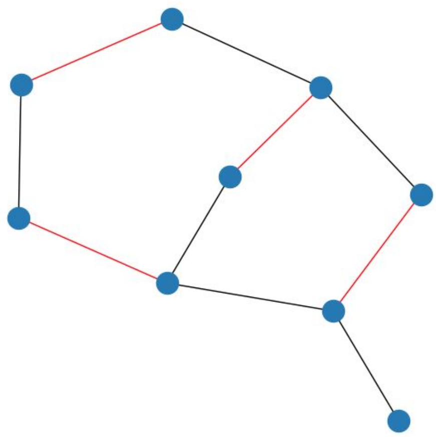
Rešavanje problema maksimalnog uparivanja grafa

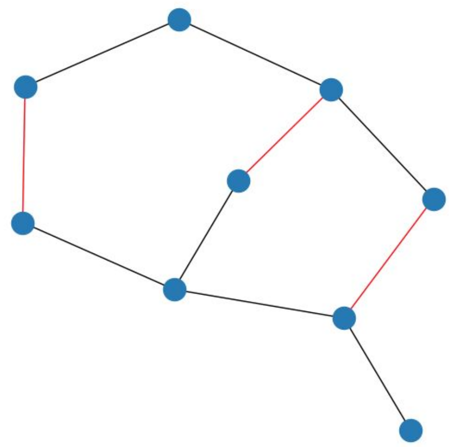
Nikola Vuković, Matija Milićević
Matematički fakultet, Univerzitet u Beogradu
Jun 2019.

1 - Uvod

Uparivanje grafa G=(V, E) je podskup ivica E' takav da nijedan par ivica ne deli čvor. Kardinalnost uparivanja označava broj ivica koje uparivanje sadrži i obeležava se sa |**E'**|.
Maksimalno uparivanje je uparivanje koje se ne može proširiti dodavanjem nove grane (slika 1). Dodavanje nove grane narušava osobinu uparivanja da ivice iz E' ne dele čvorove.
Minimum maksimalno uparivanje je maksimalno uparivanje sa najmanjim mogućim brojem grana, uparivanje za koje važi da je |E'| minimalno (slika 2).



Slika 1: Maksimalno uparivanje



Slika 2: Minimum maksimalno uparivanje

Problem maksimalnog uparivanja podrazumeva traženje maksimalnog uparivanja sa najmanjom mogućom kardinalnošću, odnosno minimum maksimalno uparivanje.

Problem pripada NP klasi [3], konkretnije klasi problema aproksimativnih u konstantnom faktoru. Moguće je pronaći rešenje problema u faktoru od 2 [1] - bilo koje maksimalno uparivanje neće biti više od 2 puta veće od minimalnog. Ako je M maksimum uparivanje grafa G, |M| = α, onda za svaku ivicu u M bar jedan čvor ivice mora biti u proizvoljnom maksimalnom uparivanju E'. Ukoliko to ne važi mogli bismo da dodamo ivicu u uparivanje E', što bi značilo da ono nije maksimalno. Dakle, broj čvorova u maksimalnom uparivanju mora biti bar α. Pošto ivica u uparivanju može najviše da sadrži dva čvora, sledi da broj ivica u proizvoljnom maksimalnom uparivanju grafa G mora biti bar $\alpha/2$. Pošto je broj ivica u maksimum uparivanju α, a svako uparivanje mora imati bar $\alpha/2$ ivica, možemo da zaključimo da je broj ivica u minimum maksimalnom uparivanju između $\alpha/2$ i α. Da bismo našli rešenje u faktoru od 2, samo treba da nađemo bilo koje maksimalno uparivanje. To možemo postići dodavanjem nesusednih ivica u skup E', sve dok nam ne preostanu samo ivice koje dele čvorove sa ivicama u skupu [2].

Dominantni skup ivica grafa G=(V, E) je podskup ivica E' takav da je svaka ivica koja nije u E' incidentna sa bar jednom ivicom iz E'.

Problem se može poistovetiti sa problemom minimalnog dominantnog skupa ivica grafa jer je maksimalno pokrivanje kardinalnosti k takođe dominantni skup ivica kardinalnosti k. Obrnutno, sa minimalnim dominantnim skupom ivica kardinalnosti k možemo u polinomijalnom vremenu da konstruišemo maksimalno uparivanje kardinalnosti k [3].

