Outerplanarni grafi

Jure Kraševec, Urh Videčnik

31. december 2025

1 Uvod

Naj bo $G = (V, E)$ graf, kjer je V množica vozlišč in E množica povezav med vozlišči. Outerplanarni graf (ang. Outherplanar graph) je graf, ki ga lahko narišemo v ravnini tako, da se nobeni dve povezavi ne sekata in da vsa vozlišča ležijo na zunanji strani oziroma zunanjem licu grafa. Takšni grafi so podmnožica planarnih oziroma ravninskih grafov, ki jih lahko narišemo v ravnini brez presekajočih se povezav, vendar vozlišča niso nujno na zunanji strani. Graf je outerplanaren natanko tedaj, ko ne vsebuje podgrafa, ki je homeomorfen z K_4 ali z $K_{2,3}$.
Množica vozlišč $S \subseteq V$ grafa G je liha neodvisna množica, če za vsako vozlišče $u \in I$ velja, da je lihe stopnje ter za vsak par $u, v \in I$ velja, da med njima ne obstaja povezava v E. Poleg tega velja, da vsako vozlišče $v \in V \setminus S$ nima nobenega soseda v S: $N(v) \cap S = \emptyset$ ali pa ima liho število sosedov v S: $|N(v) \cap S| \equiv 1 \pmod 2$ je liho število. $N(v)$ označuje množico sosedov vozlišča v. Največji moči takšne množice pravimo liho neodvisno število grafa, ki ga označimo z $\alpha_{od}(G)$.

2 Opredelitev problema

V nalogi nas bo zanimalo število $\alpha_{od}(G)$. Cilj naloge, je preveriti, če za vsak outerplanaren graf G velja neenakost

$$\alpha_{od}(G) \geqslant n/7,$$

kjer je n število vozlišč grafa G. Najprej bo potrebno implementirati funkcijo, ki za dan graf preveri, če je outerplanaren. Nato bo potrebno razviti algoritem, ki za dan outerplanaren graf izračuna liho neodvisno število $\alpha_{od}(G)$ ter preveri, če velja neenakost $\alpha_{od}(G) \geqslant n/7$. V nadaljevanju bo treba implementirati funkcijo, ki bo generirala naključne outerplanarne grafe različnih velikosti in zanje preverila zgornjo neenakost. Večje outerplanarne grafe bomo generirali z združevanjem ciklov poljubne dolžine, katerim dodajamo nepresečne diagonale. Dobljene grafe skupaj zlepimo v drevesno strukturo.

3 Načrt dela

Skupno delo bo potekalo prek GitHub-a in v programskem okolju `SageMath`, ki omogoča učinkovito delo z grafi ter vsebuje številne vgrajene metode za preverjanje njihovih lastnosti. Najprej bova implementirala funkcijo `is_outerplanar(G)`, ki za dan graf G preveri, ali je outerplanaren. Pri tem bova uporabila vgrajeno metodo `G.is_outerplanar()`, po potrebi pa tudi metode `G.is_planar()` in preverjanje prepovedanih podgrafov K_4 in $K_{2,3}$ z uporabo funkcij `G.subgraph()` ter `G.is_isomorphic(H)`.

V nadaljevanju bova definirala funkcijo `alpha_od(G)`, ki za dan graf G izračuna liho neodvisno število $\alpha_{od}(G)$. Funkcija bo temeljila na pregledu vseh podmnožic množice vozlišč grafa G z uporabo razreda `Subsets(G.vertices())`. Za vsako kandidatno množico S bova preverila, ali je neodvisna z uporabo metode `G.has_edge(u,v)` ter ali izpolnjuje pogoje lihe neodvisnosti z uporabo metode `G.neighbors(v)`. Ker je takšen pristop eksponenten, bo uporaben predvsem za grafe z manjšim številom vozlišč.

Za sistematično preverjanje majhnih grafov bova uporabila vgrajeno funkcijo `graphs(n)`, ki generira vse (neizomorfne) grafe z n vozlišči. Vsak generiran graf bova najprej testirala na outerplanarnost, nato pa izračunala vrednost $\alpha_{od}(G)$ in preverila, ali velja neenakost

$$\alpha_{od}(G) \geq \frac{n}{7}.$$

Rezultate bova po potrebi shranjevala ter primerjala z uporabo metode `G.is_isomorphic(H)`, da se izogneva podvajanju izomorfnih grafov.

Ker sistematično generiranje grafov hitro postane računsko prezahtevno, bova za večje vrednosti n uporabila naključno generiranje outerplanarnih grafov. V ta namen bova implementirala funkcijo, ki najprej generira cikel s pomočjo metode `graphs.CycleGraph(n)`, nato pa mu dodaja nekrižajoče se diagonale. Pri tem bova pazila, da ohraniva outerplanarnost grafa. Več takšnih 2-povezanih outerplanarnih grafov bova nato združila v drevesno strukturo z uporabo operacij nad vozlišči in povezavami, s čimer bova dobila splošne outerplanarne grafe.

Za dodatno testiranje bova po potrebi uporabila tudi naključne grafe, generirane z metodo `graphs.RandomGNP(n,p)`, pri čemer bova obdržala le tiste grafe, ki so outerplanarni. Za vsak generiran graf bova preverila osnovne lastnosti, kot so število vozlišč (`G.order()`), število povezav (`G.size()`), zaporedje stopenj vozlišč (`G.degree_sequence()`), ter nato izračunala vrednost $\alpha_{od}(G)$.

Na koncu bova rezultate analizirala in primerjala glede na velikost grafa ter strukturo outerplanarnih grafov. Posebno pozornost bova namenila primerom, pri katerih je vrednost $\alpha_{od}(G)$ blizu spodnje meje $\frac{n}{7}$, saj bi ti lahko kazali na potencialno ekstremne ali celo protiprimerne grafe.