Matematički fakultet

Matematičko programiranje i optimizacija

Seminarski rad

# Problem optimalnog planiranja bežičnih meš mreža

*Student:*
Lazar Mrkela
1062/2014

*Nastavnik:*
dr Zorica Stanimirović

7. juli 2016

**Sažetak**

Bežične meš mreže predstavljaju novu tehnologiju bežičnih mreža. Jedna od osnovnih karakteristika ovakvih mreža jeste niska cena instalacije i održavanja. Ovaj rad se bavi planiranjem bežičnih meš mreža s ciljem minimizacije troškova instalacije mreže. Problem je rešavan pomoću dve metaheurističke metode, genetskim algoritmom i metodom promenljivih okolina. Pored toga, urađena je i hibridizacija ove dve metode. Implementirane metode su testirane na generisanim instancama, a rezultati su upoređeni sa rezultatima egzaktnog rešavača.

*Ključne reči*: Bežične meš mreže, optimizacija, genetski algoritam, metoda promenljivih okolina.

# I. Uvod

Bežična meš mreža (BMM) (engl. *wireless mesh network*) se izgrađuje od tri vrste uređaja: ruteri, prolazi (engl. *gateway*) i klijenti. Pored toga što klijentima pružaju pristup mreži, ruteri i prolazi se mogu međusobno povezivati. Za razliku od rutera, prolazi imaju imaju sposobnost žičanog povezivanja. Oni imaju ulogu mrežnih prolaza ka žičanoj osnovi mreže (engl. *wired backbone*). Zbog navedenih dodatnih sposobnosti, prolazi imaju veću cenu instalacije. Prolazi se još nazivaju i pristupnim tačkama (engl. *access point*).

Neka su date određene lokacije na kojima se mogu instalirati ruteri i prolazi, kao i lokacije klijenata. Problem planiranja BMM predstavlja izbor nekog podskupa datih lokacija i izbor vrste uređaja koji će biti instaliran na određenoj lokaciji. Pri tome želimo da ukupni instalacioni troškovi budu što manji. Dodatno moramo uzeti u obzir pokrivenost svih klijenata i rutiranje saobraćaja kroz mrežu [1].

## I. Matematička formulacija problema

Na osnovu promenljivih i parametara definisanih u tabeli 1, planiranje BMM se formuliše kao problem linearnog programiranja:

$$\min \quad \sum_{j\in S}(c_j z_j + p_j w_{jN}) \tag{1}$$

$$\text{s.t.} \quad \sum_{j\in S} x_{ij} = 1 \quad \forall i \in I \tag{2}$$

$$x_{ij} \le z_j a_{ij} \quad \forall i \in I \ \ \forall j \in S \tag{3}$$

$$\sum_{i\in I} d_i x_{ij} + \sum_{l\in S}(f_{lj} - f_{jl}) - f_{jN} = 0 \ \ \forall j \in S \tag{4}$$

$$f_{lj} + f_{jl} \le u_{jl} y_{jl} \quad \forall j,l \in S \tag{5}$$

$$\sum_{i\in I} d_i x_{ij} \le v_j \quad \forall j \in S \tag{6}$$

$$f_{jN} \le u_{jN} w_{jN} \quad \forall j \in S \tag{7}$$

$$y_{jl} \le z_j, \ \ y_{jl} \le z_l \quad \forall j,l \in S \tag{8}$$

$$y_{jl} \le b_{jl} \quad \forall j,l \in S \tag{9}$$

$$z_{j_\ell^{(i)}} + \sum_{h=\ell+1}^{L_i} x_{ij_h^{(i)}} \le 1 \ \ \forall \ell = 1...L_i - 1 \ \ \forall i \in I \tag{10}$$

$$x_{ij}, z_j, y_{jl}, w_{jN} \in \{0,1\} \quad \forall i \in I \ \ \forall j,l \in S \tag{11}$$

**Tabela 1:** *Parametri i promenljive koji se koriste pri matematičkoj formulaciji problema. N označava osnovu mreže.*

| | |
|---|---|
| S | skup kandidatskih lokacija za instalaciju uređaja |
| I | skup klijenata |
| $S_i$ | uređen (nerastuće po jačini signala) podskup lokacija koje mogu da pokriju klijenta $i$ |
| $L_i$ | broj elemenata skupa $S_i$ |
| $c_j$ | cena instalacije rutera na lokaciji $j$ |
| $p_j$ | dodatna cena instalacije prolaza na lokaciji $j$ |
| $d_i$ | saobraćaj generisan od strane klijenta $i$ |
| $u_{jl}$ | kapacitet veze između lokacija $j$ i $l$ |
| $u_{jN}$ | kapacitet veze između lokacije $j$ i osnove (parametar $M$ iz originalnog modela koji je bio isti za sve lokacije) |
| $v_j$ | pristupni kapacitet lokacije $j$ |
| $a_{ij}$ | uređaj na lokaciji $j$ pokriva klijenta $i$ |
| $b_{jl}$ | lokacije $j$ i $l$ mogu biti povezane |
| | Promenljive odlučivanja |
| $x_{ij}$ | 1 ako je klijent $i$ pridružen lokaciji $j$, 0 inače |
| $z_j$ | 1 ako je uređaj instaliran na lokaciji $j$, 0 inače |
| $w_{jN}$ | 1 ako je prolaz instaliran na lokaciji $j$, 0 inače |
| $y_{jl}$ | 1 ako je uspostavljena veza između lokacije $j$ i $l$, 0 inače |
| $f_{jl}$ | protok saobraćaja na vezi između lokacija $j$ i $l$ |
| $f_{jN}$ | protok saobraćaja između lokacije $j$ i osnove |

Funkcija cilja (engl. *objective function*) predstavlja ukupnu cenu instalacije mreže. Ukupna cena je zbir troškova sa svih lokacija. Trošak na jednoj lokaciji predstavlja cenu instaliranja rutera na toj lokaciji i dodatnu cenu za instaliranje prolaza, ako je on instaliran na toj lokaciji. Prethodno definisana ograničenja imaju sledeća značenja:

2. Garantuje se pokrivenost svih klijenata. Svaki klijent je dodeljen tačno jednoj lokaciji.

3. Klijent može biti pridružen nekoj lokaciji samo ako je na toj lokaciji instaliran neki uređaj i ako ta lokacija pokriva tog klijenta.

4. Garantuje balans saobraćaja kroz neku lokaciju. Prva suma predstavlja ukupan sabraćaj svih klijenata pridruženih posmatranoj lokaciji, druga predstavlja razliku saobraćaja od susednih lokacija ka posmatranoj i od posmatrane lokacije ka susednim, na kraju se oduzima saobraćaj od posmatrane lokacije ka osnovi. Ukupna vrednost mora biti jednaka nuli za svaku lokaciju.

5. Ukupan saobraćaj između dve lokacije neće premašiti kapacitet veze između te dve lokacije.

6. Ukupan saobraćaj između klijenata, pridruženih nekoj lokaciji i te lokacije neće premašiti pristupni kapacitet te lokacije.

7. Saobraćaj od neke lokacije ka osnovi postoji samo ako je na toj lokaciji instaliran prolaz i da u tom slučaju ne može da premaši određeni kapacitet

8. Veza između dve lokacije može biti uspostavljena samo ako na obe lokacije postoji neki uređaj.

9. Veza između dve lokacije može biti uspostavljena samo ako to dozvoljava dati parametar.

10. Klijent mora biti pridružen najboljoj mogućoj lokaciji. Za svakog klijenta postoji uređen skup lokacija koje ga pokrivaju. Ako na nekoj lokaciji iz tog skupa postoji instaliran uređaj, onda klijent ne može biti pridružen drugoj lokaciji koja je u poretku posle posmatrane.

11. Određene promenljive mogu da uzimaju samo binarne vrednosti.

## II. Primena i načini rešavanja

Bežične meš mreže daju mogućnost različitih primena, kao što su umrežavanje u okviru preduzeća ili lokalne zajednice, sistema za nadzor, u slučaju vanrednih stanja ili na mestu nesreće, u ruralnim područjima i slično [2]. Problem planiranja BMM je privukao pažnju mnogih istraživača iz oblasti optimizacije. Većina radova se bavi višekriterijumskom optimizacijom (engl. *multi-objective optimization*), gde se uređaji mreže postavljaju s ciljem povećanja povezanosti mreže, tj. njene dzinovske komponente (engl. *giant component*) i pokrivenosti klijenata. Takav problem je rešavan tabu pretragom (engl. *tabu search*) [3], simuliranim kaljenjem (engl. *simulated annealing*) [4] i genetskim algoritmima (engl. *genetic algorithm*) [5]. Rad [6] koristi isto ograničenje zahtevanog protoka kao ovde predstavljeni model, ali s ciljem smanjivanja broja prolaza i rešavan je simuliranim kaljenjem i pretraživanjem usponom (engl. *hill climbing*). Dosta složeniji model se razmatra u radu [7] sa ciljem istovremene optimizacije instalacionih troškova i karakteristika mreže koje se odnose na protok podataka, problem je rešavan hibridom metode roja čestica (engl. *particle swarm optimization*) i genetskog algoritma. Međutim, koliko je autoru poznato, ne postoji rad koji se bavio rešavanjem prethodno predstavljenog modela metaheurističkim metodama. Autori rada u kojem je prvi put definisan ovakav model su problem rešavali heuristikom zasnovanom na linearnoj relaksaciji, ali su eksperimentalne rezultate prikazali za drugi, prošireni model [1].

