UNIVERZITET U BEOGRADU
MATEMATIČKI FAKULTET

Jovana Tomić

HEURISTIČKI PRISTUP REŠAVANJU LOKACIJSKOG PROBLEMA OGRANIČENIH KAPACITETA SA NADMETANJEM SA DELIMIČNIM ZADOVOLJENJEM POTRAŽNJE

master rad

Beograd, 2021.

Mentor:

prof. dr Zorica Stanimirović, redovni profesor
Univerzitet u Beogradu, Matematički fakultet

Članovi komisije:

prof. dr Aleksandar Savić, vanredni profesor
Univerzitet u Beogradu, Matematički fakultet

doc. dr Zorica Dražić, docent
Univerzitet u Beogradu, Matematički fakultet

Datum odbrane: _____

Naslov master rada: Heuristički pristup rešavanju lokacijskog problema ograničenih kapaciteta sa nadmetanjem sa delimičnim zadovoljenjem potražnje

Rezime: Lokacijski problem prikazan u radu je problem ograničenih kapaciteta sa nadmetanjem, sa delimičnim zadovoljenjem potražnje korisnika. Lokacijski problemi sa nadmetanjem spadaju u klasu diskretne optimizacije, koji imaju povezanost sa teorijom igara s obzirom na to da se rivali (lider i sledbenik) nadmeću na tržištu. Sastoje se u određivanju optimalnih lokacija za uspostavljanje objekata kompanije koja pruža određenu uslugu korisnicima unutar posmatrane oblasti u kojoj već postoji određen broj uspostavljenih objekata konkurentske kompanije, sa ciljem maksimizacije profita oba igrača. Problem koji se razmatra spada u klasu NP-teških problema. Za navedeni problem izložena su dva modela koja su preuzeta iz literature: u prvom modelu igrači moraju da zadovolje celokupnu potražnju korisnika, dok je u drugom modelu dozvoljeno da drugi rival zadovolji samo deo potražnje korisnika. Predložene su tri heurističke metode za rešavanje navedenog problema: genetski algoritam, osnovna metoda promenljivih okolina i memetski algoritam koji predstavlja hibridizaciju genetskog algoritma i osnovne metode promenljivih okolina. Sve implementirane metode međusobno su upoređene testiranjem na instancama male, srednje i velike dimenzije. Za rešavanje optimizacionog problema lidera korišćene su navedene heuristike, dok je za računanje funkcije cilja sledbenika korišćen egzaktni rešavač CPLEX.

Ključne reči: lokacijski problem ograničenih kapaciteta sa nadmetanjem, genetski algoritam, osnovna metoda promenljivih okolina, memetski algoritam, teorija igara, delimično zadovoljenje potražnje.

Sadržaj

Glava 1

Uvod

1.1 Lokacijski problemi

Lokacijski problemi čine značajnu klasu zadataka optimizacije koji u poslednje vreme izazivaju značajno interesovanje u savremenim operacionim istraživanjima. Teorija lokacije bavi se formulacijom i rešavanjem lokacijskih problema. Zadatak teorije lokacijskih problema predstavlja izbor jedne ili više lokacija za objekte koji pružaju određene usluge u prostoru određene dimenzionalnosti, tako da potrebe korisnika budu zadovoljene uz minimalne troškove ili maksimalan profit. Lokacija predstavlja mesto odnosno položaj gde je nešto locirano, određeno geografsko područje ili deo prostora izabran za obavljanje određene delatnosti. Objekti obično predstavljaju neku vrstu centara koji pružaju određene usluge, koji se nečesto nazivaju snabdevači ili resursi, a čije usluge koriste određeni klijenti odnosno korisnici (potrošači) tih usluga. Pri rešavanju lokacijskih problema mogu se odrediti lokacije objekata, ali i njihov broj kao i kapacitet objekata. Korisnici, odnosno njihov raspored u prostoru bitna su karakteristika lokacijskih problema, čije rešavanje vrlo često podrazumeva izbor lokacija uz istovremeno alociranje (dodeljivanje, pridruživanje) korisnika tim lokacijama. Posebna pažnja pridaje se interakcijama između samih lokacija i prostorno raspoređenih korisnika. Interakcije prema svojoj prirodi mogu biti raznovrsne i direktno zavise od prirode objekata koji se lociraju. Na primer interakcije mogu predstavljati transport robe, transport ljudi ili fizičke veze kao što su saobraćajnice, cevovodi, kablovi, komunikacije i slično. Teorija lokacije najčešće se bavi problemima lociranja tačke u dvodimenzionalnom prostoru. Potreba za rešavanjem lokacijskih problema prouzrokovala je ubrzan proces izučavanja ovog tipa problema, i kao takvi oni imaju široku primenu u industriji, trgovini transportu i slično

[31]. Na primer, prilikom određivanja lokacija supermarketa u okviru jednog trgovinskog lanca, određivanja lokacija zdravstvenih ustanova, bolnica, centara mobilne telefonije, centara kurirske službe, pozicioniranja autobuskih i železničkih stanica, određivanja lokacija škola, policijskih stanica itd.

1.2 Istorijat lokacijskih problema

Prvi lokacijski problem koji se pojavio u istoriji matematike, poznat u literaturi i kao Fermaov zadatak [56], glasi: Ako su date tri tačke u ravni, naći četvrtu tačku takvu da je suma njenih rastojanja do preostale tri tačke minimalna. Pjer de Ferma[1] se najčešće navodi u literaturi kao istraživač koji je prvi postavio lokacijski problem. Dakle, Ferma je postavio lokacijski problem u kojem je potrebno naći tačku trougla čiji je zbir rastojanja do temena minimalan. Toričeli [2] je prvi matematičar koji je rešio ovaj problem konstruktivnom metodom. Problem je rešio tako što je konstruisao jednakostranične trouglove nad svakom stranicom sa spoljašnje strane trougla. U preseku krugova opisanih oko novonastalih trouglova nalazi se tražena tačka koja je kasnije nazvana i Toričelijeva tačka. Prema drugim izvorima smatra se da je Bonaventura Cavalieri [3] prvi formulisao lokacijski problem, a kasnije ga i rešio konstruktivnim putem. Skoro ceo vek kasnije Tomas Simpson [4] predlaže još jedno konstruktivno rešenje. Simspon kasnije uvodi uopštenje ovog problema: svakoj tački trougla dodeliti težinu, a potom naći tačku takvu da je suma težinskih rastojanja od temena trougla do te tačke minimalna. Razvoju lokacijskih problema doprineli su i Vivijani [5], Roberval [6] i drugi. Dva veka nakon postavke Fermaovog zadatka,

[1] Pjer de Ferma (franc. Pierre de Fermat, 1601 - 1665) bio je jedan od najznačajnijih francuskih matematičara, kao i pravnik u tuluskom parlamentu. Jedan je od začetnika diferencijalnog računa sa svojom metodom pronalaženja najvećih i najmanjih ordinata krivih linija, analognim tada još nepoznatom diferencijalnom računu. Veliki doprinos dao je izučavanju teorije brojeva, analitičkoj geometriji i verovatnoći. Poznat je po Velikoj Fermaovoj teoremi, koja je po njemu i dobila ime.

