če obstaja taka preslikava

$$c : V(G) \to \{1, \ldots, k\}$$

da velja $c(u) \neq c(v)$ za poljubni sosednji točki u in v grafa G. Tako preslikavo imenujemo ***barvanje*** oziroma k-***barvanje*** grafa. ***Kromatično število*** $\chi(G)$ grafa G je najmanjše število k, za katero je G k-obarvljiv.

Definicija 2.2. Barvanje grafa je ***liho barvanje***, če gre za pravilno barvanje, kjer se v sosedstvu vsakega vozlišča neka barva pojavi liho mnogo krat.

Definicija 2.3. ***Krepko liho barvanje*** preprostega grafa G je pravilno barvanje vozlišč G, tako da za vsako vozlišče v in vsako barvo c velja da se bodisi barva c pojavi liho mnogo krat v odprtem sosedstvu $N_G(v)$, bodisi noben sosed v ni obarvan s c.

Krepko liho kromatično število grafa G je najmanjše število barv, potrebnih za takšno barvanje. Označimo ga z $\chi_{so}(G)$.

3 Naloge projekta

Naloge v tem projektu so naslednje:

1. Napisati postopek, ki določi $\chi_{so}(G)$ za dani graf G.
2. Določiti, kdaj ima enociklični graf krepko liho kromatično število enako 1, 2, 3 ali 4. Znano je, da je za enociklične grafe, ki niso C_5, krepko liho kromatično število omejeno z 4.
3. Generirati zunajplanarne grafe in preučiti, ali imajo kateri od njih krepko liho kromatično število večje od 7.
4. Poiskati grafe, za katere je vrednost $\chi_{so}(G)/\Delta^2(G)$ čim večja (pri čemer je $\Delta(G)$ maksimalna stopnja grafa).
5. Pokazati veljavnost neenačbe:
$$\chi_{so}(G \circ H) \leq \chi_{so}(G)\chi_{so}(H)$$
za $\circ \in \{\times, \square, \boxtimes\}$ (tj. za direktni, kartezični in krepki produkt grafov) in najti čim več grafov, pri katerih velja enakost.

3.1 Točka 1

Pri prvi točki je bila naša naloga napisati postopek, ki določi $\chi_{so}(G)$ za dani graf G. Tega smo se najprej lotili s funkcijo `strong_odd_chromatic_number.ipynb`, ki število poišče po principu backtrakinga. Funkcija vrne pravilno rešitev, vendar pa je čas izvajanja za večje število vozlišč zelo dolg, zato smo napisali še eno funkcijo, ki pa deluje na podlagi naslednjega CLP:

Imejmo graf $G = (V, E)$ z $n = |V|$ vozlišči.

Spremenljivke:
x_{ui}: Vozlišče u je barve i ($x_{ui} \in \{0, 1\}$)
y_{ui}: Ali se barva i pojavi med sosedi vozlišča u ($y_{ui} \in \{0, 1\}$)
z_{ui}: Pomožni števec za barvo i v soseščini vozlišča u (celo število)
t: Števec barv

Ciljna funkcija:
$\min t$

Pogoji:
1. Vsako vozlišče je natanko ene barve:$\forall u \in V : \sum_i x_{ui} = 1$
2. Sosedi so različnih barv: $\forall uv \in E, \forall i : x_{ui} + x_{vi} \leq 1$
3. Štetje barv: $\forall u \in V, \forall i : i \cdot x_{ui} \leq t$
4. Katere barve se pojavijo med sosedi: $\forall u \in V, \forall i : \sum_{v \sim u} x_{vi} \leq n \cdot y_{ui}$
5. Barve, ki se pojavijo, se morajo pojaviti liho mnogo krat: $\forall u \in V, \forall i : \sum_{v \sim u} x_{vi} = y_{ui} + 2 \cdot z_{ui}$

Opomba: Spremenljivka i gre od 1 do izbranega števila barv. Čeprav lahko vzamemo n barv, je hitreje začeti z manjšim številom in ga povečevati, dokler program ne vrne rešitve.