U nastavku rada se konstruišu dve metaheurističke metode, genetski algoritam i metoda promenljivih okolina (engl. *variable neighborhood search*), kao i njihova hibridizacija. Na kraju se prikazuju ekspreminetalni rezultati na generisanim instancama i vrši njihova analiza.

## II. Metoda promenljivih okolina

Osnovna ideja metode promenljivih okolina jeste sistematična promena okolina prilikom lokalne pretrage [8]. Deterministička promena okolina u okviru lokalne pretrage daje metod promenljivog spusta (engl. *variable neighborhood descent*), čiji su osnovni koraci:

- *Inicijalizacija.* Izbor skupa okolina $N_k, k = 1, ..., k_{max}$. Konstruisanje početnog rešenja $x$.
- Ponavljanje narednih koraka sve dok ima poboljšanja:
  1. Postaviti $k \leftarrow 1$;
  2. Ponavljati naredne korake sve dok je $k < k_{max}$:
     (a) *Istraživanje okoline.* Nađi najboljeg suseda $x'$ rešenja $x$ u okolini $N_k(x)$;
     (b) *Prihvatanje rešenja.* Ako je tako dobijeno rešenje $x'$ bolje od $x$, postavi $x \leftarrow x'$; inače, postavi $k \leftarrow k + 1$;

Osnovna varijanta metode promenljivih okolina (engl. *basic variable neighborhood search*) uključuje stohastičku komponentu koja se naziva razmrdavanje (engl. *shaking*). Cilj ove komponente je da spreči da se pretraga zaglavi u nekom lokalnom optimumu. Osnovni koraci ove varijante su:

- *Inicijalizacija.* Izbor skupa okolina $N_k, k = 1, ..., k_{max}$, koji se koristi u fazi razmrdavanja. Konstruisanje početnog rešenja $x$. Izbor kriterijuma zaustavljanja.
- Ponavljanje narednih koraka sve dok se ne ispuni kriterijum zaustavljanja:

1. Postaviti $k \leftarrow 1$;
2. Ponavljati naredne korake sve dok je $k < k_{max}$:
   (a) *Razmrdavanje.* Generisanje slučajne tačke $x'$ iz okoline $N_k(x)$;
   (b) *Lokalna pretraga.* Primeni neku metodu lokalne pretrage sa početnim rešenjem $x'$, rezultat pretrage označi sa $x''$;
   (c) *Prihvatanje rešenja.* Ako je tako dobijeno rešenje $x''$ bolje od $x$, postavi $x \leftarrow x''$ i $k \leftarrow 1$; inače, postavi $k \leftarrow k + 1$;

Kada se u osnovnoj varijanti kao lokalna pretraga koristi metoda promenljivog spusta, dobija se opšta metoda promenljivih okolina (engl. *general variable neighborhood search*). Metoda promenljivog spusta se izvršava kao nezavisna procedura, tj. ima poseban skup okolina, a kao ulaz dobija početno rešenje.

## I. Reprezentacija rešenja

Parcijalno rešenje $r$ se predstavlja nizom $N$ brojeva iz skupa $\{0,1,2\}$. Pozicije u nizu odgovaraju rednim brojevima lokacija, a $N$ je ukupan broj lokacija. Broj 1 na poziciji $i$ označava da je instaliran ruter na $i$-toj lokaciji, 2 da je instaliran prolaz, a 0 da nema instaliranih uređaja na toj lokaciji. Tako su određene promenljive odlučivanja $z$ i $w$. Na osnovu opisane reprezentacije, funkcija cilja se izračunava jednostavno jednačinom 12.

$$f(x) = \sum_{i=0}^{N-1} cost(i),\ cost(i) = \begin{cases} c_i, & r[i] = 1 \\ c_i + p_i, & r[i] = 2 \\ 0, & r[i] = 0 \end{cases} \tag{12}$$

Na osnovu parcijalnog rešenja i instance moguće je konstruisati celo rešenje. Prvo se vrši pridruživanje klijenata lokacijama, tj. određivanje vrednosti promenljive $x$. Na osnovu ograničenja 2, 3, 6 i 10, klijent se pridružuje prvoj lokaciji sa instaliranim uređajem iz uređenog skupa lokacija koje ga pokrivaju, ako takva lokacija nije već popunila svoj pristupni kapacitet. Pri tome je moguće da ne postoji odgovarajuća lokacija kojoj dati klijent može da se pridruži, u tom slučaju se rešenje označava nedopustivim.

Kada je obezbeđena pokrivenost svih klijenata, prelazi se na ostvarivanje zahtevanog protoka kroz mrežu, tj. na određivanje vrednosti promenljivih $y_{jl}$, $f_{jl}$, $f_{jN}$. Na osnovu ograničenja 4, 5, 7, 8 i 9, problem možemo da svedemo na određivanje maksimalnog protoka kroz transportnu mrežu (engl. *maximum flow problem*). Za svaku lokaciju, na osnovu pridruženih klijenata, znamo ukupnu zahtevanu količinu protoka. Takve lokacije označavamo kao izvore (engl. *source*), dok lokacije na kojima su instalirani prolazi označavamo ponorima (engl. *sink*) i kod kojih posmatramo kapacitet veze ka osnovi mreže. Potrebno je utvrditi da li je moguće ostvariti zahtevani protok od klijenata do osnove mreže. Kako algoritmi za problem maksimalnog protoka rade sa mrežama sa jednim izvorom i jednim ponorom, dodatno transformišemo mrežu. Od svake lokacije se ka izvoru dodaje grana sa kapacitetom jednakim zahtevanom protoku te lokacije, slično se i od svakog prolaza dodaje grana ka ponoru sa kapacitetom jednakim kapacitetu veze prolaza ka osnovi mreže. Ako je maksimalni protok kroz takvu mrežu jednak ukupnom zahtevanom, rešenje je dopustivo. Pri tome se određuju vrednosti protoka na pojedinačnim vezama. Maksimalni protok se može odrediti u polinomijalnom vremenu nekim od poznatih algoritama, kao što je Ford-Fulkersonov algoritam. Treba napomenuti da takvi algoritmi obično rade sa celobrojnim protokom (odgovara ovom problemu jer se radi o digitalnim podacima), dok je u matematičkoj formulaciji problema dozvoljeno da protok uzima realne vrednosti (tako je omogućena efikasnija primena egzaktnih rešavača). Cilj određivanja protoka u ovom modelu je utvrđivanje dopustivosti rešenja, dok se za realno rutiranje saobraćaja kroz mrežu nakon instalacije koriste druge tehnike.

## II. Okoline

Prvi korak u konstruisanju algoritma za predstavljeni problem jeste izbor okolina. Jedna grupa okolina se odnosi na operacije sa ruterima, a druga sa prolazima. Pored toga, okoline se mogu podeliti prema tome da li se koriste u fazi razmrdavanja ili fazi lokalne pretrage. Za fazu razmrdavanja se koriste sledeće okoline:

- *Ukloni ruter* ($N_1$): Susedno rešenje se dobija tako što se ukloni slučajno izabrani ruter. Ako se na taj način dobija skup nepokrivenih klijenata, slučajnim redosledom se dodaju ruteri na lokacije koje mogu da pokriju te klijente, sve dok se ne pokriju svi klijenti. Zatim, ako postoji neostvaren protok, na susedne lokacije se dodaju ruteri sve dok se zahtevani protok ne ostvari.
- *Premesti prolaz* ($N_2$): Susedno rešenje se dobija tako što se slučajno izabrani prolaz premesti na slučajno izabranu lokaciju.
- *Ukloni\premesti* ($N_3$): Ova okolina predstavlja kombinaciju prethodne dve. Susedno rešenje se dobija tako što se prvo primeni postupak opisan za okolinu $N_1$, a zatim na dobijeno rešenje postupak za okolinu $N_2$.

