Ekstremizacija neregularnih grafov z modifikacijo skupne σ-nepravilnosti

Tinč Arifović, Manca Kavčič

31. januar 2025

1 Uvod

V projektni nalogi sva se ukvarjala z ekstremizacijo neregularnih grafov z modifikacijo skupne σ-nepravilnosti. Projektna naloga je bila izvedena v programu SageMath in je dostopna na GitHubu, kjer so zbrane vse datoteke, ki sva jih uporabila pri samem projektu. Cilj je ugotoviti, ali je pri vseh spodaj podanih problemih največja vrednost $\sigma_t^{f(n)}(G)$ dosežena, ko je graf neregularen.

2 Opis problema

Imamo tri probleme, za katere iščemo rešitve.

1. Naj bo $f(n) = \frac{1}{n}$. Ali je največja vrednost $\sigma_t^{f(n)}(G)$ dosežena, ko je G neregularen graf?
2. Naj bo $f(n) = c$, kjer je c realno število iz intervala $(0, 1)$. Ali je največja vrednost $\sigma_t^{f(n)}(G)$ dosežena, ko je G neregularen graf?
3. Naj bo $f(n)$ pozitivna funkcija, za katero velja $\lim_{n\to\infty} f(n) = 0$. Identificirajte drevesa, ki dosežejo največjo vrednost $\sigma_t^{f(n)}(G)$.

Popolna $\sigma-$nepravilnost je podana kot

$$\sigma_t(G) = \sum_{u,v\subseteq V(G)} (d_G(u) - d_G(v))^2,$$

kjer $d_G(z)$ označuje stopnjo vozlišča z v grafu G. Natančneje, definiramo indeks $\sigma_t^{f(n)}(G)$ kot

$$\sigma_t^{f(n)}(G) = \sum_{u,v\subseteq V(G)} |d_G(u) - d_G(v)|^{f(n)},$$

kjer je $n = |V(G)|, f(n)$ pa je funkcija definiran za $n \geq 4$. Najina naloga je odgovoriti na zgoraj podana vprašanja s pomočjo računalniškega testiranja. Za manjše grafe, do vključno 8 vozlišč, sva uporabila sistematičen pristop iskanja, za večje grafe pa stohastičen. Poleg tega sva se pri 2. problemu še posebej osredotočila na vrednosti zelo blizu 0 in 1.

3 Rezultati

3.1 Prvi problem

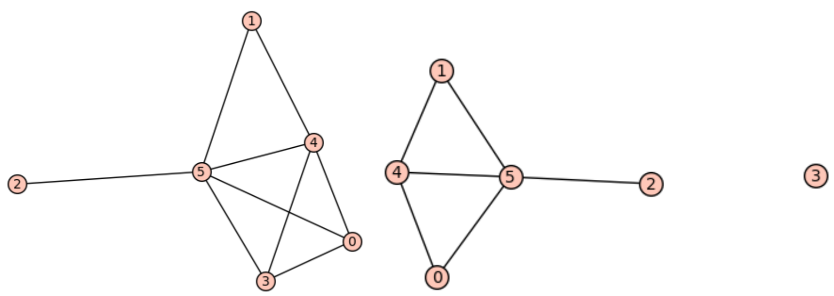
Ko je $f(n) = \frac{1}{n}$, sva pri manjših grafov povsod dobila neregularne grafe. Poleg tega se je pri vseh grafih ponavljal vzorec tega, da so grafi imeli eno vozlišče izolirano ali pa so bili grafi z le enim vozliščem stopnje 1. Prav tako se zdi, da imajo v obeh grafih z največjo vrednostjo $\sigma_t^{f(n)}(G)$ vozlišča stopnje od 0 do $n-2$, ali pa od 1 do $n-1$. Pri iskanju grafov z najvčjo $\sigma_t^{f(n)}(G)$ sva uporabila *max_total_sigma_irregularity1*, ki nama je služila kot osnova za nadgradnjo pri iskanju rešitev nadaljnjih problemov.