2 - Predložena rešenja

Testirani su različiti pristupi u rešavanju problema maksimalnog uparivanja. Svi pristupi su implementirani u programskom jeziku Python 3 uz biblioteke *NetworkX* za kreiranje i apstrahovanu internu reprezentaciju grafova, i *Matplotlib* za prikaz samih grafova, kao i grafikona koji upoređuju pristupe.

2.1 - Naivni pristup - Gruba Sila

Kako bismo ispitali sva moguća rešenja moramo da ispitamo svaku moguću permutaciju grana u grafa. Jasno je da je za graf G=(V, E) složenost O(|E|!), te gruba sila nije primenljiva u realnim okolnostima. Ipak, ovo rešenje zagarantovano daje optimalan rezultat pa se može koristiti kao reper (za primere manjeg reda veličine).
Sledeća tabela ilustruje neprimenljivost algoritma za veće grafove.

Broj grana	**Prosečno vreme pretrage**
9	Manje od sekunde
10	8 sekundi
11	90 sekundi
12	18 minuta
13	Oko 4 sata

2.2 - Heuristički pristup

Kada ubacimo granu u uparivanje automatski isključujemo iz uparivanja sve njoj incidentne grane. Ovaj pristup je vođen intuicijom da bi trebalo da prvo biramo grane koje će automatski isključiti što više grana i time imati što veći efekat. Odatle heuristika da su grane sa većim stepenom bolje. Tehnika predstavlja pohlepan algoritam što se u praksi pokazuje kao primenljivo rešenje. Algoritam zahteva sortiranje ivica po stepenu, pa ima složenost O(|E| log|E|).

Pri testiranju pristupa primećena je sledeća pojava: čvorovi mogu da imaju isti stepen (i budu jednako "kvalitetni" po heuristici) a da permutacije u kojima su izabrani daju različita rešenja. Iz ovog razloga i u ovom pristupu postoji ordeđen element slučajnosti. Zbog ove pojave implementirane su dve funkcije u okviru ovog pristupa: jedna koja vraća po heuristici najidealnije rešenje, i druga koja omogućava ispitivanje određenog broja rešenja sortiranih po heuristici. Pojava nije česta pa obe funkcije obično daju isti rezultat.

2.3 - Metaheuristički pristup - Simulirano kaljenje

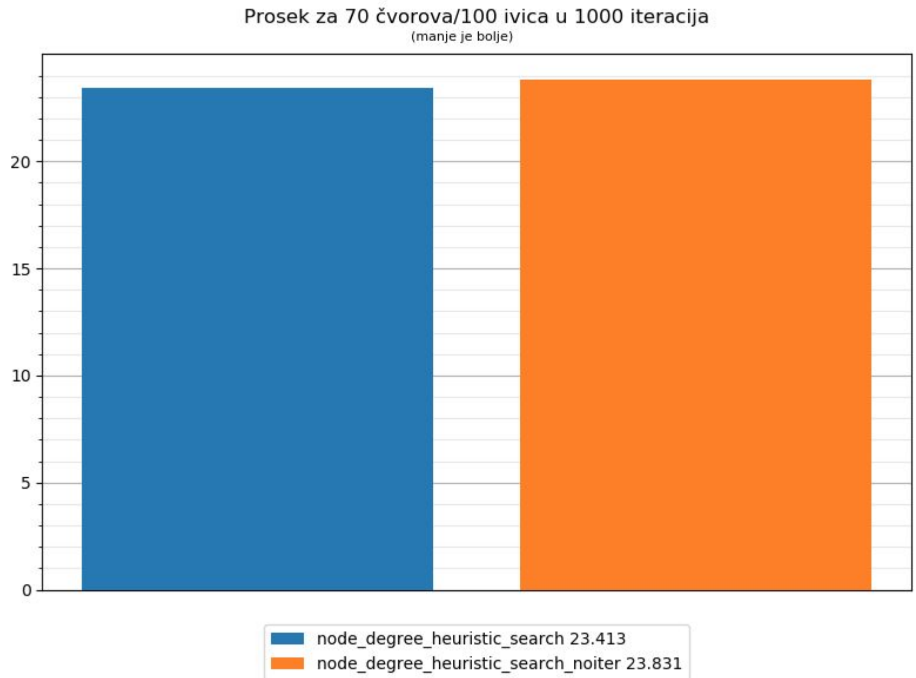
Pristup iterativne optimizacije rešenja kroz simulirano kaljenje deluje primenljivo pri rešavanju ovakvog problema. Ipak, bolji rezultati su primećeni kod kaljenja koje brže konvergira, što može da znači da bi jednostavnija pohlepna pretraga mogla da funkcioniše još bolje.