Funkcija, ki deluje na principu tega CLP se nahaja v datoteki `strong_odd_chromatic_number_ILP.ipynb`

3.2 Točka 2

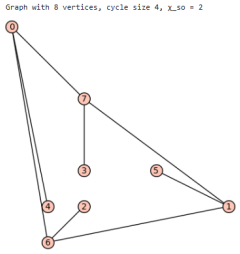
V drugi točki smo želeli določiti, kdaj ima enociklični graf krepko liho kromatično število enako 1, 2, 3 ali 4. Znano je (glej članek [1]) , da je za enociklične grafe, ki niso C_5, krepko liho kromatično število omejeno z 4. Prav tako pa vemo, da za cikle velja:

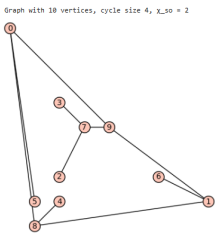
$$\chi_{so}(C_n) = \begin{cases} 3, & \text{če } 3 \mid n, \\ 4, & \text{če } 3 \nmid n \text{ in } n \neq 5, \\ 5, & \text{če } n = 5. \end{cases}$$

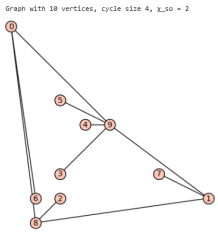
V datoteki `unicyclic_improved.ipynb` je funkcija, ki generira vse grafe na do n vozliščih, za katere velja pogoj, da imajo število povezav enako številu vozlišč, kar so natanko vsi enocilkični grafi na n vozliščih. Tem grafom nato izračuna krepko liho kromatiščo število in nam vrne tabelo, v kateri so rezultati urejeni glede na število vozlišč, dolžino cikla in krepko liho kromatično število. Prav tako nam tiste grafe, ki imajo krepko liho kromatično število nižje od 4 tudi izriše. Funkcijo nam je uspelo pognati na do 10 vozliščih, naše ugotovitve pa so naslednje:

- Noben graf nima krepkega lihega kromatičnega števila 1.
- Krepko liho kromatično število 2 se pojavi pri treh grafih. Kar je tem grafom skupno je to, da imajo cikel sode dolžine, vsako vozlišče v ciklu pa ima še liho število dodatnih sosedov, ki niso del cikla. Ti dodatni sosedi imajo lahko nato še svoje dodatne sosede, vendar mora biti stopnja vsakega vozlišča v grafu liha. Predpostavljamo torej, da lahko krepko liho kromatično število 2 najdemo zgolj pri enocikličnih grafih, ki izpolnjujejo naslednje pogoje:
  - imajo cikel sode dolžine (nujno, da je graf dvodelen)
  - stopnja vsakega vozlišča v grafu je liha

Posledično je to možno samo pri grafih, pri katerih je skupno število vozlišč vsaj dvakratnik dolžine cikla (saj moramo vsakemu vozlišču v ciklu dodati vsaj še enega soseda izven cikla, da bo število sosedov liho), poleg tega pa je skupno število vozlišč sodo.







- Ostali enocikliči grafi (razen C_5) pa imajo krepko liho kromatično število 3 ali 4. Tukaj nam natančnega pravila, kdaj je število enako 3 in kdaj 4 ni uspelo določiti, smo pa opazili naslednje:
  - Kot v trditvi v literaturi, imajo cikli (torej grafi brez dodatnih vozlišč, ki niso del cikla) dolžine deljive s tri krepko liho kromatično število 3, drugi cikli pa imajo krepko liho kromatično število 4.
  - V grafih, ki imajo cikel dolžine za eno manj kot število vseh vozlišč (torej cikli z enim dodatnim vozliščem izven cikla) imajo vedno krepko liho kromatično število 4. To je smiselno, saj če je imel cikel prej krepko liho kromatično število 4, eno dodatno vozlišče ni dovolj, da bi ga znižal na 3, če pa je imel prej krepko liho kromatično število 3, pa bomo za barvanje dodatnega vozlišča morali uporabiti novo, četrto barvo.
  - Enociklični grafi, v katerih je cikel dolžine 3, imajo lahko krepko liho kromatično število 3 zgolj, če je število dodatnih sosedov vozlišč v ciklu (torej sosedov, ki niso del cikla) sodo ali 0 (ti sosedi imajo lahko nato še dodatne sosede). Torej, mora stopnja vsakega vozlišča, ki je del tricikla ostati soda.