Okoline veličine $k$ se dobijaju ponavljanjem prethodnih postupaka $k$ puta. U okviru lokalne pretrage koriste se sledeće okoline:

- *Ukloni nepotrebni ruter* ($N_4$): Ukloni se ruter koji nije potreban.
- *Premesti ruter* ($N_5$): Ukloni se ruter sa jedne lokacije i premesti na novu lokaciju gde je cena instalacije rutera niža. Kada se ukloni ruter sa originalne pozicije dobija se skup nepokrivenih klijenata. Za novu lokaciju rutera razmatraju se samo lokacije koje pokrivaju bar jednog klijenta tog skupa, a ako je takav skup prazan, samo susedne lokacije.
- *Postavi 1 ukloni 2* ($N_6$): Postavlja se novi ruter na neku lokaciju, zatim se uklanjaju dva postojeća rutera. Razmatraju se samo lokacije koje pokrivaju bar jednog klijenta iz skupa klijenata koje pokriva novi ruter ili susedne lokacije ako je takav skup prazan.
- *Ukloni 1 postavi 2* ($N_7$): Ukloni se jedan ruter, a zatim se postave dva čiji je zbir cena instalacije manji od prvog. Kada se ukloni ruter sa originalne pozicije dobija se skup nepokrivenih klijenata. Za lokacije dva nova rutera razmatraju se samo lokacije koje pokrivaju bar jednog klijenta tog skupa, a ako je takav skup prazan, samo susedne lokacije.
- *Ukloni nepotrebni prolaz* ($N_8$): Ukloni se prolaz koji nije potreban.
- *Premesti prolaz* ($N_9$): Postojeći prolaz na nekoj lokaciji se premešta na susednu lokaciju, ako se pri tome cena rešenja smanjuje. Moguće je da se prolaz premesti na lokaciju na kojoj već postoji ruter ili na praznu lokaciju. Pri tome se može ukloniti i ruter sa originalne pozicije.

Sve gore navedene okoline sadrže samo dopustiva rešenja, pri generisanju susednog rešenja proverava se pokrivenost klijenata i ostvareni protok, ako rešenje nije dopustivo traži se novo. Lokacije su susedne ako instanca sadrži vezu između njih.

## III. Prilagođena opšta metoda promenljivih okolina

Početno rešenje se konstruiše tako što se prvo na sve lokacije postave ruteri, a zatim se prolazi postavljaju pohlepnom heuristikom. U svakom koraku se postavlja prolaz sa minimalnim odnosom cene i količine novog protoka, sve dok se ne ostvari zahtevani protok kroz mrežu. U okviru metode promenljivog spusta, okoline se istražuju sledećim redosledom: $N_4, N_8, N_5, N_9, N_6, N_7$. Razmrdavanje se vrši sledećim redosledom: $N_1$, $N_2$,

$N_3$. Pri tome su okoline $N_1$ i $N_2$ veličine 1, a $N_3$ je veličine od 1 do $k_{max} - 2$. Ovaj algoritam je označen sa VNS1.

Prilikom istraživanja okolina, često proveravanje ostvarenog protoka kroz mrežu je vremenski zahtevno, pogotovo ako je mreža gusta. Kako bi algoritam bio efikasniji moguće je rešavanje podeliti u dve faze. U prvoj fazi se određuje optimalno postavljanje rutera uz ograničenje pokrivenosti klijenata. U početnom rešenju su na svim pozicijama postavljeni ruteri, metoda promenljivih okolina za razmrdavanje koristi samo okolinu $N_1$ veličine od 1 do $k_{max}$, a u okviru lokalne pretrage se koriste okoline $N_4$, $N_5$, $N_6$ i $N_7$. Ovaj algoritam je označen sa VNS2. Zatim se pohlepnim algoritmom takvom rešenju dodaju prolazi i ruteri sve dok rešenje ne postane dopustivo u odnosu na ograničenje protoka. U svakom koraku se prvo dodaje uređaj čiji je odnos cene i količine novog protoka minimalan. Tako dobijeno rešenje se koristi kao početno rešenje za algoritam VNS1, tj. drugu fazu kompletnog algoritma koji je označen sa VNS.

Prethodno opisani algoritmi imaju sledeće zajedničke karakteristike. Kriterijum zaustavljanja je određen kombinacijom maksimalnog broja iteracija $I_{max}$ i maksimalnog broja uzastopnih iteracija bez poboljšanja rešenja $S_{max}$. U okviru lokalne pretrage, istraživanje okoline se vrši metodom prvog poboljšanja (engl. *first improvement*) i vrši se slučajnim redosledom [9]. Rešenje dobijeno lokalnom pretragom se uvek prihvata ako je bolje od trenutno najboljeg, a sa verovatnoćom $p$ se prihvata ako ima istu vrednost funkcije cilja, a različit kod , uz ograničen broj uzastopnih takvih prihvatanja na $c_{max}$ [10].

## III. Genetski algoritam

Osnovna ideja ove metaheuristike jeste simuliranje procesa prirodne evolucije. Tokom vremena, prirodne populacije evoluiraju i prilagođavaju se okruženju. Bolje prilagođene jedinke imaju veću šansu da prežive i učestvuju u razmnožavanju, a time i da prenesu svoj genetski materijal u narednu generaciju. Tako slabije jedinke i njihov genetski materijal postepeno nestaju iz populacije.

Genetski algoritam polazi od inicijalne populacije rešenja (jedinki), a zatim iterativno primenjuje genetske operatore na jedinke populacije. Prilagođenost (engl. *fitness function*) jedinke se računa na osnovu funkcije cilja. Jedinke izabrane na osnovu prilagođenosti se ukrštaju (engl. *crossover*) i stvaraju nove jedinke, koje imaju karakteristike oba roditelje. Očekuje se da se ukrštanjem dve kvalitetne jedinke nekad može dobiti još bolja jedinka. Dodatno, vrši se mutacija nekih gena jedinki, kako bi se očuvala raznovrsnost genetskog materijala. Tako dobijene jedinke zamenjuju celu prethodnu populaciju ili njen manje prilagođeni deo [11]. Osnovni koraci su:

- Generisanje inicijalne populacije
- Ponavljati naredne korake dok se ne ispuni kriterijum zaustavljanja
    1. Izračunati prilagođenost svake jedinke
    2. Selekcija
    3. Ukrštanje
    4. Mutacija

### I. Prilagođeni genetski algoritam

Reprezentacija rešenja je ista kao kod metode promenljivih okolina. Početna populacija se generiše slučajno i sadrži $P_{size}$ jedinki. Način generisanja je određen sa dva parametra, $p_{router}$ i $p_{gateway}$. Prvi označava verovatnoću da se na nekoj poziciji postavi ruter, a drugi da se postavi prolaz. Moguće je da neka rešenja ne budu dopustiva, jer nisu svi klijenti pokriveni. Takva rešenja se popravljaju pohlepnim algoritmom. Algoritam u svakom koraku postavlja ruter na poziciju koja ima minimalan odnos cene instalacije rutera i broja novih pokrivenih klijenata. [11]. Na kraju se još jednom prolazi kroz rešenje i uklanjaju se

nepotrebni ruteri, ali kako se tako može ukloniti ruter potreban za protok, ovaj korak se vrši sa određenom verovatnoćom $p_{remove}$. Rešenje i dalje može biti nedopustivo zbog neostvarenog protoka, u ovom slučaju, količina neostvarenog protoka se dodaje kao kazna (engl. *penalty function*) [12] funkciji cilja, na osnovu formule 13, gde je $F(x)$ nova funkcija cilja, $f(x)$ stara funkcija cilja, $flow(x)$ količina neostvarenog protoka, a $t$ parametar koji određuje jačinu kazne.

$$F(x) = f(x) + t \cdot flow(x) \tag{13}$$

Kriterijum zaustavljanja je određen kombinacijom maksimalnog broja generacija $I_{max}$ i maksimalnog broja uzastopnih generacija bez poboljšanja rešenja $S_{max}$. Algoritam koristi sledeće genetske operatore:

- *Selekcija.* Koristi se turnirska selekcija (engl. *tournament selection*). Iz populacije se na slučajan način bira $T_{size}$ jedinki, zatim se od izabranih se uzima najbolja. Tako se dobija jedan novi roditelj, a postupak se ponavlja onoliko puta koliko je roditelja potrebno.
- *Ukrštanje.* Dva roditelja se ukrštaju jednopozicionim ukrštanjem (engl. *one-point crossover*) i daju dve nove jedinke. Na slučajan način se bira jedna tačka ukrštanja. Prvo dete uzima deo rešenja do tačke ukrštanja od prvog roditelja, a drugi deo od drugog roditelja, drugo dete obrnuto. Ukrštanje se vrši sa verovatnoćom $p_{crossover}$, dok sa verovatnoćom $1 - p_{crossover}$ sami roditelji idu u narednu fazu.
- *Mutacija.* Do mutacije nekog gena dolazi sa verovatnoćom $p_{mutate}$. Mutacija zavisi od vrednosti gena. Na praznu lokaciju se postavlja ruter. Ako je na poziciji bio ruter, sa verovatnoćom $m_{router}$ se uklanja, a sa verovatnoćom $1 - m_{router}$ se postavlja prolaz, a ako je na poziciji bio prolaz, sa verovatnoćom $m_{gateway}$ se premešta na drugu slučajno izabranu poziciju, a sa verovatnoćom $1 - m_{gateway}$ se uklanja.