Pored standardnog pristupa simuliranim kaljenjem, implementirana je i hibridna pretraga sa heurističkim pristupom. Konkretno, vrednost heuristike (prosečan stepen grane u rešenju) korišćen je za ocenjivanje kvaliteta rešenja.

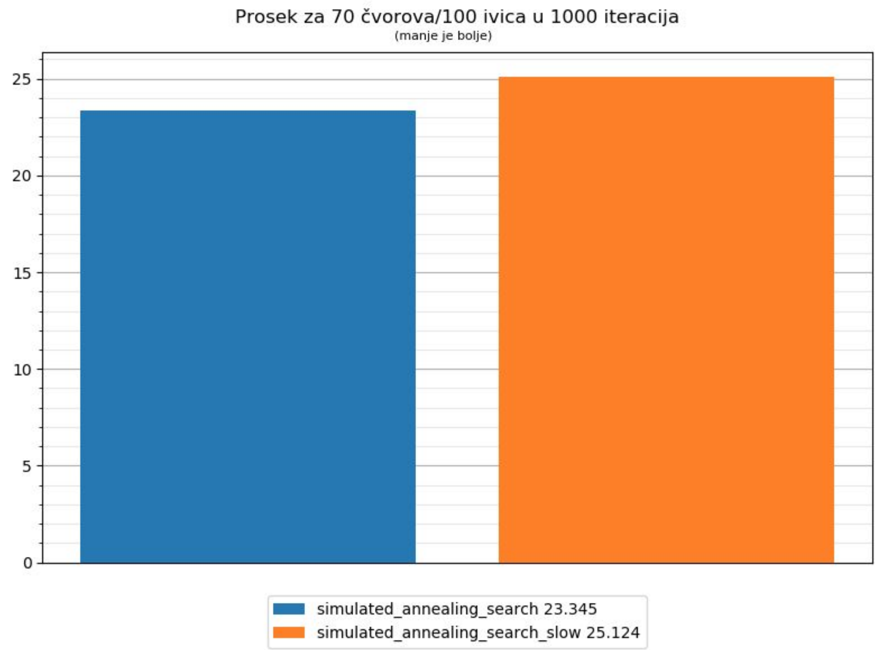
3 - Testiranje rešenja

Svi testovi su izvršeni na računaru sa Intel Core i5 procesorom na 2.7 GHz. Radi jasnijeg poređenja ni u jednom pristupu nije primenjena paralelizacija.