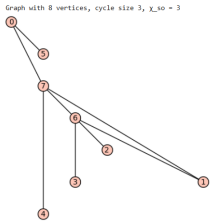


Figure 1: Primer grafa s triciklom z $\chi_{so} = 3$

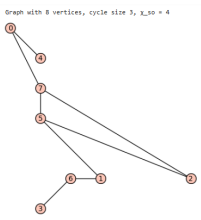


Figure 2: Primer grafa s triciklom z $\chi_{so} = 4$

Drugih povezav med strukturo enocikličnega grafa in njegovim krepkim lihim kromatičnim številom nam ni uspelo najti. Grafov za več kot 10 vozlišč nam ni uspelo preveriti, saj je že pri 11 vozliščih čas izvajanja kode presegel 4 ure in smo zato izvajanje ustavili.

Celotna tabela, ki jo vrne izvedba funkcije na 10 vozliščih in izris vseh grafov, s krepkim lihim kromatičnim številom, manjšim od 4 se nahaja v datoteki `unicyclic_improved.ipynb`

3.3 Točka 3

Definicija 3.1. Graf je ***zunanje ravninski***, če ga v ravnino lahko narišemo tako, da se nobeni dve povezavi ne sekata (je ravninski) in so pri tem vsa vozlišča na robu zunanjega lica.

V tej točki smo generirali zunanje ravninske grafe in poskušali med njimi najti kakšnega s krepkim lihim kromatičnim številom večjim od 7. Za nižje število vozlišč smo uporabili funkcijo `outerplanar.ipynb`, ki najprej generira vse možne grafe na n vozliščih in nato med njimi poišče zunanje ravninske ter vrne tiste z ustreznim krepkim lihim kromatičnim številom. Vendar pa je za večje število vozlišč tak postopek preveč časovno zahteven, zato smo napisali še funkcijo `generate_outerplanar.ipynb`, v kateri zunanje ravninske grafe generiramo tako, da generiramo in združujemo cikle, ter znotraj ciklov dodajamo povezave, ki se ne sekajo.

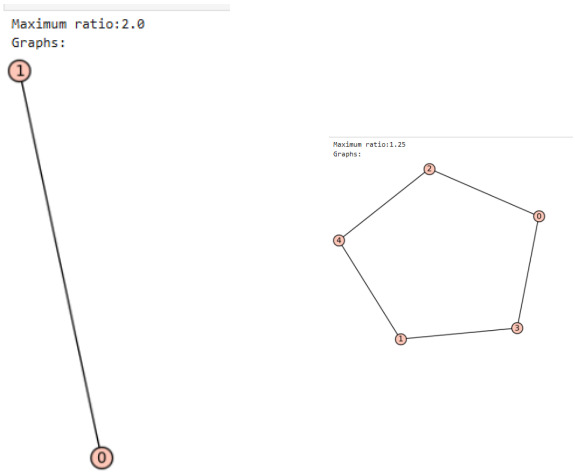
Funkcijo `outerplanar.ipynb` smo pognali na grafih do 9 vozlišč, tam nismo našli nobenega zunanje ravninskega grafa s krepkim lihim kromatičnim številom nad 7. Funkcijo `generate_outerplanar.ipynb` pa smo pognali 5000-krat in pri tem tudi nismo bili uspešni pri iskanju grafa z dovolj visokim krepkim lihim kromatičnim številom. To sicer ni nič presenetljivega, saj je tudi v podani literaturi predlagano, da bi bila 7 lahko možna zgornja meja za krepko liho kromatično število zunanje ravninskih grafov in smo torej v bistvu iskali bolj izjemne primere, ki bi se najbrž (če sploh so) pojavili pri veliko večjih in bolj kompleksnih grafih, kot smo jih bili mi sposobni preveriti.