[2] Evanđelista Toričeli (it. Evangelista Torricelli, 1608 - 1647) bio je italijanski fizičar i matematičar, najpoznatiji po svom izumu barometra. Poznat je još i po Toričelijevom zakonu, Toričelijevoj trubi, a dao je doprinos i dokazu formule za sumu geometrijskog niza. Asteroid 7437 Toričeli je nazvan u njegovu čast.

[3] Bonaventura Kavalijeri, (it. Bonaventura Cavalieri, 1598 - 1647) bio je italijanski matematičar koji se bavio optikom, kretanjem, logaritmima, beskonačno malim veličinama i integralnim računom.

[4] Tomas Simpson, (eng. Thomas Simpson 1710 - 1761) bio je engleski matematičar i pronalazač. Poznat je po Simpsonovom pravilu o približavanju definitivnih integrala.

[5] Vićenco Vivijani (it. Vincenzo Viviani, 1622 - 1703) bio je italijanski naučnik koji se bavio fizikom i geometrijom. Bio je Toričelijev učenik i Galilejev pomoćnik.

[6] Žil de Roberval (franc. Gilles de Roberval, 1602 - 1675) bio je francuski matematičar i astronom koji je podržao Kopernikov heliocentrični sistem.

1909. godine engleski ekonomista Veber [7] je našao praktiču primenu ovog modela u industriji i izneo je u radu [1]. Veber je preformulisao Fermaov problem tako da se može primeniti na praktični problem minimizacije troškova transporta, koji se često javlja u industriji. Naime, dva od tri temena trougla Veber označava kao resurse sirovina, a treće teme trougla označava kao lokaciju potrošača. Svakom temenu je pridružio određenu težinu. Zadatak predstavlja traženje optimalne tačke odnosno lokacije za uspostavljanje industrijskog postrojenja sa ciljem da cena transporta bude minimalna. Veber je rešio problem konstruktivnim putem i predložio isti postupak za dimenzije veće od 3. Zanimljivo je da je uopštena težinska verzija problema sa m fiksiranih tačaka kasnije vezana za Veberovo ime, iako je on nije ni prvi put formulisao niti ju je prvi rešio. Veberov problem tj. uopštena težinska verzija navedenog problema sa m fiksiranih tačaka glasi: Neka su $(X_i, Y_i), i = 1, ..., n$ koordinate tačaka (lokacije korisnika) i W_i, $i = 1, ..., n$, odgovarajuće težine (cena transporta do datog korisnika po jedinici rastojanja). Odrediti tačku (X^*, Y^*) takvu da suma težinskih rastojanja do ostalih tačaka bude minimalna (naći optimalnu lokaciju skladišta tako da se minimizuju ukupni troškovi). Veberov problem nalaženja optimane lokacije skladišta se matematički formuliše na sledeći način:

$$min \quad W(x, y) = \sum_{i=1}^{n} w_i d_i(x, y) \tag{1.1}$$

gde $d_i(x, y) = \sqrt{(x - x_i)^2 + (y - y_i)^2}$ predstavlja euklidsko rastojanje tačaka (x, y) i (x_i, y_i) za $i = 1, 2, ..., n$.

Kako se Veberov problem proučava i danas, razvijene su brojne metode i modeli za rešavanje raznih njegovih modifikacija, dopuna i uopštenja. Od sredine šezdesetih godina 20. veka teorija lokacijskih problema počinje sve više da se razvija, matematički se formulišu različiti tipovi lokacijskih problema i uvode se različiti algoritmi za njihovo rešavanje. U to vreme rad na lokacijskim problemima sastojao se prvenstveno od istraživanja raznih pojedinačnih problema i njihovih primena. Vremenom počinju da se istražuju primene u određivanju lokacija vatrogasnih centara, deponija, fabričkih postrojenja, u telefonskoj industriji, železničkoj industriji i slično. Navedene primene mogu se naći u radovima [53], [44], [41], [42] i [20].

[7] Alfred Veber (nem. Alfred Weber, 1868 - 1958) bio je nemački ekonomista, geograf, sociolog i teoretičar kulture čiji je rad uticao na razvoj moderne ekonomske geografije.

1.3 Klasifikacija lokacijskih problema

Imajući u vidu veliku oblast primene lokacijski problemi su tokom poslednjih decenija izučavani i sa teorijskog i sa praktičnog aspekta. S obzirom na širok spektar tipova ovih problema, vremenom se javila potreba za njihovom sistematizacijom. Uprkos brojnim pokušajima i dalje ne postoji tačna podela, takva da obuhvati sve varijante ovih problema. Naime, lokacijski problemi se mogu klasifikovati prema raznim kriterijumima, a neki od njih su sledeći:

- **Topografija prostora**
  Jedna od najbitnijih podela lokacijskih problema jeste podela po vrsti prostora koji se pretražuje. Prostor pretrage može biti konačan i beskonačan. Na osnovu prostora pretrage lokacijski problemi se dele na *diskretne, kontinualne* i *mrežne probleme*. Ukoliko dopustivi skup pretrage ima beskonačno mnogo tačaka, onda se govori o kontinualnim lokacijskim problemima, što znači da se objekti mogu locirati bilo gde u dopustivom prostoru ($\mathbb{R}^2, \mathbb{R}^3, \ldots$). Ukoliko dopustivi skup pretrage predstavlja konačan skup lokacija gde se mogu locirati novi objekti onda se govori o diskretnim lokacijskim problemima. U mrežnim lokacijskim problemima objekti se mogu locirati na bilo kom čvoru zadate mreže.

- **Broj objekata**
  Lokacijski problemi se mogu klasifikovati prema tome da li je broj objekata koje treba uspostaviti unapred zadat ili ne. *Endogeni lokacijski problemi* su oni problemi kod kojih je broj objekata koje treba uspostaviti unapred poznat, dok su *egzogeni lokacijski problemi* oni problemi kod kojih broj objekata koje treba uspostaviti nije unapred zadat, već i taj broj predstavlja rezultat optimizacije problema. Takođe, endogeni problemi se dalje mogu klasifikovati prema broju objekata koje je potrebno uspostaviti. Uglavnom se razlikuju modeli kod kojih je potrebno uspostaviti jedan objekat (*eng. single facility*) i modeli kod kojih je potrebno uspostaviti više od jednog objekta (*eng. multi-facility*).