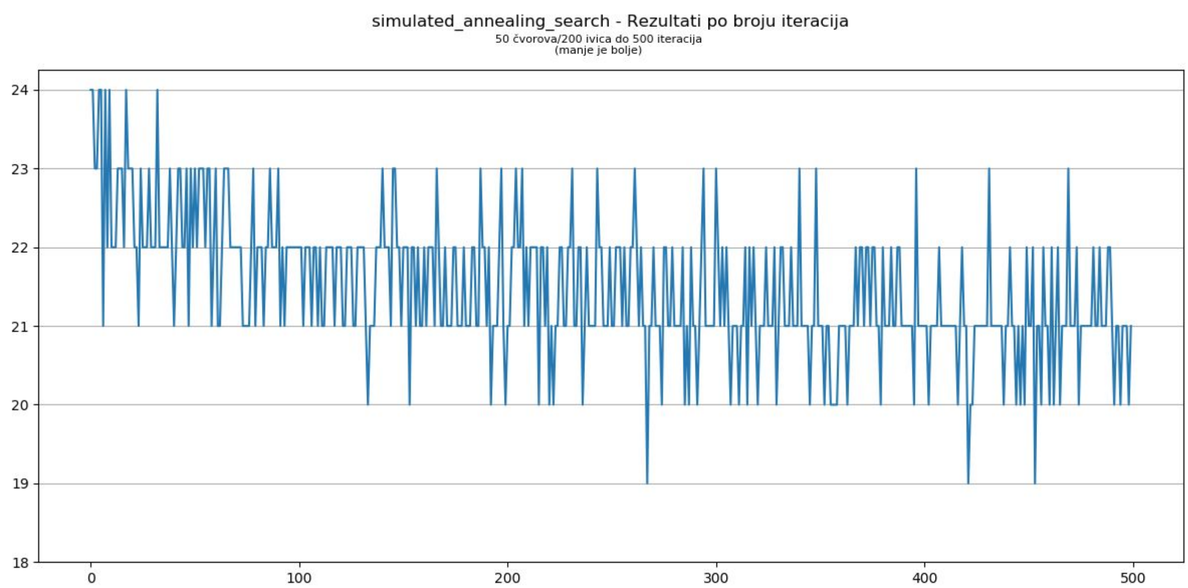
3.1 - Analiza pojedinačnih rešenja



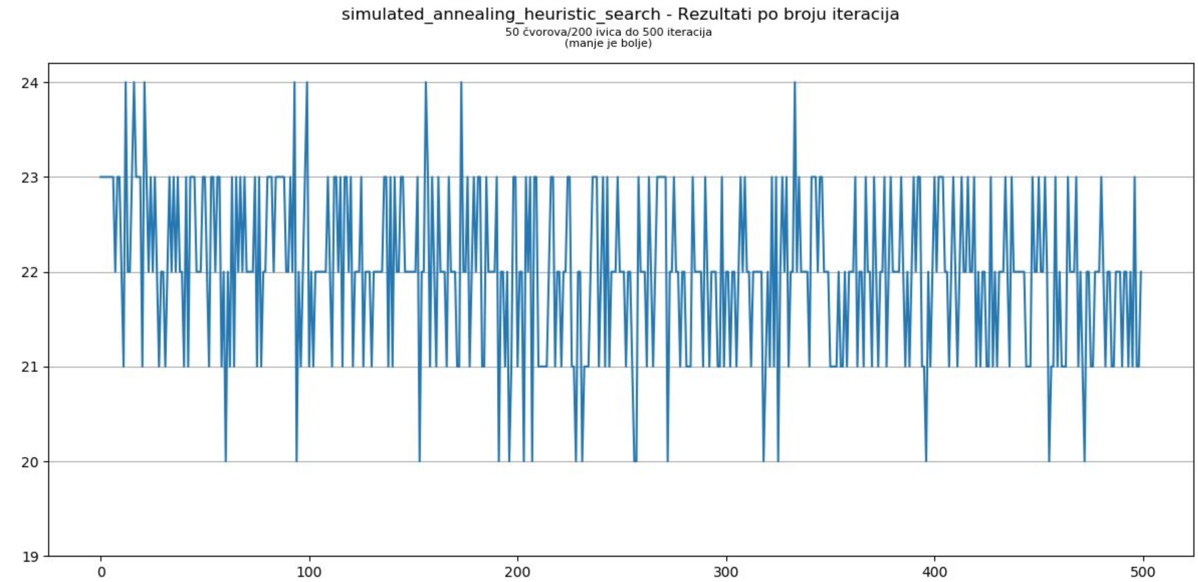
Dve varijacije heurističke pretrage: Vidimo da verzija koja prolazi kroz određen broj rešenja (ovde 100) daje malo bolje prosečne rezultate nego ispitivanje isključivo prvog rešenja (rešenja imaju prosečnu kardinalnost od 23.413 naspram 23.831). U gornjem testu prvi pristup je bio bolji u 421 slučaju, dok su u 579 slučajeva obe funkcije dale iste rezultate.