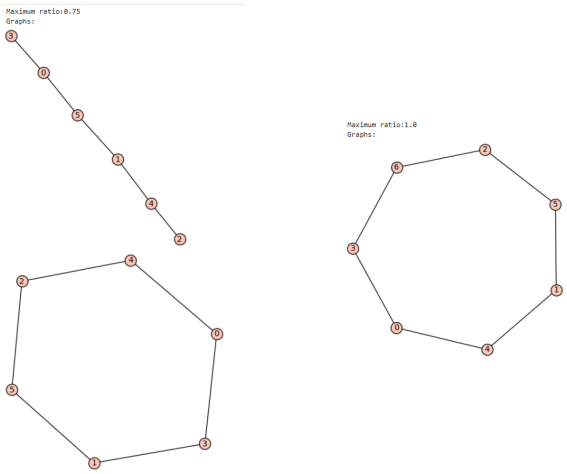
3.4 Točka 4

V tej točki smo poskušali poiskati grafe, za katere je vrednost $\chi_{so}(G)/\Delta^2(G)$ čim večja (pri čemer je $\Delta(G)$ maksimalna stopnja grafa).

Tudi tukaj smo najprej napisali funkcijo `ratio.ipynb`, ki izračuna razmerje za vse grafe na n vozliščih in vrne tistega z najboljšim. To funkcijo nam je uspelo pognati na grafih do 7 vozlišč, za več vozlišč pa smo uporabili funkcijo `ratio_random.ipynb`, ki generira k povezanih grafov z n vozlišči in vrne graf z najboljšim razmerjem izmed teh.

Najboljše razmerje, ki smo ga našli je bilo 2.0 na preprostem grafu z dvema vozliščema in eno povezavo. Drugo najboljše razmerje je bilo pri grafu C_5, to je 1.25. Za ostala števila vozlišč pa so bila najboljša najdena razmerja 1.0 (v primeru cikla, ko število vozlišč ni deljivo s 3) ali 0.75 (v primeru cikla, ko je število vozlišč deljivo s 3 ali pa poti). Kakšnih drugih grafov z višjim razmerjem na grafih nad 7 vozlišč nam pri naključnem testiranju na do 17 vozliščih pri 1000 ponovitvah ni uspelo najti.





3.5 Točka 5

Definicija 3.2. Naj bosta grafa $G = (V_G, E_G)$ in $H = (V_H, E_H)$. Množica vozlišč v vseh vrstah produktov, obravnavanih tukaj, je definirana kot:

$$V_G \times V_H := \{(g, h) \mid g \in V_G, h \in V_H\}$$

(standardni kartezični produkt ustreznih množic vozlišč).

Množice povezav so definirane na naslednji način:

(i) **Kartezični produkt**, $G\Box H$:

$$E(G\Box H) := \{(g,h)(g',h') \mid (g = g' \wedge hh' \in E_H) \vee (gg' \in E_G \wedge h = h'), g, g' \in V_G, h, h' \in V_H\}.$$

(ii) **Direktni produkt**, $G \times H$:

$$E(G \times H) := \{(g,h)(g',h') \mid gg' \in E_G \wedge hh' \in E_H, g, g' \in V_G, h, h' \in V_H\}.$$

(iii) **Krepki produkt**, $G \boxtimes H$:

$$E(G \boxtimes H) := \{(g,h)(g',h') \mid g' \in N[g] \wedge h' \in N[h], g, g' \in V_G, h, h' \in V_H, (g,h) \neq (g',h')\}.$$

Zadnja naloga našega projekta je bila potrditi veljavnost neenačbe:

$$\chi_{so}(G \circ H) \leq \chi_{so}(G)\chi_{so}(H)$$

za $\circ \in \{\times, \Box, \boxtimes\}$ (tj. za direktni, kartezični in krepki produkt grafov) in najti čim več grafov, pri katerih velja enakost.