- **Oblik funkcije cilja**
  Prema obliku funkcije cilja lokacijski problemi se dele na *Min-Sum* i *Min-Max* modele. Kod *min-sum* modela funkcija cilja minimizuje ukupnu sumu rastojanja između objekata i korisnika, dok u *min-max* modelu funckija cilja minimizuje maksimalno rastojanje između objekta i klijenta.

- **Kapacitet objekata**
  Objekti koji se uspostavljaju na određenim lokacijama mogu imati *ograničen* ili *neograničen maksimalni kapacitet.* U literaturi postoji veliki broj lokacijskih problema sa neograničenim kapacitetima, međutim ograničenje kapaciteta objekata koji se uspostavljaju na određenim lokacijama umnogome bolje modeluje realne situacije. Na osnovu navedenog lokacijski problemi mogu se podeliti na *lokacijske probleme ograničenih kapaciteta* i *lokacijske probleme neograničenih kapaciteta.*

- **Način lokacije i alokacije**
  Lokacijski problemi se mogu podeliti na *lokacijske, alokacijske* i *lokacijsko-alokacijske probleme.* Preciznije mogu se locirati objekti na unapred određenim lokacijama, mogu se alocirati korisnici za već unapred zadate objekte na unapred određenim lokacijama, a mogu se ujedno i locirati objekti na unapred određenim lokacijama i alocirati klijenti za tako uspostavljene objekte.

- **Način pridruživanja**
  Ukoliko je neophodno locirati više uslužnih objekata naknadno je potrebno alocirati korisnike uspostavljenim uslužnim objektima. Korisnik se može pridružiti samo jednom uspostavljenom objektu, svakom od uspostavljenih objekata ili fiksiranom broju r uspostavljenih objekata, koji je manji ili jednak ukupnom broju uspostavljenih objekata. Na osnovu navedenog, lokacijski problemi se mogu podeliti na osnovu tipa alokacije, respektivno, na *probleme sa jednostrukom alokacijom (eng. single allocation problems), probleme sa višestrukom alokacijom (eng. multiple allocation problems)* i *probleme sa r alokacijom (eng. r – allocation problems).*

- **Karakteristike korisnika**
  Korisnici mogu imati posebne zahteve kada je u pitanju izbor uspostavljenih objekata. Zahtevi korisnika mogu biti unapred poznati i nepromenljivi, a mogu se menjati u zavisnosti od posebnih karakteristika objekata kao što su udaljenost, cena, snabdevenost, kvalitet i slično. Na osnovu navedenog, lokacijski problemi se prema zahtevima korisnika dele na *lokacijske probleme sa neelastičnim zahtevima korisnika* i *lokacijske probleme sa elastičnim zahtevima korisnika*, respektivno. Takođe, ponašanje korisnika pri odluci za koji objekat će se alocirati može biti unapred određeno u zavisnosti od nekog konkretnog faktora, na primer od udaljenosti ili cene, a može zavisiti i od više faktora. Na

osnovu navedenog, lokacijski problemi se prema ponašanju korisnika mogu podeliti na lokacijske probleme kod kojih je ponašanje korisnika *determinističko* i *probabilističko*, respektivno. U probalističkom slučaju uvodi se funkcija privlačnosti sa ciljem da se pomoću nje meri uticaj svih karakteristika uslužnih objekata na odluku korisnika.

- **Karakteristike objekta**
  U odnosu na poželjnost objekata od strane korisnika, lokacijski problemi se dele na *lokacijske probleme sa poželjnim (eng. pull objective), nepoželjnim (eng. push objective)* i *kombinovano poželjno-nepoželjnim (eng. pull-push objective)* karakteristikama objektima. Na primer, ako se lociraju marketi u nekom gradu, svi korisnici žele da se objekti lociraju što bliže njima, ali ako se na primer lociraju deponije, izabrane lokacije treba da budu što dalje od korisnika.

- **Način određivanja cene usluge**
  Pri određivanju cilja lokacijskog problema bitno je imati u vidu prirodu problema. Na primer, uglavnom se u privatnom sektoru dobit meri u novčanim jedinicama. Međutim, postoje slučajevi kada se cilj ne može predstaviti u novčanim jedinicama, na primer pri određivanju lokacija hitnih medicinskih i vatrogasnih službi, te se u tim slučajevima funkcija cilja ne može predstaviti samo cenom prevoza. Na osnovu navedenog, lokacijski problemi se žargonski rečeno mogu podeliti na *probleme privatnog i javnog sektora.* Takođe, postoje razni načini određivanja cene proizvoda. Na primer, kompanija koja je uspostavila objekat na određenoj lokaciji ima mogućnost da u cenu svog proizvoda ne uključi troškove prevoza *(eng. mill – price policy)* ili da u cenu svog proizvoda uključi troškove prevoza *(eng. delivered price policy).*

- **Tip nadmetanja**
  Kao što se može videti, postoji mnogo načina za klasifikaciju lokacijskih problema. Daljom razradom svaki tip lokacijskih problema se dalje može klasifikovati. Detaljnija klasifikacija se može naći u radu [28]. Kako ovaj rad govori o lokacijskom problemu sa nadmetanjem, ovde se napominje i način podele na osnovu tipa nadmetanja. Lokacijski problemi mogu biti *statički, dinamički* i *sekvencijalni.* Statički problemi podrazumevaju da su karakteristike kompanija koje se nadmeću (lokacija njenih objekata, strategija, cena proizvoda koje nude i slično) unapred poznate, odnosno poznate pre ulaska na tržište, kao i da su date karakteristike fiksirane i ne menjaju se pri uspostavljanju novih obje-

kata od strane konkurentske firme. Kod dinamičkih problema sve kompanije donose svoje odluke istovremeno, ali su moguće odgovarajuće reakcije konkurentskih kompanija na izmene situacije na tržištu. Kod sekvencijalnih problema postoji određena hijerarhija u uspostavljanju objekata, ali su takođe moguće odgovarajuće reakcije konkurentskih kompanija na izmene situacije na tržištu.