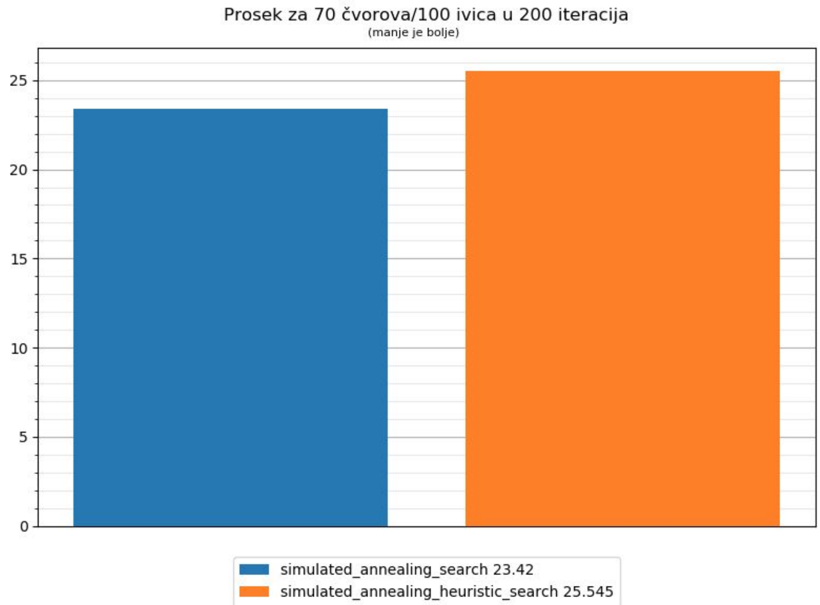
Različite brzine konvergiranja ka lokalnom minimumu u simuliranom kaljenju: Kroz testiranje zaključeno je da se simulirano kaljenje ponaša bolje kada konvergira brže ka rešenju (često je lokalni minimum i globalni). Prikazane su varijante gde su verovatnoće prihvatanja lošijeg rešenja $1/n$ i $1/\sqrt{n}$ gde je n broj trenutne iteracije. Prvi pristup je bio bolji u 787 slučajeva, drugi u svega 71, dok su u 142 slučaja rezultati bili isti.



Poboljšanje simuliranog kaljenja sa brojem iteracija: Vidimo da sa većim brojem iteracija simulirano kaljenje zaista daje bolje rezultate. Ipak, moguće je videti da algoritam povremeno “nazaduje” sa kvalitetom rešenja.

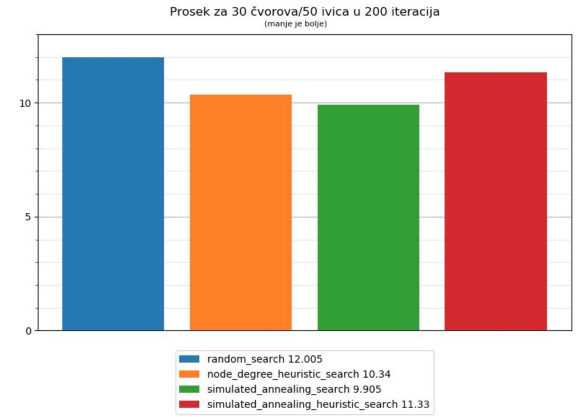


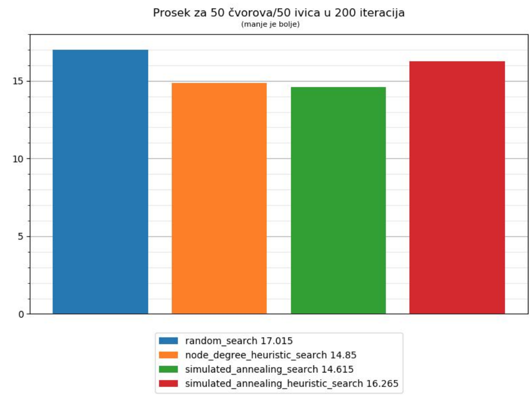
Poboljšanje simuliranog kaljenja sa brojem iteracija (heuristički pristup): Ovaj pristup mnogo češće "nazaduje" sa kvalitetom rešenja jer se ispostavlja da striktno gledanje prosečnog stepena grane u rešenju nije dovoljan indikator kvaliteta rešenja.