Prethodno opisani operatori ne čuvaju dopustivost rešenja, a taj problem se rešava na isti način kao i kod generisanja inicijalne populacije. Kako bi se najbolje jedinke sačuvale za sledeću generaciju uvodi se elitizam. $E_{count}$ najboljih jedinki se prenosi u narednu generaciju bez promena, takve jedinke se nazivaju elitnim. Konvergencijom algoritma može doći do smanjivanja raznovrsnosti genetskog materijala populacije. Može se desiti da neke pozicije gena jedinki imaju iste vrednosti kod velikog dela populacije, takve pozicije se nazivaju zaleđenim (engl. *frozen bits*) [13]. Zato se uvodi nova verovatnoća mutacije $p_{frozen}$ za ovakve gene, uz parametar $f_{threshold}$ koji označava koliko procentualno jedinki populacije mora da ima istu vrednost pozicije kako bi ona bila označena zaleđenom. Pored toga, duplikati jedinki se uklanjaju iz populacije.

## IV. Hibridizacija

Jedan od načina da se izvrši hibridizacija dve opisane metaheuristike jeste da se rešenja genetskog algoritma iskoriste kao početna rešenja metode promenljivih okolina. Na kraju izvršavanja genetskog algoritma određeni broj jedinki poslednje populacije se poboljšava. Kako izvršavanje ne bi postalo previše zahtevno, poboljšavaju se samo dve jedinke. Prva je najbolja jedinka populacije, a druga se bira iz prvih 20 jedinki, kao ona jedinka koja je na najvećem rastojanju od prve. Rastojanje se računa kao broj pozicija na kojima se vrednosti rešenja razlikuju. Pre pokretanja metode promenljivih okolina, prolazi se uklanjaju iz rešenja. Kako se na taj način ne bi izgubilo najbolje rešenje genetskog algoritma, ono se prvo sačuva. Kao rezultat se vraća najbolje od ova tri rešenja.

## V. Eksperimentalni rezultati

Opisane metaheuristike su implementirane u programskom jeziku *C++*. Za problem određivanja maksimalnog protoka korišćena je *Boost* biblioteka za rad sa grafovima i Bojkov-Kolmogorov algoritam [14, 15]. Kao egzaktni rešavač korišćen je *IBM ILOG CPLEX Teaching Edition 12.1*, čije je vreme izvršavanja ograničeno na 4 sata. Instance su rešavane na računaru sa procesorom AMD A6 2.70 GHz i 8 GB RAM memorije.

Za svaku instancu je prikazano rešenje egzaktnog rešavača, sa zvezdicom su označena rešenja za koje nije dokazana optimalnost posle isteka vremenskog ograničenja, dok je sa crticom označeno da rešavač nije uspeo da nađe rešenje usled nedostatka memorije. Svaka metaheuristika je pokretana 10 puta za instance malih i srednjih dimenzija, a 5 puta za instance velikih dimenzija. Na osnovu tih pokretanja izračunate su sledeće vrednosti: najbolje rešenje *best*, prosečno ukupno vreme izvršavanja $t_{tot}$, prosečno početno vreme $t_{best}$, prosečno procentualno odstupanje od najboljeg rešenja *agap* i standardna devijacija tog odstupanja $agap_\sigma$. Dodatno, za genetski algoritam je naveden i prosečan broj generacija $gen_{avg}$. Rešenje *best* je podebljano ako je bolje od rešenja dobijenog egzaktnim rešavačem.

## I. Instance

Instance problema su generisane po uzoru na situacije iz prakse [1]. Kreiran je i generator instanci u programskom jeziku *C#*. Aplikacija omogućava zadavanje različitih parametara na osnovu kojih se generišu instance, kao i grafički prikaz generisanih instanci i rešenja problema. Parametri za generisanje instanci su dati tabelom 2. Vrednosti parametara su izražene u proizvoljnim jedinicama. Na primer za parametre $a$, $d$ i $s$ mogu se uzeti metri, a za $t$, $c_1$, $c_2$, $c_3$ Mbps. Vrednost funkcije cilja može se računati u stotinama dolara.

**Tabela 2:** *Opis parametara za generisanje instanci*

| | |
|---|---|
| $N$ | Broj kandidatskih lokacija |
| $M$ | Broj klijenata |
| $a$ | Dužina stranice kvadrata na kojem će biti raspoređene kandidatske lokacije i klijenti |
| $d$ | Domet veze između dve lokacije |
| $s$ | Domet pristupne veze lokacija |
| $r$ | Odnos između cene instaliranja rutera i prolaza |
| $t$ | Saobraćaj koji zahteva svaki klijent |
| $c_1$ | Kapacitet pristupne veze svake lokacije |
| $c_2$ | Kapacitet veze između svake dve lokacije |
| $c_3$ | Kapacitet veze između svake lokacije i osnove. |

Prvo se generiše $N$ lokacija. Za svaku koordinatu se uzima slučajan ceo broj iz intervala $[0, a]$. Klijenti se postavljaju tako da instanca bude dopustiva. Za svakog klijenta prvo se na slučajan način bira neka lokacija, zatim se klijent na slučajan način postavlja u domet te lokacije, tj. u krug prečnika $s$ sa centrom u toj lokaciji. Tako je svaki klijent pokriven bar jednom lokacijom. Za svakog klijenta, nalazimo lokacije koje su na rastojanju manjem od $s$ i sortiramo ih neopadajuće po tom rastojanju (kod realnih instanci, sortiranje bi se vršilo po jačini signala). Dalje, za svake dve lokacije koje su na rastojanju manjem od $d$ pravimo jednu vezu. Cene instalacije rutera i prolaza se lako određuju na osnovu parametra $r$.

Prethodno opisanim načinom generisana je prva grupa od ukupno 60 instanci, podeljenih u tri podgrupe od po 20 instanci. Podgrupe su formirane prema veličini instanci, redom male, srednje i velike. Pri generisanju instanci menjani su parametri $N$, $M$, $a$, $r$, $t$. Parametri $d$, $s$, $c_1$, $c_2$, $c_3$ su fiksirani za sve instance i uzimaju vrednosti redom 250, 100, 54, 54, 128. Ovako generisane instance imaju iste cene uređaja po svim lokacijama (engl. *unicost*), jedino se razlikuje odnos cena rutera i prolaza.

Druga grupa instanci je genirasana na isti način, ali sa manje fiksiranih parametara. Instance iz ove grupe imaju uniformnu raspodelu cena instalacije uređaja po lokacijama, za rutere iz intervala [0.1, 5], a za prolaze iz [7, 11], sa korakom 0.1. Pored toga, ima i uniformnu raspodelu vrednosti parametara $t$, $c_2$, $c_3$ iz skupova vrednosti redom {1, 2, 3}, {18, 36, 54, 72, 90}, {32, 64, 96, 128, 160}.

## II. Analiza rezultata

Prvo se analiziraju rezultati implementiranih metaheuristika na prvoj grupi instanci. Vrednosti parametara metode promenljivih okolina date su u tabeli 3, a rezultati su prikazani u tabelama 5, 6 i 7. Kod ovakvih instanci, kada su ruteri postavljeni i svi klijenti pokriveni, postavljanje uređaja za ostvarivanje protoka je relativno jednostavno, stoga se parametrima pojačava prva faza algoritma, dok se druga skraćuje. Na instancama malih i srednjih dimenzija, algoritam pronalazi optimalno rešenje na svim instancama u svih 10 pokretanja. Kod skoro svih instanci srednjih dimenzija vreme izvršavanja je kraće nego kod CPLEX-a. Za instance velikih dimenzija, algoritam bar jednom pronalazi rešenje jednako ili bolje od rešenja CPLEX-a. Pri tome vreme izvršavanja ne prelazi 15 minuta, a vrednost $agap$ je uvek manja od 1%, dok je $agap_\sigma$ uvek manja od 0.5.

Vrednosti nekih parametara genetskog algoritma su date tabelom 4, a rezultati su prikazani u tabelama 8, 9 i 10. Različite vrednosti parametra $I_{max}$ su postavljene za male, srednje i velike instance, redom 200, 500, 2000. Isto i za parametar $S_{max}$, redom 30, 100, 250. Algoritam pronalazi optimalna rešenja kod svih instanci malih dimenzija, ali rezultati nisu stabilni kao kod VNS-a. Kod 4 instance srednjih dimenzija, ne pronalazi se optimalno rešenje. Mogući uzrok tome je povećan zahtevani protok kod tih instanci, što daje veću verovatnoću generisanja nedopustivih instanci, tj. instanci kod kojih zahtevani protok nije ostvaren. Rezultati na instancama velikih dimenzija su lošiji od rezultata VNS-a, a kod tri instance dobijeni su rezultati bolji od CPLEX-a.