Opomba: v navodilu za projekt je bila podana naloga pokazati obratno neenakost, torej:

$$\chi_{so}(G)\chi_{so}(H) \leq \chi_{so}(G \circ H)$$

vendar pa tako iz literature, kot z malo testiranja hitro vidimo, da ta neenakost ne drži, temveč drži obratno. Zato smo pri tej točki potrjevali prvo neenakost.

Za nižje število vozlišč smo uporabljali funkcijo `products.ipynb`, ki generira vse grafe na n in na m vozliščih ter njihove produkte ter preverja, ali drži neenakost ali pa morda celo enakost, vrne pa tiste grafe, za katere drži enakost. Funkcijo smo uporabili za $n \in \{2, 3, 4\}$ in $m \in \{2, 3, 4\}$. Dobili smo naslednje rezultate:

- Za vse testirane grafe velja relacija $\leq$.

- Za n=2, m=2: našli smo eno enakost za kartezični in eno za krepki produkt.

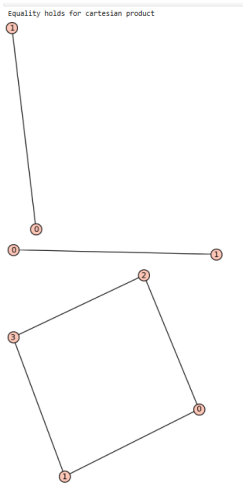


Figure 3: Kartezični produkt

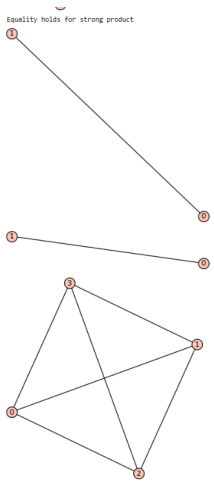


Figure 4: Krepki produkt

- Za n=3, m=2: našli smo eno enakost za kartezični in dve za krepki produkt.

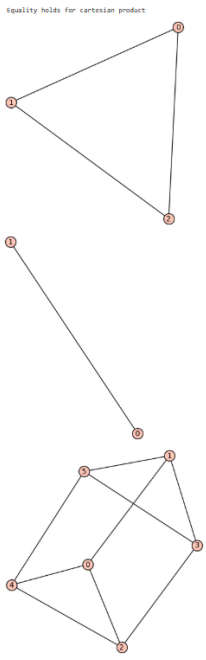


Figure 5: Kartezični produkt

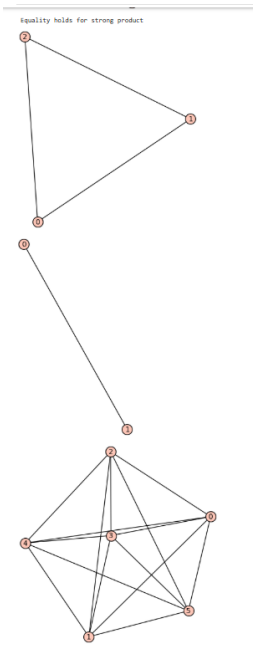


Figure 6: Krepki produkt

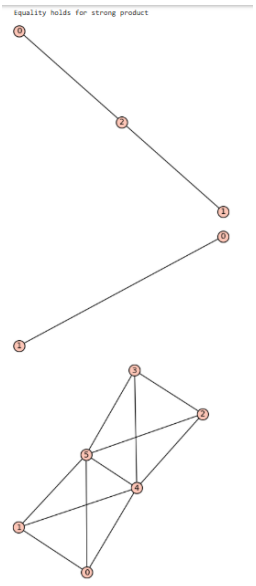


Figure 7: Krepki produkt

- Za n=3, m=3: našli smo eno enakost za direktni, eno za kartezični in tri za krepki produkt.