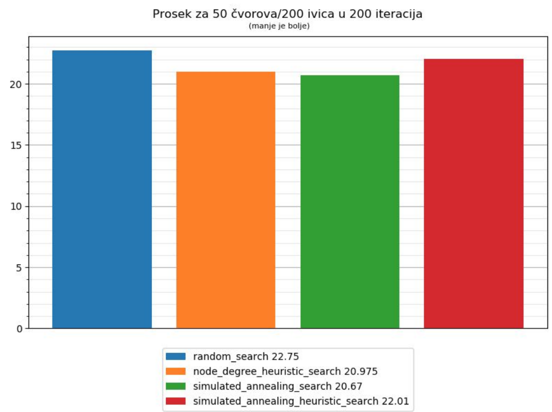


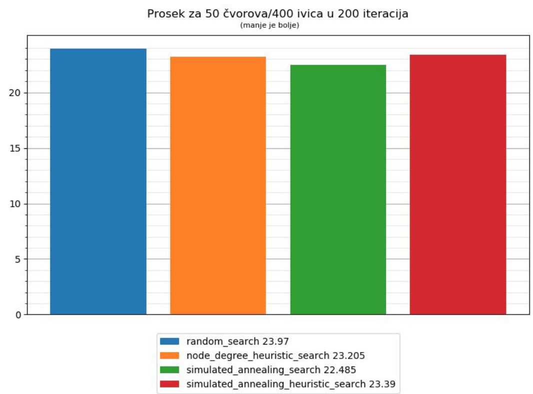
Heuristički pristup simuliranog kaljenja se ponaša dosta gore od običnog: od 200 slučajeva lošiji je u 160, jednak u 27, a bolji u samo 13.