Hibridna metaheuristika ima za cilj poboljšanje rešenja VNS-a, stoga je testirana samo na instancama velikih dimenzija. Parametri pojedinačnih metoda hibridnog algoritma su isti kao prikazani u tabelama 3 i 4, s tim što je za GA parametar $I_{max}$ postavljen na 500, a $S_{max}$ na 50. Rezultati su prikazani u tabeli 11. Na 6 instanci su postignuti bolji rezultati u odnosu na VNS, a na dve instance lošiji. Cena ovog poboljšanja jeste duže vreme izvršavanja, koje sada ide i do 35 minuta.

Instance iz druge grupe su generisane sa više slučajnosti i pokazale su se težim za rešavanje. Za ove instance, parametri metode promenljivh okolina prikazani su u tabeli 3, a rezultati u tabeli 12. Kod ovakvih instanci potrebno je pojačati drugu fazu algoritma, što dovodi do dužeg vremena izvršavanja. Poređenjem vremena $t_{tot}$ i $t_{best}$, ispostavlja se da je parametar $S_{max}$ moguće postaviti na manju vrednost. U nekim slučajevima početno rešenje, dobijeno u prvoj fazi, predstavlja lokalni optimum iz kojeg algoritam druge faze ne uspeva da izađe. To može biti uzrok lošijim rezultatima u odnosu na prvu grupu instanci. Od prvih 5 instanci, algoritam ne uspeva da pronađe optimalno rešenje za dve instance, iako su one malih dimenzija. Kod drugih 5 većih instanci, algoritam za samo jednu instancu daje isto rešenje kao CPLEX, a za ostale približno rešenje. Mogući uzrok ovakvom ponašanju jeste razmrdavanje u okolini $N_2$, ograničeno samo na dopustivi deo okoline. Jedno rešenje bi bilo razmrdavanje u celoj okolini uz popravljanje rešenja. Međutim, prethodno opisani pohlepni algoritam je neefikasan za popravljanje pri svakom razmrdavanju.

Parametri genetskog algoritma za ovu grupu instanci su isti kao u tabeli 4, dok su $I_{max}$ i $S_{max}$ postavljeni redom na 2000 i 200. Rezulati, prikazani u tabeli 13, su na većini instanci lošiji nego kod VNS-a. Međutim, za instancu 3, GA postiže optimalno rešenje, za razliku od VNS-a. Mogući razlog je to što genetski algoritam dopušta i pretragu nedopustivog dela prostora rešenja, jer se jedinke sa neostvarenim protokom ne odbacuju.

Hibridna metaheuristika za ove instance ima iste vrednosti parametara kao i pojedinačne metode, a za GA parametri $I_{max}$ i $S_{max}$ su postavljeni redom na 1000 i 50. Rezultati su prikazani u tabeli 14. Hibridni algoritam je postigao jednaka ili bolja rešenja od pojedinačnih metaheuristika na svim instancama ove grupe.

## VI. Zaključak

Na osnovu prikazanih rezultata, dolazi se do zaključka da su implementirane metode pogodne za rešavanje problema optimalnog planiranja bežičnih meš mreža. Za instance koje nisu rešene egzaktnim rešavačem, pronađena su rešenja u razumnom vremenu. Neka rešenja za koja nije dokazana optimalnost su poboljšana. Metoda promenljivih okolina se pokazala kao bolji način rešavanja u odnosu na genetski algoritam, dok su

**Tabela 3:** *Vrednosti parametara metode promenljivih okolina*

| Parametar | male | | srednje i velike | | druga grupa | |
|---|---|---|---|---|---|---|
| | VNS2 | VNS1 | VNS2 | VNS1 | VNS2 | VNS1 |
| $I_{max}$ | 250 | 100 | 250 | 100 | 100 | 250 |
| $S_{max}$ | 15 | 5 | 30 | 15 | 30 | 50 |
| $k_{max}$ | 5 | 5 | 7 | 5 | 5 | 5 |
| $p$ | 0.6 | 0.4 | 0.6 | 0. 4 | 0.6 | 0.6 |
| $c_{max}$ | 10 | 5 | 10 | 5 | 10 | 5 |

**Tabela 4:** *Vrednosti parametara genetskog algoritma. $N$ je broj kandidatskih lokacija instance.*

| Parametar | Vrednost | Parametar | Vrednost |
|---|---|---|---|
| $P_{size}$ | 100 | $p_{frozen}$ | $2/N$ |
| $T_{size}$ | 3 | $f_{threshold}$ | 0.9 |
| $p_{crossover}$ | 0.8 | $p_{remove}$ | 0.5 |
| $p_{mutate}$ | $1/N$ | $E_{count}$ | 12 |
| $m_{router}$ | 0.8 | $p_{router}$ | 0.3 |
| $m_{gateway}$ | 0.8 | $p_{gateway}$ | 0.1 |
| $t$ | 10 | | |

neka rešenja dodatno poboljšana hibridizacijom ove dve metode. Metode su dale bolje rezultate na instancama iz prve grupe. Rezultati se mogu dalje unaprediti dodatnim podešavanjem parametara za genetski agloritam i poboljšanjem faze razmrdavanja metode promenljivih okolina.

## Bibliografija

[1] Amaldi, E., Capone, A., Cesana, M., Filippini, I. and Malucelli, F., 2008. Optimization models and methods for planning wireless mesh networks. *Computer Networks*, 52(11), pp.2159-2171.

[2] Akyildiz, I. and Wang, X., 2009. *Wireless mesh networks* (Vol. 3). John Wiley & Sons.

[3] Xhafa, F., Sánchez, C., Barolli, A. and Takizawa, M., 2015. Solving mesh router nodes placement problem in Wireless Mesh Networks by Tabu Search algorithm. *Journal of Computer and System Sciences*, 81(8), pp.1417-1428

[4] Sakamoto, S., Kulla, E., Oda, T., Ikeda, M., Barolli, L. and Xhafa, F., 2014. Performance evaluation considering iterations per phase and SA temperature in WMN-SA system. *Mobile Information Systems*, 10(3), pp.321-330.

[5] Xhafa, F., Sánchez, C. and Barolli, L., 2010, April. Genetic algorithms for efficient placement of router nodes in wireless mesh networks. In *2010 24th IEEE International Conference on Advanced Information Networking and Applications* (pp. 465-472). IEEE.

[6] Nawaf, L., Mumford, C. and Allen, S., 2015, September. Optimizing the Placement of ITAPs in Wireless Mesh Networks by Implementing HC and SA Algorithms. In *International Conference on Ad Hoc Networks* (pp. 29-41). Springer International Publishing.

[7] Benyamina, D., Hafid, A., Hallam, N., Gendreau, M. and Maureira, J.C., 2012. A hybrid nature-inspired optimizer for wireless mesh networks design. *Computer Communications*, 35(10), pp.1231-1246.

**Tabela 5:** *Rezultati metode promenljivih okolina na instancama malih dimenzija*

| Instanca | $N$ | $M$ | CPLEX | | VNS | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $t_{best}(s)$ | $agap(\%)$ | $agap_{\sigma}$ |
| small1.txt | 20 | 50 | 31.000 | 0.044 | opt | 0.071 | 0.028 | 0.000 | 0.000 |
| small2.txt | 40 | 100 | 50.000 | 0.071 | opt | 0.319 | 0.210 | 0.000 | 0.000 |
| small3.txt | 60 | 150 | 65.000 | 0.390 | opt | 0.863 | 0.574 | 0.000 | 0.000 |
| small4.txt | 80 | 180 | 89.000 | 2.887 | opt | 1.021 | 0.773 | 0.000 | 0.000 |
| small5.txt | 100 | 250 | 99.000 | 1.933 | opt | 1.843 | 1.452 | 0.000 | 0.000 |
| small6.txt | 20 | 50 | 29.000 | 0.053 | opt | 0.139 | 0.091 | 0.000 | 0.000 |
| small7.txt | 20 | 50 | 38.000 | 0.109 | opt | 0.139 | 0.091 | 0.000 | 0.000 |
| small8.txt | 20 | 50 | 47.000 | 0.112 | opt | 0.153 | 0.099 | 0.000 | 0.000 |
| small9.txt | 20 | 50 | 56.000 | 0.141 | opt | 0.143 | 0.091 | 0.000 | 0.000 |
| small10.txt | 20 | 50 | 20.000 | 0.038 | opt | 0.127 | 0.089 | 0.000 | 0.000 |
| small11.txt | 40 | 100 | 49.000 | 0.071 | opt | 0.370 | 0.264 | 0.000 | 0.000 |
| small12.txt | 40 | 100 | 51.000 | 0.095 | opt | 0.247 | 0.171 | 0.000 | 0.000 |
| small13.txt | 40 | 100 | 47.000 | 0.088 | opt | 0.569 | 0.478 | 0.000 | 0.000 |
| small14.txt | 40 | 100 | 41.000 | 2.358 | opt | 0.903 | 0.825 | 0.000 | 0.000 |
| small15.txt | 40 | 100 | 38.000 | 6.263 | opt | 0.830 | 0.760 | 0.000 | 0.000 |
| small16.txt | 60 | 150 | 68.000 | 1.585 | opt | 0.589 | 0.450 | 0.000 | 0.000 |
| small17.txt | 60 | 150 | 44.000 | 0.421 | opt | 0.529 | 0.389 | 0.000 | 0.000 |
| small18.txt | 60 | 150 | 36.000 | 0.247 | opt | 0.511 | 0.377 | 0.000 | 0.000 |
| small19.txt | 60 | 150 | 33.333 | 0.278 | opt | 0.583 | 0.440 | 0.000 | 0.000 |
| small20.txt | 60 | 150 | 32.400 | 0.194 | opt | 0.615 | 0.422 | 0.000 | 0.000 |

[8] Hansen, P. and Mladenović, N., 2001. Variable neighborhood search: Principles and applications. *European journal of operational research*, 130(3), pp.449-467.