Algorithm 1 Iskanje grafov z največjo skupno sigma nepravilnostjo

Require: n ▷ Število vozlišč
Ensure: Grafi z največjo skupno sigma nepravilnostjo

```
grafi ← seznam vseh grafov na n vozliščih          ▷ Ustvarimo vse grafe na n vozliščih
sigme ← prazen seznam                 ▷ Seznam za vrednosti sigma nepravilnosti
for G ∈ grafi do
    Dodaj σ_1(G) v sigme        ▷ Izračunamo sigma nepravilnost in shranimo vrednost
end for
koncni_grafi ← prazen seznam      ▷ Seznam za grafe z največjo sigma nepravilnostjo
for i ∈ indeksi največjih elementov v sigme do
    Dodaj grafi[i] v koncni_grafi   ▷ Shranjujemo grafe z največjo sigma nepravilnostjo
end for
Izpiši |V| = n                                      ▷ Izpis števila vozlišč
for graf ∈ koncni_grafi do
    Prikaži graf                              ▷ Prikažemo ustrezne grafe
end for
    return None
```

Spodaj je prikazan primer grafov s 6 vozlišči. Levi ima stopnje vozlišč $(1, 2, 3, 3, 4, 5)$, desni pa $(0, 1, 2, 2, 3, 4)$.



Slika 1: *Grafa s 6 vozlišči*

Za večje grafe sva s pomočjo funkcije *regularni_grafi* generirala regularne grafe in s pomočjo funkcije *nakljucni_grafi* generirala naključne grafe.

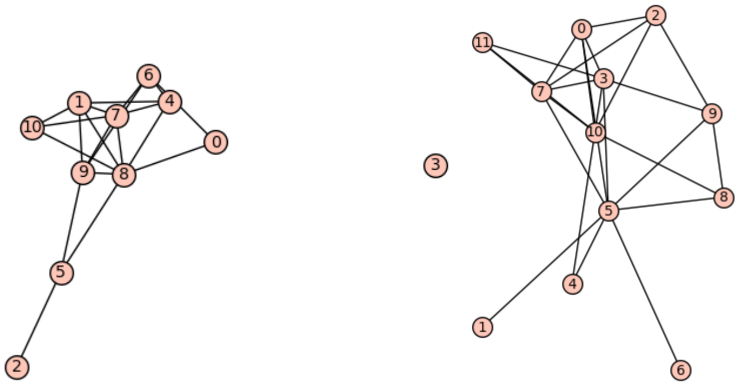
Algorithm 2 Generiranje regularnih grafov

Require: n ▷ Število vozlišč
Ensure: Seznam vseh regularnih grafov na n vozliščih
$regularni\_grafi \leftarrow$ prazen seznam ▷ Inicializiramo seznam za shranjevanje regularnih grafov
for $i \leftarrow 2$ to $n-1$ **do**
  if $n \cdot i$ ni sodo število **then** ▷ Za obstoj i-regularnega grafa mora biti $n \cdot i$ sodo
    continue ▷ Preskočimo to vrednost i
  else
    $niz \leftarrow$ niz ukazov za nauty geng ▷ Oblikujemo niz za generiranje regularnih grafov
    for $G \in$ graphs.nauty_geng(niz) **do**
      Dodaj G v $regularni\_grafi$ ▷ Dodamo graf v seznam
    end for
  end if
end for
return $regularni\_grafi$

Algorithm 3 Generiranje naključnih grafov

Require: n, st ▷ Število vozlišč in število grafov za generiranje
Ensure: Seznam naključnih grafov
$seznam\_grafov \leftarrow$ prazen seznam ▷ Inicializiramo seznam za shranjevanje grafov
for $i \leftarrow 1$ to st **do**
  $mozne\_povezave \leftarrow$ vse možne povezave med pari vozlišč ▷ Ustvarimo seznam vseh možnih povezav
  $izbrane\_povezave \leftarrow$ naključno izbran podmnožica iz $mozne\_povezave$ ▷ Naključno izberemo povezave
  $G \leftarrow$ prazen graf na n vozliščih ▷ Ustvarimo prazen graf
  Dodaj povezave $izbrane\_povezave$ v G ▷ Dodamo izbrane povezave v graf
  Dodaj G v $seznam\_grafov$ ▷ Dodamo graf v seznam
end for
return $seznam\_grafov$