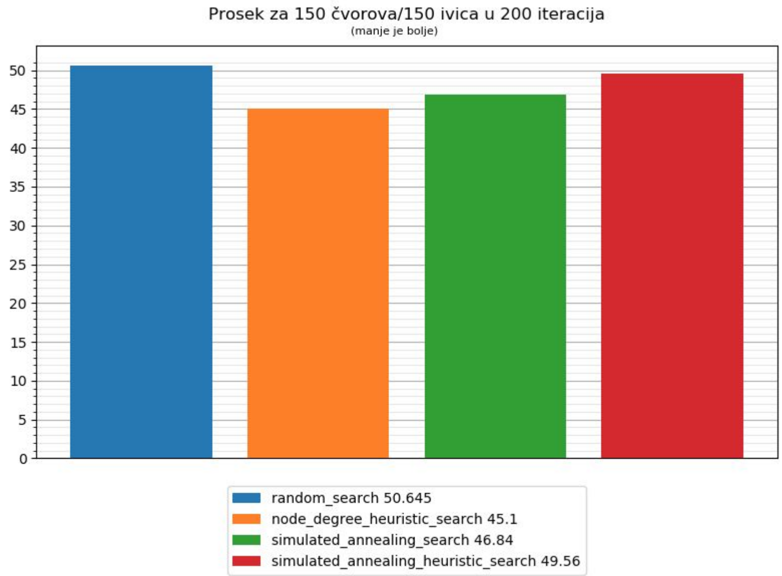
3.2 - Poređenje rešenja

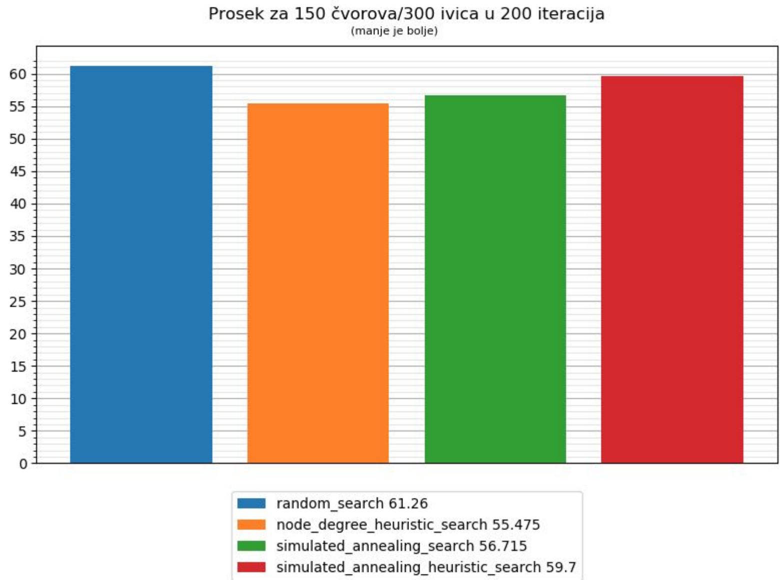


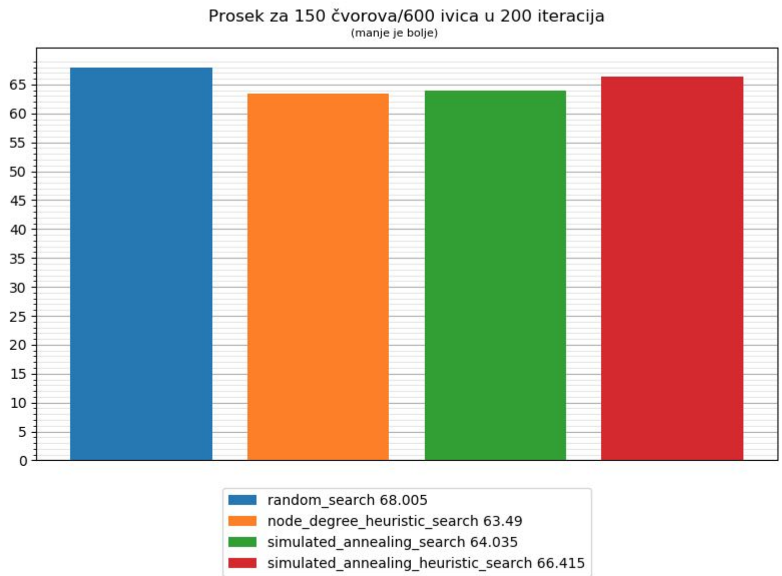


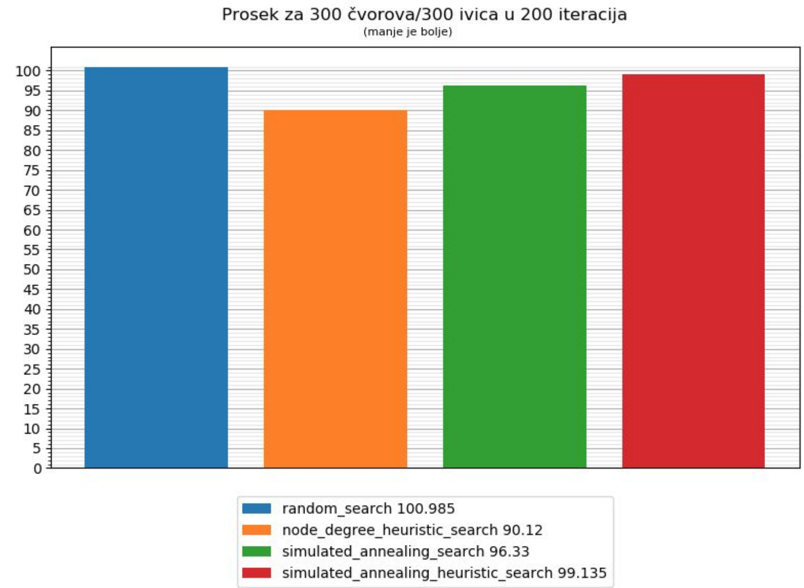


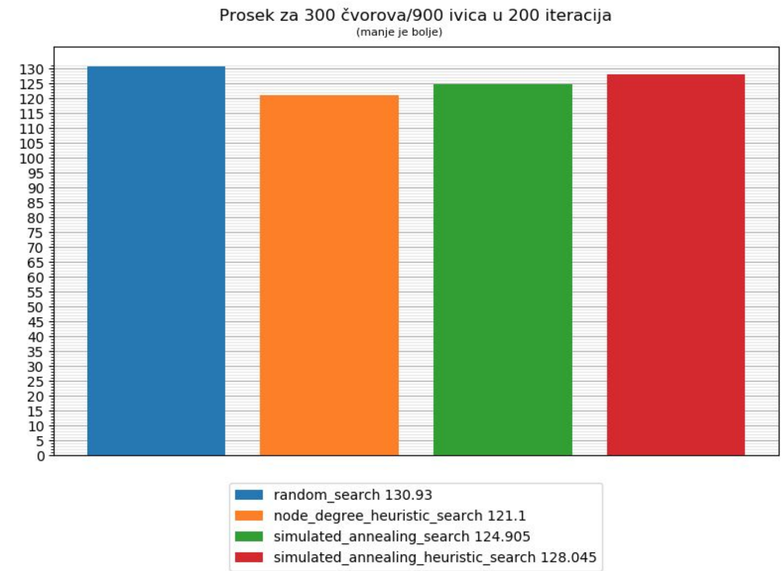


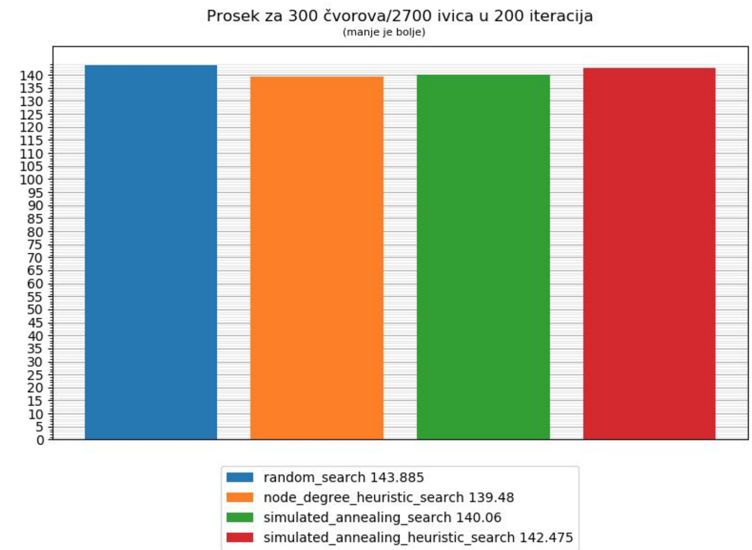

Prosek za 150 čvorova/150 ivica u 200 iteracija
(manje je bolje)
0
5
10
15
20
25
30
35
40
45
50
random_search 50.645
node_degree_heuristic_search 45.1
simulated_annealing_search 46.84
simulated_annealing_heuristic_search 49.56


Prosek za 150 čvorova/300 ivica u 200 iteracija
(manje je bolje)
0
5
10
15
20
25
30
35
40
45
50
55
60
random_search 61.26
node_degree_heuristic_search 55.475
simulated_annealing_search 56.715
simulated_annealing_heuristic_search 59.7


Prosek za 150 čvorova/600 ivica u 200 iteracija
(manje je bolje)
0
5
10
15
20
25
30
35
40
45
50
55
60
65
random_search 68.005
node_degree_heuristic_search 63.49
simulated_annealing_search 64.035
simulated_annealing_heuristic_search 66.415


Prosek za 300 čvorova/300 ivica u 200 iteracija
(manje je bolje)
random_search 100.985
node_degree_heuristic_search 90.12
simulated_annealing_search 96.33
simulated_annealing_heuristic_search 99.135


Prosek za 300 čvorova/900 ivica u 200 iteracija
(manje je bolje)
random_search 130.93
node_degree_heuristic_search 121.1
simulated_annealing_search 124.905
simulated_annealing_heuristic_search 128.045


Prosek za 300 čvorova/2700 ivica u 200 iteracija
(manje je bolje)
random_search 143.885
node_degree_heuristic_search 139.48
simulated_annealing_search 140.06
simulated_annealing_heuristic_search 142.475