[9] Hoos, H.H. and Stützle, T., 2004. *Stochastic local search: Foundations & applications*. Elsevier.

[10] Dražić, Z., 2012. Variable Neighborhood Search for the File Transfer Scheduling Problem. *Serdica Journal of Computing*, 6(3), pp.333p-348p.

[11] Beasley, J.E. and Chu, P.C., 1996. A genetic algorithm for the set covering problem. *European Journal of Operational Research*, 94(2), pp.392-404.

[12] Yeniay, Ö., 2005. Penalty function methods for constrained optimization with genetic algorithms. *Mathematical and Computational Applications*, 10(1), pp.45-56.

[13] Stanimirović, Z., 2012. A genetic algorithm approach for the capacitated single allocation p-hub median problem. *Computing and Informatics*, 29(1), pp.117-132.

[14] Boost.org. (2016). *Boost C++ Libraries*. [online] Available at: http://www.boost.org/ [Accessed 24 Jun. 2016].

[15] Boost.org. (2016). *Boost Graph Library: Boykov-Kolmogorov Maximum Flow - 1.60.0.* [online] Available at: http://www.boost.org/doc/libs/1_60_0/libs/graph/doc/boykov_kolmogorov_max_flow.html [Accessed 24 Jun. 2016].

**Tabela 6:** *Rezultati metode promenljivih okolina na instancama srednjih dimenzija*

| Instanca | $N$ | $M$ | CPLEX | | VNS | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $t_{best}(s)$ | $agap(\%)$ | $agap_\sigma$ |
| medium1.txt | 150 | 400 | 154.000 | 24.322 | opt | 5.965 | 5.279 | 0.000 | 0.000 |
| medium2.txt | 200 | 500 | 192.000 | 39.584 | opt | 8.625 | 7.473 | 0.000 | 0.000 |
| medium3.txt | 250 | 600 | 236.000 | 169.704 | opt | 10.505 | 8.742 | 0.000 | 0.000 |
| medium4.txt | 300 | 700 | 270.000 | 174.961 | opt | 17.369 | 15.375 | 0.000 | 0.000 |
| medium5.txt | 350 | 900 | 328.000 | 243.509 | opt | 22.588 | 19.594 | 0.000 | 0.000 |
| medium6.txt | 250 | 600 | 143.000 | 457.158 | opt | 10.675 | 10.034 | 0.000 | 0.000 |
| medium7.txt | 250 | 600 | 233.000 | 74.857 | opt | 11.876 | 10.572 | 0.000 | 0.000 |
| medium8.txt | 250 | 600 | 359.000 | 147.344 | opt | 14.839 | 11.832 | 0.000 | 0.000 |
| medium9.txt | 250 | 600 | 485.000 | 62.819 | opt | 15.390 | 11.849 | 0.000 | 0.000 |
| medium10.txt | 250 | 600 | 611.000 | 91.045 | opt | 19.125 | 14.307 | 0.000 | 0.000 |
| medium11.txt | 150 | 400 | 150.000 | 37.116 | opt | 5.917 | 5.319 | 0.000 | 0.000 |
| medium12.txt | 150 | 400 | 143.000 | 74.970 | opt | 6.497 | 5.936 | 0.000 | 0.000 |
| medium13.txt | 150 | 400 | 143.000 | 114.607 | opt | 10.925 | 10.248 | 0.000 | 0.000 |
| medium14.txt | 350 | 900 | 345.000 | 408.515 | opt | 22.830 | 19.632 | 0.000 | 0.000 |
| medium15.txt | 350 | 900 | 382.000 | 120.088 | opt | 17.156 | 12.863 | 0.000 | 0.000 |
| medium16.txt | 300 | 700 | 270.000 | 142.905 | opt | 15.258 | 13.249 | 0.000 | 0.000 |
| medium17.txt | 300 | 700 | 168.000 | 143.992 | opt | 14.595 | 12.600 | 0.000 | 0.000 |
| medium18.txt | 300 | 700 | 134.000 | 9.182 | opt | 14.462 | 12.585 | 0.000 | 0.000 |
| medium19.txt | 300 | 700 | 121.250 | 7.638 | opt | 15.130 | 13.153 | 0.000 | 0.000 |
| medium20.txt | 300 | 700 | 118.700 | 11.352 | opt | 14.363 | 12.253 | 0.000 | 0.000 |

**Tabela 7:** *Rezultati metode promenljivih okolina na instancama velikih dimenzija*

| Instanca | $N$ | $M$ | CPLEX | | VNS | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $t_{best}(s)$ | $agap(\%)$ | $agap_\sigma$ |
| large1.txt | 500 | 1000 | 359.000 | 6012.964 | opt | 94.530 | 87.781 | 0.000 | 0.000 |
| large2.txt | 600 | 1400 | 465.000 | 9952.312 | opt | 226.452 | 215.104 | 0.086 | 0.105 |
| large3.txt | 800 | 2000 | 625.000* | - | 625.000 | 454.991 | 433.618 | 0.192 | 0.120 |
| large4.txt | 900 | 2300 | - | - | 704.000 | 639.130 | 608.615 | 0.114 | 0.057 |
| large5.txt | 1000 | 2500 | - | - | 771.000 | 700.594 | 665.905 | 0.130 | 0.116 |
| large6.txt | 1000 | 2500 | - | - | 787.000 | 542.581 | 510.255 | 0.280 | 0.203 |
| large7.txt | 1000 | 2500 | - | - | 806.000 | 563.140 | 527.070 | 0.149 | 0.093 |
| large8.txt | 1000 | 2500 | - | - | 824.000 | 501.352 | 465.740 | 0.073 | 0.059 |
| large9.txt | 900 | 2300 | - | - | 236.500 | 623.553 | 597.078 | 0.761 | 0.493 |
| large10.txt | 900 | 2300 | - | - | 278.000 | 650.314 | 622.114 | 0.576 | 0.288 |
| large11.txt | 900 | 2300 | - | - | 494.000 | 537.452 | 506.680 | 0.324 | 0.243 |
| large12.txt | 700 | 1600 | 501.000* | - | **500.000** | 281.666 | 269.374 | 0.280 | 0.204 |
| large13.txt | 700 | 1600 | 704.000* | - | **506.000** | 249.410 | 237.473 | 0.119 | 0.097 |
| large14.txt | 700 | 1600 | 513.000* | - | 513.000 | 337.730 | 325.550 | 0.039 | 0.078 |
| large15.txt | 700 | 1600 | 518.000* | - | 518.000 | 237.257 | 220.979 | 0.039 | 0.077 |
| large16.txt | 800 | 2000 | 627.000* | - | **624.000** | 407.723 | 386.494 | 0.096 | 0.128 |
| large17.txt | 800 | 2000 | 951.000* | - | **768.000** | 370.840 | 345.131 | 0.052 | 0.104 |
| large18.txt | 800 | 2000 | 961.000* | - | **915.000** | 403.681 | 370.718 | 0.087 | 0.082 |
| large19.txt | 800 | 2000 | 1051.000* | - | **1049.000** | 478.239 | 437.208 | 0.095 | 0.060 |
| large20.txt | 800 | 2000 | 1329.000* | - | **1328.000** | 434.526 | 380.457 | 0.286 | 0.324 |