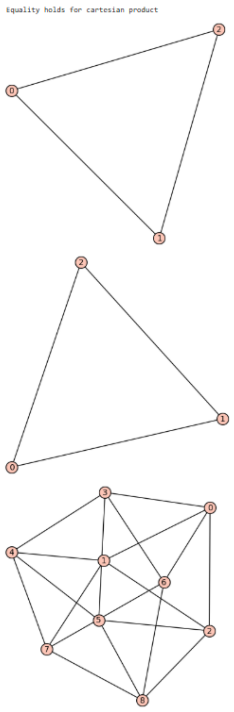


Figure 8: Kartezični produkt

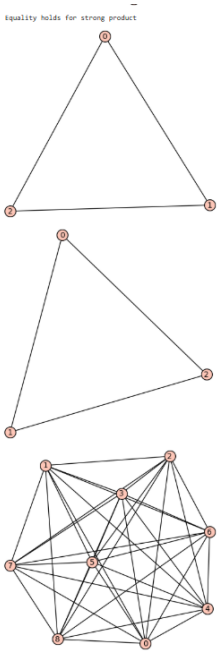


Figure 9: Krepki produkt

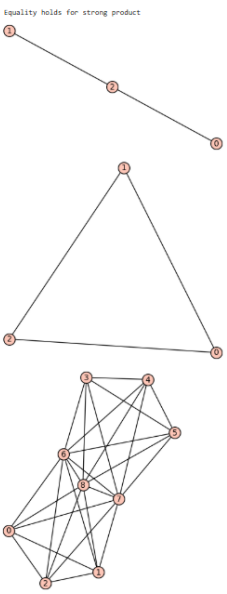


Figure 10: Krepki produkt

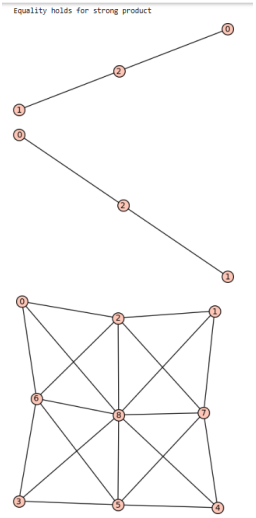


Figure 11: Krepki produkt

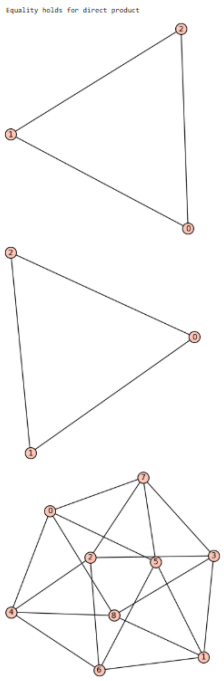


Figure 12: Direktni produkt

- Za n=4, m=2: našli smo dve enakosti za kartezični in 6 za krepki produkt.