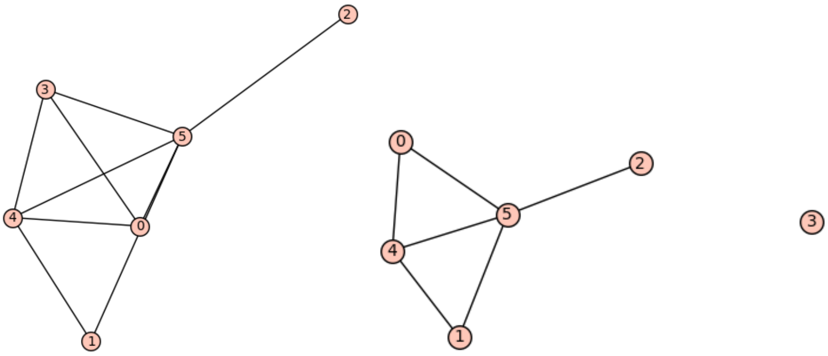
Tudi pri večjih grafih sva dobila povsod neregularne grafe kot rezultat. Zdi se, da tudi tukaj velja podobna lastnost kot pri manjših grafih. Ker pa nisva pregledala vseh grafov, ne moreva reči zagotovo, da velja ista razporeditev stopenj, kot sva jo opazila pri manjših grafih.



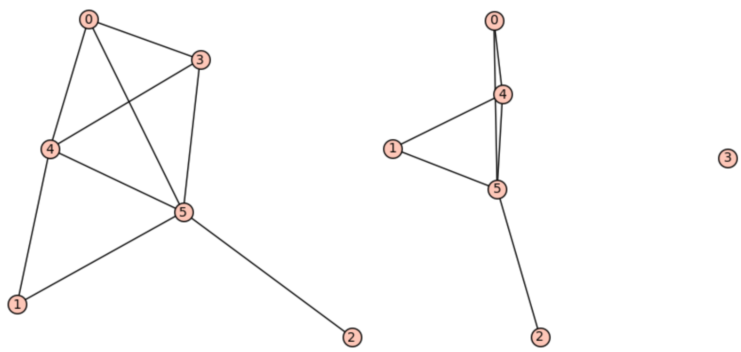
Slika 2: *Grafa z 11 in 12 vozlišči*

3.2 Drugi problem

Pri drugem problemu sva pri manjših grafih za zelo majhne c blizu 0 in blizu 1 dobivala enake grafe kot pri prvem problemu, prav tako za vse vmesne vrednosti c. Za c sva uporabila vrednosti $\{\frac{1}{1000}, \frac{1}{100}, \frac{1}{10}, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, \frac{9}{10}, \frac{99}{100}, \frac{999}{1000}\}$. Pri vseh vrednostih sva za manjše grafe dobila neregularne grafe.

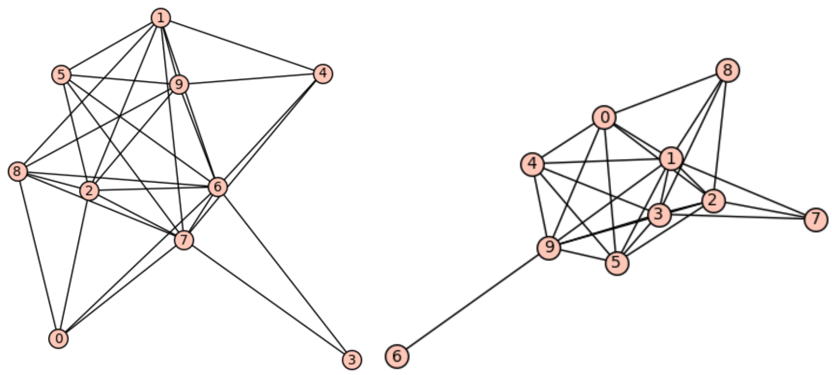


Slika 3: *Grafa s 6 vozlišči za* $c = \frac{1}{1000}$



Slika 4: *Grafa s 6 vozlišči za* $c = \frac{999}{1000}$

Pri večjih grafih sva prav tako dobila povsod neregularne grafe. Spodaj je podan primer grafa z 10 vozlišči. Ker se algoritem ni izvedel na vseh grafih, za nekatere grafe kot rezultat nisva dobila grafa, ki bi imel vozlišče stopnje 0 ali 1.