**Tabela 8:** *Rezultati genetskog algoritma na instancama malih dimenzija*

| Instanca | $N$ | $M$ | CPLEX | | GA | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $t_{best}(s)$ | $gen_{avg}$ | $agap(\%)$ | $agap_{\sigma}$ |
| small1.txt | 20 | 50 | 31.000 | 0.044 | opt | 1.4 | 0.242 | 10.1 | 0 | 0 |
| small2.txt | 40 | 100 | 50.000 | 0.071 | opt | 4.78 | 0.686 | 21.2 | 0 | 0 |
| small3.txt | 60 | 150 | 65.000 | 0.390 | opt | 8.39 | 2.74 | 27 | 1.23 | 3.69 |
| small4.txt | 80 | 180 | 89.000 | 2.887 | opt | 12.7 | 4.26 | 27.3 | 0.112 | 0.337 |
| small5.txt | 100 | 250 | 99.000 | 1.933 | opt | 17.1 | 7.14 | 31.2 | 0.606 | 0.67 |
| small6.txt | 20 | 50 | 29.000 | 0.053 | opt | 2.13 | 0.252 | 20.6 | 0 | 0 |
| small7.txt | 20 | 50 | 38.000 | 0.109 | opt | 2.25 | 0.414 | 22.2 | 0 | 0 |
| small8.txt | 20 | 50 | 47.000 | 0.112 | opt | 2.18 | 0.338 | 21.4 | 0.213 | 0.638 |
| small9.txt | 20 | 50 | 56.000 | 0.141 | opt | 0.587 | 0.13 | 23.8 | 0.536 | 0.818 |
| small10.txt | 20 | 50 | 20.000 | 0.038 | opt | 0.505 | 0.0512 | 20.6 | 0 | 0 |
| small11.txt | 40 | 100 | 49.000 | 0.071 | opt | 0.981 | 0.137 | 21.3 | 0 | 0 |
| small12.txt | 40 | 100 | 51.000 | 0.095 | opt | 1.13 | 0.326 | 26.2 | 0 | 0 |
| small13.txt | 40 | 100 | 47.000 | 0.088 | opt | 0.999 | 0.163 | 21.9 | 0 | 0 |
| small14.txt | 40 | 100 | 41.000 | 2.358 | opt | 1.17 | 0.338 | 26 | 0 | 0 |
| small15.txt | 40 | 100 | 38.000 | 6.263 | opt | 1.39 | 0.532 | 30.1 | 0.263 | 0.789 |
| small16.txt | 60 | 150 | 68.000 | 1.585 | opt | 1.85 | 0.631 | 28.2 | 0.588 | 0.72 |
| small17.txt | 60 | 150 | 44.000 | 0.421 | opt | 1.69 | 0.498 | 26.1 | 0.909 | 1.11 |
| small18.txt | 60 | 150 | 36.000 | 0.247 | opt | 1.76 | 0.599 | 28.1 | 0.556 | 1.11 |
| small19.txt | 60 | 150 | 33.333 | 0.278 | opt | 1.79 | 0.641 | 28.9 | 0.3 | 0.9 |
| small20.txt | 60 | 150 | 32.400 | 0.194 | opt | 1.76 | 0.598 | 28.1 | 0.617 | 1.23 |

**Tabela 9:** *Rezultati genetskog algoritma na instancama srednjih dimenzija*

| Instanca | $N$ | $M$ | CPLEX | | GA | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $t_{best}(s)$ | $gen_{avg}$ | $agap(\%)$ | $agap_{\sigma}$ |
| medium1.txt | 150 | 400 | 154.000 | 24.322 | opt | 24.702 | 10.050 | 164.500 | 0.195 | 0.298 |
| medium2.txt | 200 | 500 | 192.000 | 39.584 | opt | 33.281 | 14.464 | 173.500 | 0.573 | 0.491 |
| medium3.txt | 250 | 600 | 236.000 | 169.704 | opt | 45.175 | 22.212 | 192.500 | 0.297 | 0.271 |
| medium4.txt | 300 | 700 | 270.000 | 174.961 | 271.000 | 74.833 | 47.172 | 262.700 | 0.480 | 0.406 |
| medium5.txt | 350 | 900 | 328.000 | 243.509 | opt | 81.509 | 47.700 | 236.800 | 0.183 | 0.149 |
| medium6.txt | 250 | 600 | 143.000 | 457.158 | opt | 48.444 | 24.320 | 198.000 | 0.629 | 0.660 |
| medium7.txt | 250 | 600 | 233.000 | 74.857 | opt | 51.279 | 27.815 | 214.100 | 0.644 | 0.346 |
| medium8.txt | 250 | 600 | 359.000 | 147.344 | 360.000 | 45.805 | 22.013 | 188.700 | 0.472 | 0.374 |
| medium9.txt | 250 | 600 | 485.000 | 62.819 | 486.000 | 65.757 | 42.048 | 273.900 | 0.453 | 0.288 |
| medium10.txt | 250 | 600 | 611.000 | 91.045 | 613.000 | 71.926 | 47.964 | 294.500 | 0.522 | 0.290 |
| medium11.txt | 150 | 400 | 150.000 | 37.116 | opt | 23.923 | 9.020 | 157.800 | 0.933 | 0.611 |
| medium12.txt | 150 | 400 | 143.000 | 74.970 | opt | 26.877 | 12.199 | 175.100 | 0.559 | 0.685 |
| medium13.txt | 150 | 400 | 143.000 | 114.607 | opt | 25.592 | 10.509 | 164.800 | 0.839 | 0.523 |
| medium14.txt | 350 | 900 | 345.000 | 408.515 | opt | 94.244 | 59.137 | 258.000 | 0.432 | 0.503 |
| medium15.txt | 350 | 900 | 382.000 | 120.088 | opt | 70.653 | 35.734 | 197.900 | 0.445 | 0.236 |
| medium16.txt | 300 | 700 | 270.000 | 142.905 | opt | 66.424 | 37.290 | 222.100 | 0.296 | 0.277 |
| medium17.txt | 300 | 700 | 168.000 | 143.992 | opt | 69.023 | 38.565 | 221.900 | 0.655 | 0.494 |
| medium18.txt | 300 | 700 | 134.000 | 9.182 | opt | 60.087 | 29.925 | 195.900 | 1.343 | 0.870 |
| medium19.txt | 300 | 700 | 121.250 | 7.638 | opt | 61.572 | 31.370 | 197.900 | 0.825 | 0.738 |
| medium20.txt | 300 | 700 | 118.700 | 11.352 | opt | 70.318 | 39.171 | 226.200 | 0.758 | 0.454 |

**Tabela 10:** *Rezultati genetskog algoritma na instancama velikih dimenzija*

| Instanca | $N$ | $M$ | CPLEX | | GA | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $t_{best}(s)$ | $gen_{avg}$ | $agap(\%)$ | $agap_{\sigma}$ |
| large1.txt | 500 | 1000 | 359.000 | 6012.964 | 360.000 | 382.326 | 263.695 | 788.300 | 0.611 | 0.461 |
| large2.txt | 600 | 1400 | 465.000 | 9952.312 | 469.000 | 510.028 | 346.417 | 763.000 | 0.832 | 0.375 |
| large3.txt | 800 | 2000 | 625.000* | - | 632.000 | 892.636 | 655.803 | 935.600 | 0.538 | 0.348 |
| large4.txt | 900 | 2300 | - | - | 712.000 | 1138.423 | 861.208 | 1013.500 | 0.801 | 0.491 |
| large5.txt | 1000 | 2500 | - | - | 782.000 | 1550.538 | 1241.743 | 1242.000 | 0.435 | 0.304 |
| large6.txt | 1000 | 2500 | - | - | 799.000 | 1175.852 | 879.573 | 978.300 | 0.526 | 0.284 |
| large7.txt | 1000 | 2500 | - | - | 812.000 | 1417.300 | 1116.767 | 1165.400 | 0.936 | 0.474 |
| large8.txt | 1000 | 2500 | - | - | 831.000 | 1187.710 | 890.574 | 985.800 | 0.710 | 0.435 |
| large9.txt | 900 | 2300 | - | - | 245.500 | 974.505 | 701.623 | 883.800 | 2.933 | 1.489 |
| large10.txt | 900 | 2300 | - | - | 290.000 | 1017.717 | 749.269 | 934.600 | 1.069 | 0.823 |
| large11.txt | 900 | 2300 | - | - | 505.000 | 1067.584 | 802.770 | 998.300 | 0.871 | 0.640 |
| large12.txt | 700 | 1600 | 501.000* | - | 509.000 | 898.144 | 687.588 | 1083.800 | 0.511 | 0.423 |
| large13.txt | 700 | 1600 | 704.000* | - | **511.000** | 691.615 | 482.882 | 814.400 | 0.626 | 0.624 |
| large14.txt | 700 | 1600 | 513.000* | - | 522.000 | 708.980 | 500.936 | 843.600 | 0.153 | 0.307 |
| large15.txt | 700 | 1600 | 518.000* | - | 524.000 | 718.174 | 512.804 | 876.600 | 0.382 | 0.209 |
| large16.txt | 800 | 2000 | 627.000* | - | 629.000 | 998.311 | 749.889 | 999.200 | 0.636 | 0.389 |
| large17.txt | 800 | 2000 | 951.000* | - | **778.000** | 948.458 | 709.640 | 965.800 | 0.463 | 0.450 |
| large18.txt | 800 | 2000 | 961.000* | - | **928.000** | 949.716 | 707.929 | 950.600 | 0.129 | 0.081 |
| large19.txt | 800 | 2000 | 1051.000* | - | 1064.000 | 1052.618 | 804.030 | 1050.200 | 0.244 | 0.401 |
| large20.txt | 800 | 2000 | 1329.000* | - | 1346.000 | 1218.922 | 988.261 | 1284.200 | 0.282 | 0.202 |