Glava 2

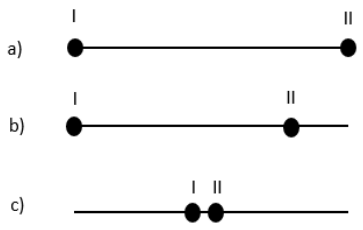
Lokacijski problemi sa nadmetanjem

Lokacijski problemi sa nadmetanjem *(eng. Compatitive Facility Location Problem, CFLP)* predstavljaju klasu problema teorije lokacije u kojima se određene kompanije takmiče na tržištu. U klasičnim lokacijskim problemima nalazi se jedna kompanija koja želi da maksimizuje svoj profit tako što će uspostaviti snabdevače na određenim lokacijama i svojim karakteristikama pridobiti korisnike. Naime, osnovna varijanta lokacijskih problema podrazumeva da se na tržištu nalazi jedna kompanija i da ona nema konkurenciju, što predstavlja monopol. Takođe, u osnovnoj varijanti lokacijskih problema kompanija koja se nalazi na tržištu jednostavno prihvata cenu koja preovladava na tržištu. Ono što lokacijske probleme sa nadmetanjem razlikuje od osnovne varijante ovih problema jeste uzimanje u obzir konkurencije i određenih efekata koje ta konkurencija nosi sa sobom. S obzirom da je uticaj konkurencije od ključnog značaja u realnim problemima, granaju se i klase lokacijskih problema sa nadmetanjem. Osnovnu varijantu ovog problema čini određen broj potrošača kojima je potrebna određena usluga, određen broj proizvođača koji nude datu uslugu i lokacije na kojima proizvođači mogu da uspostave svoje objekte. Zadatak navedenog problema jeste da proizvođači, imajući u vidu konkurenciju na tržištu, pronađu lokacije na kojima da uspostave svoje objekte kako bi maksimizovali svoj profit. Više o lokacijskim problemima sa nadmetanjem može se naći u radovima [24], [54], [52] i [19]. Lokacijski problem sa nadmetanjem među prvima je izučavao statističar Harold Hoteling[1] , odnosno prvi se bavio analizom uticaja nadmetanja na ponašanje konkurentskih kompanija na tržištu. Hoteling je 1929. godine u radu [13] izneo

[1] Harold Hoteling, (eng. Harold Hotelling, 1895 - 1973), američki statističar i ekonomista. Pored drugih doprinosa, uveo je Hotelingovo pravilo, Hotelingovu lemu i Hotelingov zakon u oblasti ekonomije i T-kvadrat raspodelu u oblasti statistike.

princip minimalne diferencijacije (*eng. principle of minimum differentiation*), koji je još poznat i kao Hotelingov zakon. Hotelingov zakon govori o tome da je kompanijama često najisplativije da svoju uslugu što više prilagode usluzi koju pružaju konkurentske kompanije. Na primer, čest je slučaj da trgovinski lanci, restorani, apoteke, benzinske pumpe imaju svoje objekte na jako maloj razdaljini od objekata svojih konkurenata. Zanimljiv primer predstavlja primena Hotelingovog zakona na političke partije koje tokom izbora pokušavaju da približe svoje političke programe drugim partijama i na taj način privuku što više glasača.

U nastavku je predstavljen linearni Hotelingov model. Pretpostavlja se da na tržištu vlada duopol, odnosno da postoje dve kompanije (igrači) koje žele da otvore svoju prodavnicu u jednoj ulici. Stoga, govori se o duopolu i o linearnom modelu. S obzirom da je usluga (cena, kvalitet, ...) obe kompanije ista, pretpostavlja se da će kupci birati kompaniju koja im je bliža. Cilj oba igrača je da maksimizuju svoj udeo na tržištu povećanjem tražnje, pri čemu se zanemaruju svi troškovi kao što su troškovi uspostavljanja, troškovi transporta i slično. Zadatak se svodi na to da se nađe ravnotežna tačka, koja podrazumeva da svaka od kompanija opslužuje po pola tržišta. Iako postoji više načina da se postigne navedeni cilj, problem se oslikava u tome što svaka kompanija teži da poveća svoj uticaj na tržištu. Ukoliko se dve firme pozicioniraju na krajevima ulice svaka kompanija će opslužiti po pola tržišta, što zadovoljava krajnji cilj problema i prikazan je na slici 2.1a. Međutim, postoji opasnost za jednu kompaniju da se druga približi i preuzme veći deo tržišta, što je predstavljeno na slici 2.1b. Kako bi kompanije izbegle ovaj rizik, obe se postepeno približavaju ka centru, i jedini način da svaka od njih opsluži polovinu tržiša, a da ne postoji rizik da se druga kompanija približi i time prvoj preotme kupce, je da se obe firme poziciniraju na centar ulice, jako blizu jedna drugoj, kao što je prikazano na slici 2.1c. Iako bi sa tačke gledišta kupca najviše odgovaralo da se prodavnice lociraju na četvrtini udaljenosti od krajeva ulice, kompanije se ipak odlučuju za ovo rešenje, jer ne žele da se udalje od sredine i na taj način dopuste rivalu da se proširi.



Slika 2.1: Slika 1

2.1 Veza teorije igara i lokacijskih problema sa nadmetanjem

Teorija igara predstavlja matematičku teoriju i metodologiju koja se koristi za analizu i rešavanje konflitknih situacija. Svaku igru definišu igrači, potezi igrača i isplata. Igrači mogu biti pojedinci, države, kompanije, bilo ko ko ima neki interes od igre. Igrači donose odluke od kojih zavisi dalji tok igre. Potez ili odluka igrača se drugačije zove i strategija igrača, dok isplata predstavlja doprinos igre igračima. Lokacijski problemi sa nadmetanjem predstavljaju tipičan problem optimizacije koji je usko povezan sa teorijom igara, gde kompanije predstavljaju igrače. Kada se na tržištu nalaze dve kompanije data igra predstavlja duopol. Simultane igre nemaju vremensku osu i odluke se saopštavaju simultano, tj. u isto vreme. U sekvencijalnim igrama igrači naizmenično razmenjuju poteze i na taj način se omogućava analiza strategija igrača u odnosu na vreme.

Kod sekvencijalnih igri kompanije na tržištu na kome vlada duopol se najčešće nazivaju vođa (lider) i sledbenik (pratilac), (*eng. Lider and Follower*). U zavisnosti od toga da li vođa i sledbenik donose odluku istovremeno ili jedan pa drugi, model će slediti Nešovu[2] ili Stackelbergovu [3] ravnotežnu tačku (ekvilibrijum).

[2]Džon Forbes Neš (eng. John Forbes Nash, 1928 - 2015) bio je američki matematičar i bavio se teorijom igara, diferencijalnom geometrijom i delom diferencijalnim jednačinama. Dobitnik je Nobelove nagrade za ekonomiju 1994. godine i Abelove nagrade 2015. godine. Po njemu je nazvan Nešov ekvilibrijum.

[3]Hajnrih Frajher fon Stakelberg (eng. Heinrich Freiherr von Stackelberg, 1905 - 1946) bio je nemački ekonomista koji je doprineo teoriji igara i industrijskoj organizaciji i poznat je po Stakelbergovom modelu liderstva.