Primećujemo da algoritmi daju dosta slično rešenje u proseku. Čak ni razlika između najboljeg rešenja i nasumično izabranog često nije velika.
Heuristička pretraga i standardno simulirano kaljenje daju najbolje rezultate. Heuristička pretraga daje bolje rezultate na ređim grafovima (gde su razlike između stepena grana veće). Na gustijim grafovima preporučujemo pristup kaljenjem zbog manje složenosti a uporedivih rezultata.

Zaključak

Problem nije moguće rešiti na optimalan način. Ipak, često i nasumično odabrano rešenje daje "dovoljno dobro rešenje". Ovde izuzimamo manje grafove gde je razlika od jedne grane u kardinalnosti čini veliku relativnu razliku. Svi implementirani pristupi daju marginalno bolje rešenje, pa je pitanje koliko je bitno optimalno rešenje u kontekstu primene i da li opravdava dužu optimizaciju zarad relativno malog rezultata.

Na manjim grafovima pristup kaljenjem deluje kao bolji pristup, dok se za veće grafove (preko 50 čvorova), kao i ređe grafove, preporučuje heuristički pristup ukoliko složenost sortiranja liste grana nije problematična

Potencijalno poboljšanje bi moglo da bude otkriveno u korišćenju genetskih algoritama. Blossom algoritam takođe ostaje neistražen, pa se ostavlja kao sugestija čitaocu.

Literatura

1. *A compendium of NP optimization problems, Minimum maximal matching* https://www.nada.kth.se/~viggo/wwwcompendium/node21.html
2. Arobinda Gupta, *Advanced Graph Theory (course materials)*, 2012. http://cse.iitkgp.ac.in/~agupta/graph/Sol-T5.pdf
3. Yannakakis, Gavril, *Edge dominating sets in graphs*, 1980. http://cgi.di.uoa.gr/~vassilis/co/dominating-sets.pdf