Slika 5: *Grafa z 10 vozlišči za* $c = \frac{1}{100}$ *(levo) in* $c = \frac{99}{100}$ *(desno)*

3.3 Tretji problem

Za ta problem sva se odločila uporabiti 9 funkcij, za katere velja

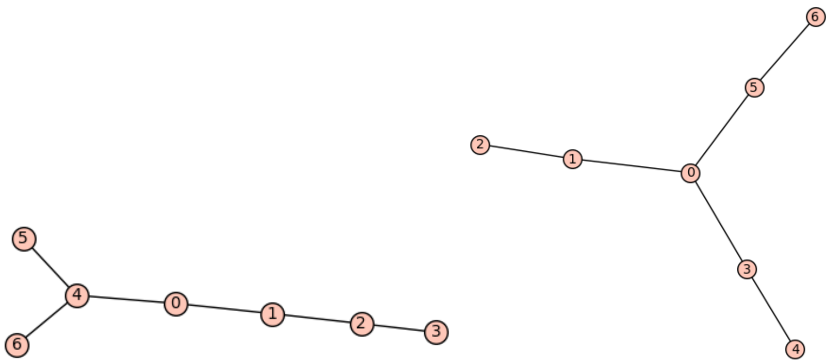
$$\lim_{n\to\infty} f(n) = 0.$$

Funkcije so razvrščene glede na hitrost njihovega približevanja vrednosti 0, od najhitreje padajočih do najpočasneje padajočih:

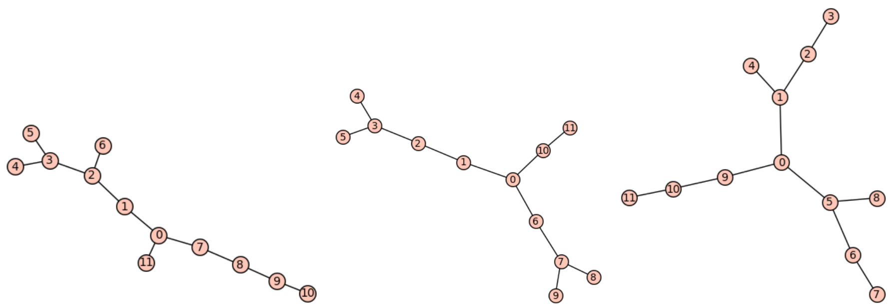
1. e^{-x^2}
2. $\frac{e^{-x}}{\sqrt{x}}$
3. e^{-x}
4. $\frac{1}{x^2}$
5. $e^{-\sqrt{x}}$
6. $\frac{|\cos(x)|}{x}$
7. $\frac{\sin^2(x)}{x}$
8. $\frac{1}{\ln^2(x)}$
9. $\frac{1}{\sqrt{x}}$

Pri tem delu sva nalogo razdelila na dva dela. Najprej sva si ogledala grafe z največjo vrednostjo $\sigma_t^{f(n)}(G)$ in ugotovila, da je bila povsod dosežena pri neregularnih grafih tako pri malih kot pri velikih grafih. Pri malih grafih sva ponekod

dobila enake grafe kot pri prvem problemu, kar ni ravno presenetljivo, glede na to, da je tudi tam $\lim_{n\to\infty} \frac{1}{n} = 0$. Prav tako sva za velike grafe dobila grafe s podobnimi lastnostmi kot pri prejšnjih nalogah.
Nato pa sva definirala funkcijo *drevesa* in s pomočjo *TreeIterator* generirala drevesa, na katerih sva potem uporabila funkcijo *max_total_sigma_irregularity3_drevesa*. Vsa dobljena drevesa so v obliki daljših poti (verige), od katerih se odcepi vsaj eno vozlišče (razen pri grafih s 4 vozlišči, kjer v določenih primerih največjo vrednost doseže pot dolžine 4). S povečevanjem števila vozlišč se poveča število vozlišč, odcepljenih od najdaljše poti.



Slika 6: *Drevesi s 7 vozlišči in* $f(n) = e^{-\sqrt{n}}$



Slika 7: *Drevesa z 12 vozlišči in* $f(n) = e^{-n^2}$. Pri grafu z 12 vozlišči, kot je zgoraj prikazano, se odcepijo 3 vozlišča, če se vsako odcepi posamično ali 4, če se iz enega vozlišča odcepi skupek dveh vozlišč ali 5, če se iz enega vozlišča odcepi skupek treh vozlišč.

4 Zaključek

Kot komentar bi dodala, da se je čas izvajanja programa povečeval s povečevanjem števila vozlišč, saj se z večjim številom vozlišč poveča število različnih grafov, obenem pa je za vsak graf pri izračunu $\sigma_t^{f(n)}(G)$ potrebnih več računskih operacij. Pri drugem problemu je bilo za c zelo blizu 0 in 1 potrebnega več časa za izračun vrednosti kot pri ostalih vrednostih c.