U teoriji igara Nešov ekvilibrijum predstavlja profil strategija (po jedna za svakog igrača) definisanih tako da nijedan igrač, imajući u vidu izbor strategija ostalih igrača, neće postići bolji rezultat za sebe jednostrano. Dakle, Nešov ekvilibrijum čini onaj par strategija koji predstavlja uzajamno najbolji odgovor igrača na postavljenu igru. Neš je 1950. godine izneo teoriju da svaka igra koja ima konačni broj igrača sa konačnim brojem strategija (konačna igra) ima najmanje jednu ravnotežnu tačku, koja je kasnije i nazvana po njemu - Nešova ravnotežna tačka.

Igrači ne moraju uvek biti ravnopravni, već jedan može biti dominantan. Radi izbegavanja zabune, pod terminom „dominantnosti" ovde se ne smatra stroga dominacija koja je poznata u teoriji igara. Dominantnost je posledica veličine ili moći, brzine obrade podataka ili jednostavno nedostataka informacija kod jednog od igrača. Na primer, time što igrač 1 dopušta igraču 2 da prvi odabere svoju strategiju, igrač 2 dominira.

Neka igračima 1 i 2 stoji na raspolaganju izbor po jedne promenljive s_1 i s_2, a njihove funkcije efekta su $P(s_1, s_2)$ i $Q(s_1, s_2)$, respektivno. Oba igrača teže da minimizuju svoje P odnosno Q pri čemu je izbor promenljivih ograničen: $s_1 \in S_1$ i $s_2 \in S_2$, a pri tome prvo bira igrač 2, a potom igrač 1. Ako postoji preslikavanje T kojim se za svako $s_2 \in S_2$ određuje $s_1 = T(s_2)$, i to tako da je $P(T(s_2), s_2) \leq P(s_1, s_2)$ za svako $s_1 \in S_1$ i ako postoji $s_2^* \in S_2$ takvo da je $Q(T(s_2^*), s_2^*) \leq Q(T(s_2), s_2)$ tada par $(s_1^* = T(s_2^*), s_2^*)$ predstavlja optimalne strategije za oba igrača , kada je igrač 2 dominantan. Ove strategije nazivaju se optimalne u smislu Stakelberga, odnosno Stakelbergove ravnotežne tačke (ekvilibrijumi). Na sličan način definišu se i optimalne strategije (s_1^*, s_2^*) kada je igrač 1 dominantan. Ukoliko je u igri u kojoj nijedan igrač nije dominantan ravnotežna tačka ista kao i ravnotežna tačka (u istoj igri) u kojoj je neki igrač dominantan, to znači da dominantni igrač nema koristi od toga što je u dominantnom položaju.

2.2 Heurističke metode za rešavanje lokacijskih problema sa nadmetanjem

Problemi kod kojih se određeno rešenje X može verifikovati u polinomskom vremenu, tj. može se utvrditi da li je X rešenje problema, ali ne i utvrditi njegova optimalnost za polinomsko vreme, nazivaju se NP teški problemi. Većina problema kombinatorne optimizacije su NP-teški problemi. Postoji mnogo praktičnih problema velike dimenzije, na primer u industriji, za koje egzaktne metode ne mogu naći

optimalno rešenje usled nedovoljno memorijskog prostora ili nedopustivo velikog vremena izvršavanja algoritma. Rapidan tehnološki razvoj je doprineo primetnom poboljšanju moći i performansi računarskih sistema. Međutim, instance većine NP-teških problema velikih dimenzija i dalje je praktično nemoguće rešiti usled ograničenja vremenskih i memorijskih resursa. Budući da su lokacijski problemi uglavnom NP-teški, vreme izvršavanja algoritma koji ih rešavaju može postati nedopustivo dugo, što algoritam čini praktično neupotrebljivim.

Metode koje u prihvatljivom vremenu dolaze do dovoljno dobrih, neretko i optimalnih rešenja jesu heurističke metode [30]. One predstavljaju klasu aproksimativnih metoda za rešavanje optimizacionih problema. Pregled metoda optimizacije za rešavanje lokacijskih problema može se naći u radovima [47], [21] i [22]. Kod heurističkih metoda ne postoji sigurnost da će algoritam dostići optimalno rešenje, kao kod egzaktnih metoda, međutim kada se heuristika dobro osmisli i prilagodi problemu koji se rešava, ove metode dolaze do rešenja dovoljno bliskog optimalnom ili dostižu optimalno rešenje. Pri rešavanju praktičnih problema u realnim životnim situacijama često je bolje brzo naći dovoljno dobro rešenje koje je blisko optimalnom, (ili jednako optimalnom), nego utrošiti mnogo vremena da se sa sigurnošću dostigne optimalno rešenje. Kako bi se obezbedila visoka preciznost metoda se mora dobro prilagoditi konkretnom problemu koji se rešava. Kako bi se prevazišao ovaj problem, stvorene su metaheurstičke metode koje predstavljaju uopštenje heurističkih metoda za rešavanje više vrsta problema. Izučavanje navedenih metoda započeto je sredinom 20. veka i do sada je razvijeno mnoštvo njih, dok su prve metaheurstičke metode konstruisane tokom osamdesetih godina istog veka. Najviše metoda koje su razvijene inspirisane su prirodnim procesima poput procesa prenošenja genetskog algoritma, ponašanja roja pčela, ponašanje skupa mrava i slično. Za veliki broj problema optimizacije dokazano je da su NP-teški. Navedena činjenica povećava značaj heurističkih metoda i ostavlja prostor za dalji razvoj istih.