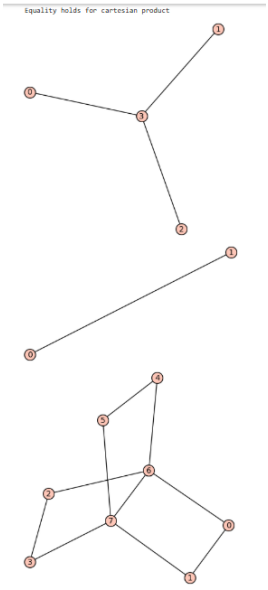


Figure 13: Kartezični produkt

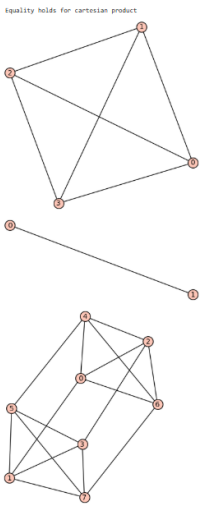


Figure 14: Kartezični produkt

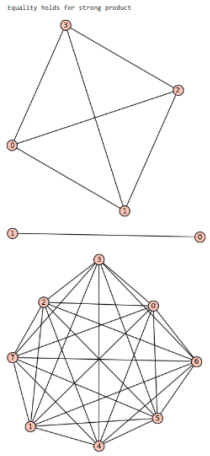


Figure 15: Krepki produkt

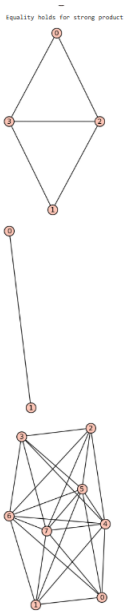


Figure 16: Krepki produkt

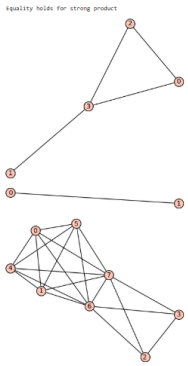


Figure 17: Krepki produkt

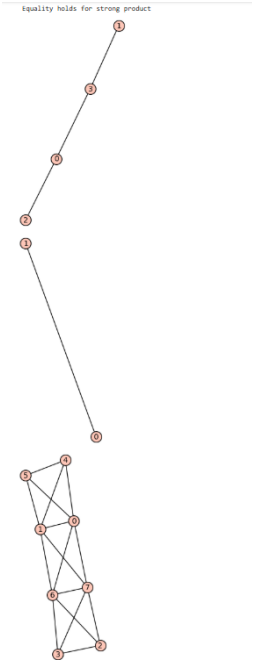


Figure 18: Krepki produkt

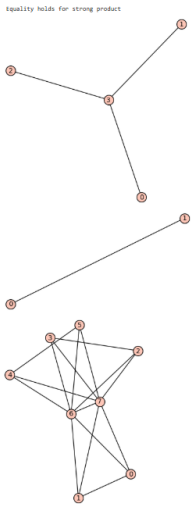


Figure 19: Krepki produkt

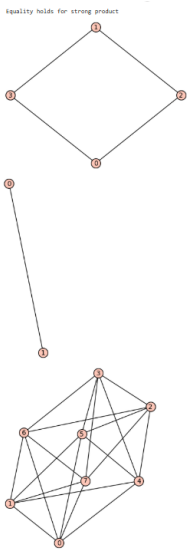


Figure 20: Krepki produkt

- Za n=4, m=3: našli smo dve enakosti za krepki produkt. (Opomba: tukaj smo kodo po 4 urah izvajanja ustavili.)

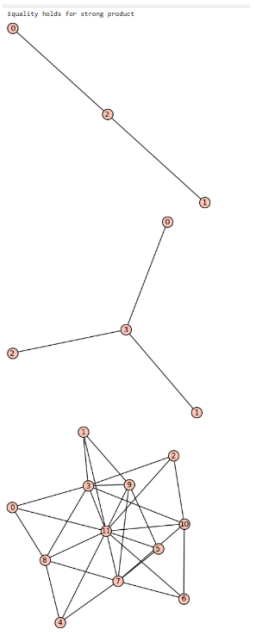


Figure 21: Krepki produkt

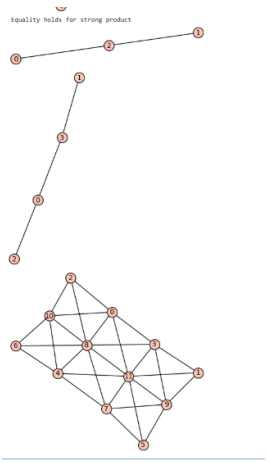


Figure 22: Krepki produkt

- Za n=4, m=4: našli smo eno enakost za kartezični in dve za krepki produkt. (Opomba: tudi tukaj smo kodo po 4 urah izvajanja ustavili.)

