Grafi z liho neodvisno množico velikosti 1

Mia Nardin, Tilen Žabkar

13. november 2025

1 Uvod

Naj bo $G = (V, E)$ graf, kjer je V množica vozlišč in E množica povezav med njimi. Liha neodvisna množica $S \subseteq V$ je posebna vrsta neodvisne množice, kar pomeni, da nobeni dve vozlišči iz S nista neposredno povezani. Poleg te lastnosti mora množica S izpolnjevati še dodatni pogoj: za vsako vozlišče v, ki ne pripada množici S (torej za vsak $v \in V \setminus S$), velja, da nima nobenega soseda v S ($N(v) \cap S = \emptyset$), ali pa ima liho število sosedov v S ($|N(v) \cap S| \equiv 1 \pmod 2$). Tukaj $N(v)$ označuje odprto množico vseh sosednjih vozlišč vozlišča v. Največja možna moč takšne množice v grafu G se imenuje liho neodvisno število grafa in jo označimo z $\alpha_{od}(G)$.
Barvanje grafa je močno liho, če velja, da se med vsemi sosednjimi vozlišči, vsakega vozlišča vsaka barva pojavi liho mnogokrat. Močno liho kromatično število $\chi_{so}(G)$ je najmanjše število barv, ki omogoča močno liho barvanje grafa G.
Povezava med $\alpha_{od}(G)$ in $\chi_{so}(G)$ je

$$\alpha_{od}(G) \cdot \chi_{so}(G) \geqslant |G|.$$

2 Opredelitev problema

V nadaljevanju obravnavamo le grafe za katere velja $\alpha_{od}(G) = 1$. Zanimale naju bodo njihove skupne lastnosti. Primeri takih grafov so:

- K_2, K_3 in na splošno vsi K_n (polni grafi),
- P_3 (pot s tremi vozlišči),
- C_4 (cikel s štirimi vozlišči), C_5 (cikel s petimi vozlišči).

Očitno je, da mora imeti vsak tak graf premer največ 2, kar pomeni, da je razdalja med katerimkoli parom vozlišč največ 2. Povezana pogoja sta še:

1. Če velja $\alpha_{od}(G) = 1$, potem velja tudi $\chi_{so}(G + K_r) = 1$.
2. Če je graf claw-free, potem velja, da je $\alpha_{od}(G) = 1$ natanko tedaj, ko ima graf premer največ 2.

3 Načrt dela

Skupno delo bo potekalo prek GitHub-a in program bo napisan v Sage-u. Najprej bo potrebno definirati funkcijo `alpha_od(G)`, ki za dan graf G vrne $\alpha_{od}(G)$. Funkcija bo implementirala celoštevilski linearni program iz navodil. Če se bo to izkazalo za prepočasno, bova raziskala druge načine, za hitrejše iskanje $\alpha_{od}(G)$.

Za generiranje grafov z največ 9 vozlišči bova uporabila vgrajeno funkcijo iz Sage-a `graphs(n)`. Za grafe z več kot 10 vozlišči bova uporabila vgrajeno funkcijo `graphs.RandomGNP(n, p)` in poskusila izračunati $\alpha_{od}(G)$ le, če je `G.diameter()` ≤ 2. Smiselno bi bilo grafe sproti shranjevati in preverjati, ali je naključno generiran graf izomorfen že obstoječemu grafu z uporabo Sage metode `G.is_isomorphic(H)`.

Prav tako bo potrebno definirati funkcijo `claw_free(G)`, ki preveri ali je dan graf G claw-free. Ta funkcija bo prevedla problem iskanja $\alpha_{od}(G)$ za G claw-free na preverjanje, ali je premer G manjši ali enak 2.

Definirala bova tudi funkcijo `chi_so(G)`, ki vrne močno liho kromatično število $\chi_{so}(G)$ zaradi pogoja 1 in preverjanja lastnosti najdenih grafov.

Iz priloženih člankov v navodilih bova poiskala še več zadostnih ali potrebnih pogojev, da velja $\alpha_{od}(G) = 1$, in jih implementirala za boljše iskanje. V članku so nekateri taki grafi že našteti, npr. $C_3, C_4, C_5, K_p \square K_q$.

Najdene grafe G bova primerjala glede na naslednje lastnosti: premer (`G.diameter()`), polmer (`G.radius()`), število povezav (`G.size()`), število vozlišč (`G.order()`), zaporedje stopenj vozlišč (`G.degree_sequence()`), ali je graf claw-free (`claw_free(G)`), $\chi_{so}(G)$ (`chi_so(G)`), ali je graf del kakšnih družin grafov (claw-free `claw_free(G)`, regularni grafi `G.is_regular()`), in še druge lastnosti, ki jih bova odkrila sproti.

Cilj naloge je poiskati skupne lastnosti najdenih grafov poleg že danih pogojev 1, 2 in lastnosti, da mora biti premer manjši ali enak 2.