Osnovna podela metaheuristika je podela na S i P metaheuristike. S - metaheuristike su metode zasnovane na iterativnom poboljšanju jednog rešenja (*eng. single solution*), dok su P - metaheuristike zasnovane na poboljšanjima skupa rešenja - populacije (*eng. population based*). Najpoznatije S-heuristike su lokalna pretraga, iterativna lokalna pretraga, pohlepna stohastičko-adaptivna procedura pretrage, simulirano kaljenje, tabu pretraga i metoda promenljvih okolina. Lokalno pretraživanje (*eng. Local Search, LS*) predstavlja najstariju i najjednostavniju S-metaheuristiku. Ideja ove metode je doći do najboljeg rešenja sukcesivnim pretraži-

vanjem okolina rešenja koja se posećuju tokom procesa pretrage. Iterativna lokalna pretraga (*eng. Iterated local search, ILS*) predstavlja metaheuristiku zasnovanu na ponavljanju lokalne pretrage sa ciljem povećavanja kvaliteta uzastopnih dobijenih rešenja. Pohlepna stohastičko-adaptivna procedura pretrage (*eng. Greedy Randomized Adaptive Search Procedure, GRASP*) je metaheuristička metoda sa višestartnim pristupom zasnovana na lokalnom pretraživanju. Svaka iteracija se sastoji iz dve osnovne faze: nasumično-pohlepnog algoritma za konstrukciju inicijalnog rešenja i lokalne pretrage. Simulirano kaljenje (*eng. Simulated annealing, SA*) je takođe zasnovano na lokalnom pretraživanju koje dozvoljava posećivanje lošijih rešenja u cilju nalaženja globalnog optimuma. Inspirisana je procesom kaljenja čelika. Tabu pretraživanje (*eng. Tabu Search, TS*) je S - metaheuristika zasnovana na lokalnom pretraživanju koja koristi memoriju procesa pretrage u cilju izbegavanja zamke lokalnog optimuma. Metoda promenljivih okolina (*eng. Variable neighborhood Search, VNS*) je S-metaheuristika čija je osnovna ideja sistematska promena unapred definisanih okolina čime se izbegava zaglavljivanje pretrage u lokalnom optimumu. Najpoznatije P-heuristike su genetski algoritam, optimizacija rojevima čestica, optimizacija mravljim kolonijama i optimizacija kolonijom pčela. Genetski algoritam (*eng. Genetic algorithm, GA*) zasnovan je na simuliranju procesa biološke evolucije populacije jedinki, pri čemu proces pretraživanja skupa dopustivih rešenja odgovara procesu evolucije populacije jedinki. Optimizacija rojevima čestica (*eng. Particle swarm optimization, PSO*) predstavlja metaheurističku metodu inspirisanu inteligencijom grupe. Udruživanjem jedinki u grupe, interakcijom i razmenom znanja jedinki formira se kolektivna inteligencija koja utiče na veoma brz napredak čitave grupe. Optimizacija mravljim kolonijama (*eng. Ant colony optimization algorithms, ACO*) zasnovana je na ponašanju mrava u procesu pronalaženja hrane. Izučavanjem ove vrste insekata, uočeno je da mravi uvek pronalaze najkraći put od mravinjaka do izvora hrane što omogućava transportovanje hrane sa najmanjim utroškom vremena. Optimizacija kolonijom pčela (*eng. Bee Colony Optimization, BCO*) je P - heuristika koja je inspirisana ponašanjem pčela u potrazi za hranom. Glavna ideja je imitacija ponašanja kolonije prirodnih medonosnih pčela. Detaljniji opis i klasifikacija navedenih metaheuristika mogu se naći u radu [51].

Lokacijski problemi sa nadmetanjem i njegove varijacije rešavani su kako egzaktnim metodama, tako i heurističkim metodama. Heuristički pristup rešavanju lokacijskog problema sa nadmetanjem pomoću LS metaheurstike može se naći u radu [4], dok se u radu [55] nalazi druga varijanta lokacijskog problema sa nadmetanjem reše-

na metaheurističkim metodama lokalne pretrage i osnovnom metodom promeljivih okolina. Primena pohlepne (*eng. greedy*) metode za rešavanje lokacijsog problema sa nadmetanjem sa konkavnom potražnjom korisnika može se videti u radu [40] . U slučaju problema sa nadmetanjem, često se dešava da se optimizacioni problem jednog igrača rešava nekom egzaktnom metodom, a optimizacioni problem drugog igrača (njegov potproblem) rešava nekom metaheurističkom metodom. U radu [14] lokacijski problem sa nadmetanjem rešen je kombinacijom tabu pretrage i jedne egzaktne metode. Nasuprot ovakvom pristupu, moguće je rešavati optimizacione probleme oba igrača heurističkom metodom. Navedeni pristup može se naći u radu [5], gde je na oba optimizaciona problema primenjena PSO metaheuristika. U radu [49] predloženo je pet heuristika za rešavanje lokacijskog problema sa višestrukim nadmetanjem, gde se najbolje pokazala S - metaheuristika simuliranog kaljenja. Optimizacija rojevima čestica i genetski algoritam predloženi su za rešavanje lokacijskog problema sa nadmetanjem u radu [35], gde se bolje pokazao genetski algoritam. Još heurističkih pristupa za rešavanje lokacijskih problema sa nadmetanjem može se naći u radovima [50], [39], [45] i [31].

Glava 3

Lokacijski problemi ograničenih kapaciteta sa nadmetanjem sa delimičnim zadovoljenjem potražnje

Lokacijski problemi sa nadmetanjem (CFLP) su predmet izučavanja brojnih naučnih radova. U dosadašnjoj literaturi predložene su brojne varijante ovih problema. U ovom radu razmatran je lokacijski problem ograničenih kapaciteta sa nadmetanjem sa delimičnim zadovoljenjem potražnje (*eng. Capacitated competitive facility location problem with partial demand satisfaction, CCFLP-PDS*) koji je prvi put uveden u literaturi u radu [35]. Navedeni lokacijski problem po svojim karakteristikama predstavlja diskretni, egzogeni, statički, lokacijsko-alokacijski problem ograničenih kapaciteta sa nadmetanjem.

Pretpostavlja se da postoje dve rivalske kompanije (takmičara, igrača) i unapred određen i fiksiran broj korisnika na tržištu. Navedene dve kompanije nameravaju da usposatave nove objekte na unapred poznatim lokacijama sa ciljem da prikupe što više kupaca kako bi uvećale svoj profit na tržištu. Broj objekata koji kompanije mogu da uspostave nije fiksan, a kapacitet objekta koji će uspostaviti kompanije mogu birati iz unapred ponuđenih kapaciteta. Svaki korisnik se pridružuje najbliže uspostavljenom objektu koji zadovoljava njegovu celu ili deo potražnje. Drugim rečima, na izbor objekata od strane korisnika utiču i blizina i kapacitet objekta.