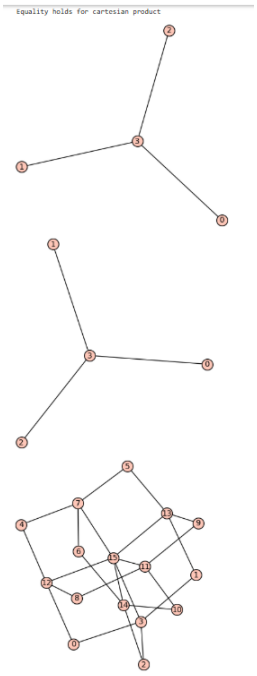


Figure 23: Kartezični produkt

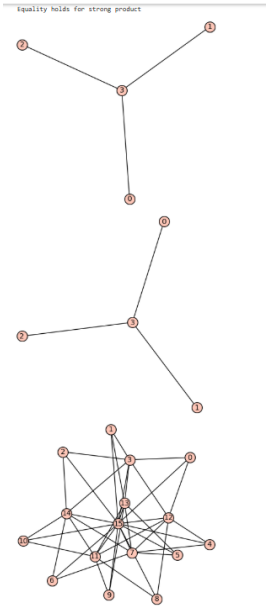


Figure 24: Krepki produkt

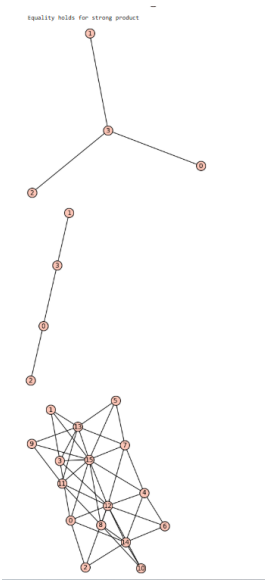


Figure 25: Krepki produkt

Prav tako nam je s tem "brute force" načinom, preden smo izvajanje kode po nekaj urah prekinili uspelo najti še par enakosti za $n \in \{5, 6, 7\}$ in m=2:

- Za n=5, m=2: ena enakost za kartezični, tri za krepki produkt.
- Za n=6, m=2: dve enakost za kartezični, 5 za krepki produkt.
- Za n=7, m=2: ena enakost za kartezični, 5 za krepki produkt

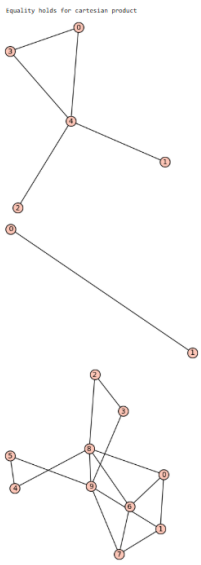


Figure 26: Kartezični produkt

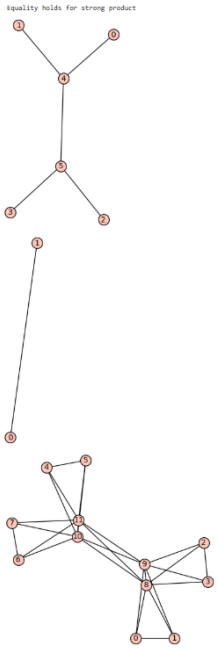


Figure 27: Krepki produkt

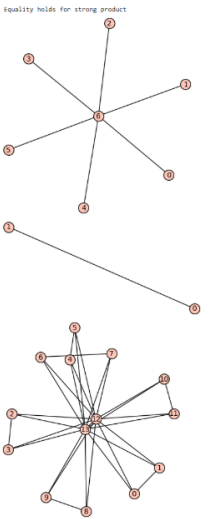


Figure 28: Krepki produkt

Za $n \in \{5, 6, 7\}$ in $m \in \{3, 4\}$ smo nekaj produktov stestirali še s funkcijo `products_random.ipynb`, vendar pa tudi tukaj z naključno generacijo nismo imeli sreče pri iskanju enakosti, kar tudi ni presenetljivo, saj je enakost prej izjema kot pravilo. Poleg tega že na majhnem številu grafov funkcija zaradi velikega števila vozlišč produktov deluje zelo počasi, zato nam za vsako kombinacijo vozlišč ni uspelo stestirati več kot 10 kombinacij grafov .

Viri

[1] Y. Caro, M. Petruševski, R. Škrekovski, and Z. Tuza, *On strong odd colorings of graphs*, arXiv preprint arXiv:2410.02336, 2024. Available at: https://arxiv.org/abs/2410.02336