**Tabela 11:** *Rezultati hibridne metaheuristike na instancama velikih dimenzija*

| Instanca | $N$ | $M$ | CPLEX | | GA-VNS | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $gen_{avg}$ | $agap(\%)$ | $agap_{\sigma}$ |
| large1.txt | 500 | 1000 | 359.000 | 6012.964 | opt | 310.030 | 164.600 | 0.000 | 0.000 |
| large2.txt | 600 | 1400 | 465.000 | 9952.312 | opt | 670.423 | 217.800 | 0.043 | 0.086 |
| large3.txt | 800 | 2000 | 625.000* | - | 625.000 | 1222.939 | 259.400 | 0.224 | 0.128 |
| large4.txt | 900 | 2300 | - | - | 704.000 | 1599.697 | 271.200 | 0.114 | 0.057 |
| large5.txt | 1000 | 2500 | - | - | 771.000 | 2020.885 | 311.000 | 0.104 | 0.052 |
| large6.txt | 1000 | 2500 | - | - | 788.000 | 2037.459 | 366.000 | 0.102 | 0.095 |
| large7.txt | 1000 | 2500 | - | - | 805.000 | 1660.845 | 273.800 | 0.124 | 0.192 |
| large8.txt | 1000 | 2500 | - | - | 824.000 | 1593.227 | 280.600 | 0.024 | 0.049 |
| large9.txt | 900 | 2300 | - | - | 237.500 | 1478.879 | 253.000 | 0.253 | 0.206 |
| large10.txt | 900 | 2300 | - | - | 277.000 | 1826.134 | 280.600 | 0.433 | 0.270 |
| large11.txt | 900 | 2300 | - | - | 493.000 | 1676.221 | 255.200 | 0.243 | 0.298 |
| large12.txt | 700 | 1600 | 501.000* | - | **500.000** | 1020.031 | 252.200 | 0.080 | 0.098 |
| large13.txt | 700 | 1600 | 704.000* | - | **506.000** | 821.802 | 241.600 | 0.079 | 0.097 |
| large14.txt | 700 | 1600 | 513.000* | - | **512.000** | 845.555 | 244.600 | 0.117 | 0.096 |
| large15.txt | 700 | 1600 | 518.000* | - | 518.000 | 730.099 | 247.000 | 0.000 | 0.000 |
| large16.txt | 800 | 2000 | 627.000* | - | **624.000** | 1142.016 | 279.200 | 0.000 | 0.000 |
| large17.txt | 800 | 2000 | 951.000* | - | **767.000** | 1389.184 | 362.200 | 0.078 | 0.104 |
| large18.txt | 800 | 2000 | 961.000* | - | **914.000** | 1488.821 | 368.400 | 0.131 | 0.082 |
| large19.txt | 800 | 2000 | 1051.000* | - | **1049.000** | 1640.994 | 406.000 | 0.095 | 0.085 |
| large20.txt | 800 | 2000 | 1329.000* | - | **1328.000** | 2062.596 | 461.600 | 0.045 | 0.060 |

**Tabela 12:** *Rezultati metode promenljivih okolina na drugoj grupi instanci*

| Instanca | $N$ | $M$ | CPLEX | | VNS | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $t_{best}(s)$ | $agap(\%)$ | $agap_\sigma$ |
| random1.txt | 40 | 100 | 25.400 | 0.398 | opt | 2.162 | 0.470 | 0.000 | 0.000 |
| random2.tx | 40 | 120 | 37.000 | 5.505 | opt | 5.207 | 1.863 | 0.000 | 0.000 |
| random3.txt | 100 | 250 | 110.600 | 9.519 | 111.000 | 29.039 | 6.055 | 0.216 | 0.177 |
| random4.txt | 100 | 250 | 96.400 | 0.634 | opt | 6.940 | 1.613 | 0.000 | 0.000 |
| random5.txt | 250 | 600 | 190.400 | 559.960 | 190.600 | 33.377 | 6.669 | 0.000 | 0.000 |
| random6.txt | 350 | 900 | 345.300 | 133.431 | 345.600 | 116.535 | 57.003 | 0.081 | 0.069 |
| random7.txt | 350 | 900 | 237.200* | - | 237.200 | 236.779 | 73.435 | 0.118 | 0.126 |
| random8.txt | 500 | 1200 | 314.100* | - | 314.700 | 485.559 | 270.217 | 0.680 | 0.770 |
| random9.txt | 800 | 2000 | 431.500 | 10920.205 | 432.300 | 959.904 | 416.527 | 0.083 | 0.054 |
| random10.txt | 800 | 2000 | 343.000* | - | 343.400 | 1474.888 | 712.264 | 0.012 | 0.023 |

**Tabela 13:** *Rezultati genetskog algoritma na drugoj grupi instanci*

| Instanca | $N$ | $M$ | CPLEX | | GA | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $t_{best}(s)$ | $gen_{avg}$ | $agap(\%)$ | $agap_\sigma$ |
| random1.txt | 40 | 100 | 25.400 | 0.398 | opt | 7.767 | 0.881 | 222.900 | 0.157 | 0.315 |
| random2.tx | 40 | 120 | 37.000 | 5.505 | opt | 11.403 | 3.378 | 280.700 | 0.703 | 0.422 |
| random3.txt | 100 | 250 | 110.600 | 9.519 | opt | 41.018 | 22.454 | 439.200 | 0.687 | 1.120 |
| random4.txt | 100 | 250 | 96.400 | 0.634 | opt | 36.631 | 18.771 | 408.100 | 1.535 | 0.947 |
| random5.txt | 250 | 600 | 190.400 | 559.960 | 191.000 | 123.843 | 79.026 | 549.800 | 1.199 | 1.587 |
| random6.txt | 350 | 900 | 345.300 | 133.431 | 350.000 | 416.259 | 348.577 | 1230.200 | 0.566 | 0.630 |
| random7.txt | 350 | 900 | 237.200* | - | 239.100 | 360.614 | 293.944 | 1078.800 | 2.083 | 2.323 |
| random8.txt | 500 | 1200 | 314.100* | - | 337.100 | 666.859 | 569.209 | 1353.800 | 1.115 | 0.710 |
| random9.txt | 800 | 2000 | 431.500 | 10920.205 | 453.900 | 1478.799 | 1339.013 | 1675.400 | 3.886 | 2.621 |
| random10.txt | 800 | 2000 | 343.000* | - | 361.800 | 1447.554 | 1303.507 | 1680.000 | 2.742 | 1.519 |

**Tabela 14:** *Rezultati hibridne metaheuristike na drugoj grupi instanci*

| Instanca | $N$ | $M$ | CPLEX | | GA-VNS | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | sol | t(s) | best | $t_{tot}(s)$ | $gen_{avg}$ | $agap(\%)$ | $agap_\sigma$ |
| random1.txt | 40 | 100 | 25.400 | 0.398 | opt | 9.132 | 80.700 | 0.000 | 0.000 |
| random2.tx | 40 | 120 | 37.000 | 5.505 | opt | 20.102 | 91.400 | 0.000 | 0.000 |
| random3.txt | 100 | 250 | 110.600 | 9.519 | opt | 77.105 | 179.100 | 0.145 | 0.177 |
| random4.txt | 100 | 250 | 96.400 | 0.634 | opt | 35.093 | 149.700 | 0.000 | 0.000 |
| random5.txt | 250 | 600 | 190.400 | 559.960 | 190.600 | 205.084 | 322.100 | 0.000 | 0.000 |
| random6.txt | 350 | 900 | 345.300 | 133.431 | 345.500 | 560.709 | 460.000 | 0.081 | 0.085 |
| random7.txt | 350 | 900 | 237.200* | - | 237.200 | 628.581 | 583.600 | 0.067 | 0.068 |
| random8.txt | 500 | 1200 | 314.100* | - | 314.300 | 1100.054 | 667.800 | 0.458 | 0.567 |
| random9.txt | 800 | 2000 | 431.500 | 10920.205 | 432.200 | 2405.870 | 917.200 | 0.130 | 0.094 |
| random10.txt | 800 | 2000 | 343.000* | - | 343.000 | 2763.478 | 889.000 | 0.082 | 0.079 |