Problem je definisan u dva nivoa - gornji i donji. Donji predstavlja ugnježden (unutrašnji) problem. Svaki nivo predstavlja optimizaciju ciljne funkcije svog takmičara. Pretpostavlja se da oba igrača pametno biraju svoje poteze, tj. da biraju najbolje moguće lokacije za uspostavljanje svojih objekata, a da su pritom svesni

rizika da izgube određen deo profita time što prepuštaju neke druge lokacije svom suparniku. U ovoj igri takmičari povlače svoje poteze jedan pa drugi, tj. lider prvi bira lokacije na kojima će uspostaviti svoje objekte, a potom sledbenik bira lokacije za svoje objekte imajući u vidu koje lokacije je izabrao lider. Drugim rečima prvi igrač je dominantan u odnosu na drugi. Kao što je u tekstu ranije navedeno, ovaj tip igre kao rezultat ima Stakelbergov ekvilibrijum gde se dva problema optimizacije rešavaju na dva nivoa. Bolje je rešenje ono koje ima veću vrednost funkcije cilja lidera, bez obzira na vrednost funkcije cilja sledbenika. Ukoliko su vrednosti funkcije cilja lidera za dva rešenja ista, onda je bolje rešenje ono koje ima veću vrednost funkcije cilja sledbenika. Navedena igra predstavlja igru sa nenultom sumom, tj. učesnici sabiraju svoje dobitke i gubitke do neke vrednosti različite od nule koja ne mora biti konstanta, pri čemu napredovanje jednog učesnika ne mora obavezno da znači nazadovanje drugog.

U radu [35] predložena su dva modela navedenog problema. U oba modela takmičari biraju lokacije gde će uspostaviti svoje objekte, kapacitete datih objekata, kao i korisnike koji će im se pridružiti. U prvom modelu se pretpostavlja da ukoliko neki sledbenikov objekat nema dovoljan kapacitet da zadovolji celu potražnju korisnika, korisnik se ne može pridružiti datom objektu. U drugom modelu se ovaj uslov popušta za sledbenika i njemu se dozvoljava delimično zadovoljenje potražnje korisnika. Drugim rečima, ukoliko sledbenikov objekat nije dalji od liderovog objekta i ima dovoljan kapacitet da zadovolji određen deo potražnje, sledbenik ima pravo da preuzme od lidera celu potražnju kupca ili samo deo potražnje. Na ovaj način korisnik će deo svoje potražnje zadovoljiti kod lidera, a deo kod sledbenika. Međutim, može se desiti i slučaj da se korisnik nije alocirao liderovom objektu, a da sledbenik zadovolji samo deo potražnje kupca. Na ovaj način korisnik biva nezadovoljan, jer mu deo potražnje ostaje nezadovoljen. Problem se rešava u tri faze:

- U prvoj fazi lider bira lokacije na kojima će uspostaviti svoje objekte i kapacitete navedenih objekata. Pri izboru lokacija, lider ima u vidu činjenicu da će deo potražnje kupaca preuzeti sledbenik, ukoliko izabere lokaciju koja je bliža korisniku nego liderov objekat. Ove pretpostavke čine lidera pametnim igračem.

- U drugoj fazi sledbenik bira najbolje lokacije za svoje objekte imajući u vidu liderov izbor.

- U trećoj fazi, korisnici imaju informacije o uspostavljenim liderovim i sledbenikovim objektima, njihovim kapacitetima i na osnovu toga biraju objekte koji su im najbliži, a koji zadovoljavaju njihovu potražnju. Na taj način korisnici takmičarskim kompanijama vraćaju profit.

3.1 Matematička formulacija problema

U cilju predstavljanja matematičke formulacije, uvedene su sledeće oznake.

Skupovi

- I - skup svih potencijalnih lokacija za uspostavljanje novih objekata, $I = \{0, \ldots, n-1\}$
- I^L - skup svih potencijalnih lokacija za uspostavljanje novih objekata od strane lidera $I^L \subseteq I$
- I^F - skup svih potencijalnih lokacija za uspostavljanje novih objekata od strane sledbenika $I^F \subseteq I$
- J - skup kupaca $J = \{0, \ldots, m-1\}$
- S_i - skup kapaciteta za objekte koji mogu biti uspostavljeni na lokaciji $i \in I$, $s \in S_i = \{0, \ldots, k_i - 1\}$, gde je k_i broj različitih kapaciteta objekta koji se mogu uspostaviti na lokaciji i

Ulazni parametri

- p_{ij} - prihod lidera koji ostvaruje ako kupac $j \in J$ celu svoju potražnju zadovolji liderovim objektom na lokaciji $i \in I^L$
- q_{ij} - prihod sledbenika koji ostvaruje ako kupac $j \in J$ celu svoju potražnju zadovolji sledbenikovim objektom na lokaciji $i \in I^F$
- w_j - celokupna potražnja kupca $j \in J$
- f_{is} - fiksni troškovi potrebni za uspostavljanje objekta kapaciteta $s \in S_i$ na lokaciji $i \in I^L$ od strane lidera
- g_{is} - fiksni troškovi potrebni za uspostavljanje objekta kapaciteta $s \in S_i$ na lokaciji $i \in I^F$ od strane sledbenika

- lc_{is} - maksimalna količina potražnje koja može biti zadovoljena uspostavljanjem objekta kapaciteta $s \in S_i$ na lokaciji $i \in I^L$ od strane lidera
- fc_{is} - maksimalna količina potražnje koja može biti zadovoljena uspostavljanjem objekta kapaciteta $s \in S_i$ na lokaciji $i \in I^F$ od strane sledbenika
- d_{ij} - rastojanje između kupca $j \in J$ i lokacije $i \in I$
- dc_{ij} - transportni troškovi potrebni da bi se zadovoljila cela potražnja kupca $j \in J$ objektom na lokaciji $i \in I$
- M - dovoljno veliki broj

Promenljive

$$X_{is} = \begin{cases} 1, & \text{ako je lider uspostavio objekat kapaciteta } s \text{ na lokaciji } i \\ 0, & \text{inače} \end{cases} \tag{3.1}$$

$$x_{ij} = \begin{cases} 1, & \text{ako je potražnja kupca } j \text{ zadovoljena objektom na lokaciji } i \\ & \text{uspostavljenim od strane lidera} \\ 0, & \text{inače} \end{cases} \tag{3.2}$$

$$Z_{is} = \begin{cases} 1, & \text{ako je sledbenik uspostavio objekat kapaciteta } s \text{ na lokaciji } i \\ 0, & \text{inače} \end{cases} \tag{3.3}$$

$$z_{ij} = \begin{cases} 1, & \text{ako je cela potražnja ili deo potražnje kupca } j \text{ zadovoljena} \\ & \text{objektom na lokaciji } i \text{ koja je uspostavljena od strane sledbe-} \\ & \text{nika} \\ 0, & \text{inače} \end{cases} \tag{3.4}$$

γ_{ij} - deo potražnje kupca j koji je alociran objektu na lokaciji i uspostavljenom od strane sledbenika (koristi se samo u Modelu 2), $\gamma_{ij} \in [0,1], \gamma_{ij} \in \mathbb{R}$.

3.2 Matematička formulacija Modela 1

Kada je poznat potez lidera, odnosno kada je poznato X_{is} i x_{ij}, rešava se optimizacioni problem sledbenika. Optimalno rešenje sledbenika označeno je promenljivama $\tilde{Z}_{is}, \tilde{z}_{ij} \in \{0,1\}$. Ako je $\tilde{z}_{ij} = 1$ onda se korisnik j vezao za sledbenikov objekat

i i tim objektom zadovoljio celu potražnju korisnika j, u suprotnom ova promenljiva ima vrednost 0. Nije potrebno eksplicitno definisati promenljivu $\tilde{Z}_{is}$ obzirom da $\tilde{Z}_{is}$ utiče na ciljnu funkciju lidera (3.5) kroz promenljivu $\tilde{z}_{ij}$. Imajući u vidu navedene oznake, u nastavku se nalazi matematička formulacija lokacijskog problema ograničenih kapaciteta sa nadmetanjem sa zadovoljenjem cele potražnje korisnika.

L (Lider):

$$max \quad \sum_{j\in J}(\sum_{i\in I^L}(p_{ij} - dc_{ij})\, x_{ij})\,(1 - \sum_{i\in I^F}\tilde{z}_{ij}) - \sum_{i\in I^L, s\in S_i} f_{is}X_{is} \tag{3.5}$$

Pri ograničenjima:

$$\sum_{i\in I^L} x_{ij} \leq 1, \qquad \forall j \in J \tag{3.6}$$

$$\sum_{s\in S_i} X_{is} \leq 1, \qquad \forall i \in I^L \tag{3.7}$$

$$\sum_{j\in J} w_j x_{ij} \leq \sum_{s\in S_i} lc_{is}X_{is}, \qquad \forall i \in I^L \tag{3.8}$$

$$X_{is}, x_{ij} \in \{0,1\}, \qquad \forall i \in I^L, \forall j \in J, \forall s \in S_i \tag{3.9}$$

S (Sledbenik):

$$max \quad \sum_{i\in I^F, j\in J}(q_{ij} - dc_{ij})\, z_{ij} - \sum_{i\in I^F, s\in S_i} g_{is}Z_{is} \tag{3.10}$$

Pri ograničenjima:

$$\sum_{i\in I^F} z_{ij} \leq 1, \quad \forall j \in J \tag{3.11}$$

$$\sum_{j\in J} w_j z_{ij} \leq \sum_{s\in S_i} fc_{is}Z_{is}, \quad \forall i \in I^F \tag{3.12}$$

$$\sum_{s\in S_i} Z_{is} \leq 1, \quad \forall i \in I^F \backslash (I^L \cap I^F) \tag{3.13}$$

$$\sum_{s\in S_i} X_{is} + \sum_{s\in S_i} Z_{is} \leq 1, \quad \forall i \in (I^L \cap I^F) \tag{3.14}$$

$$\sum_{i\in I^F} d_{ij}z_{ij} \leq \sum_{i\in I^L} d_{ij}x_{ij} + M(1 - \sum_{i\in I^L} x_{ij}), \quad \forall j \in J \tag{3.15}$$

$$Z_{is}, z_{ij} \in \{0,1\}, \qquad \forall i \in I^F, \forall j \in J, \forall s \in S_i \tag{3.16}$$

Problem predstavlja dvonivovski problem u kojem formule (3.5 - 3.9) predstavljaju optimizacioni problem lidera (gornji problem), dok formule (3.10 - 3.16) predstavljaju optimizacioni problem sledbenika (donji problem). Ciljna funkcija lidera direktno zavisi od lokacija koje je izabrao sledbenik i alokacija korisnika sledbenika, što se uvodi kroz promenljive $\tilde{z}_{ij}$. Drugim rečima, lider uzima u obzir najbolju reakciju konkurenta, koja se oslikava kroz promeljivu $\tilde{z}_{ij}$ u ciljnoj funkciji lidera (3.5), pod pretpostavkom da i konkurent preduzima najbolje rešenje (optimalno rešenje).

Funkcija cilja lidera (3.5) predstavlja profit koji lider ostvaruje od korisnika koji su pridruženi objektima koje je uspostavio, od čega se oduzimaju troškovi uspostavljanja objekata i troškovi transporta. Uslovi (3.6) garantuju da se svaki kupac može pridružiti najviše jednom objektu lidera. Na svakoj lokaciji može se uspostaviti objekat samo jednog od ponuđenih kapaciteta od strane lidera, što omogućavaju uslovi (3.7). Za svaki objekat, uslovi (3.8) obezbeđuju da potražnja kupaca koji su pridruženi objektu koji je uspostavio lider budu manji ili jednaki od kapaciteta navedenog objekta, kako bi se obezbedilo zadovoljenje potražnje kupaca vezanih za taj objekat. Promenljive X_{is} i x_{ij} su binarne, što garantuju uslovi (3.9).

Funkcija cilja sledbenika (pratioca) (3.10) predstavlja profit koji sledbenik ostvaruje od korisnika koji su pridruženi njegovim objektima od čega se oduzimaju troškovi uspostavljanja objekata i troškovi transporta. Uslovi (3.11) govore da svaki kupac može da se pridruži najviše jednom objektu koji je uspostavio sledbenik. Uslovi (3.12) obezbeđuju da za svaki sledbenikov objekat potražnja svih korisnika koji su vezani za njega ne sme biti veća od kapaciteta navedenog objekta. Na lokaciji i može se uspostaviti objekat tačno jednog kapaciteta, što se garantuje uslovima (3.13). Na lokaciji i može se uspostaviti objekat najviše od strane jednog igrača, ili lidera ili sledbenika, što je formulisano uslovima (3.14). S obzirom da prvo igra lider, tj. lider prvi izabere lokacije na kojima će uspostaviti objekte, uslovi (3.15) obezbeđuju da ukoliko sledbenik izabere da uspostavi objekat na lokaciji koja je korisniku bliža od liderovih uspostavljenih objekata ili jednako udaljena od liderovih objekata, korisnik će se dealocirati sa liderovog objekta i pridružiti sledbenikovom objektu, pod uslovom da objekat sledbenika zadovoljava potražnju datog kupca. Promenljive Z_{is} i z_{ij} su binarne, što se vidi u uslovima (3.16). Za Model 1 prostor dopustivih rešenja čine ona rešenja kod kojih su svi korisnici zadovoljili svoju celu potražnju.

Kao što je već rečeno, model 1 je dvonivovski model, gde je liderov problem

gornji problem, a sledbenikov donji (unutrašnji). Lider prvi bira svoj potez, i za njega je od važnog značaja zaključiti da li su sledbenikovi potezi „blagonakloni" ili „pakosni" prema lideru. U igrama koje nisu sume nula ili sume k ne postoji mogućnost odrediti prirodu poteza sledbenika, odnosno nije moguće odrediti da li je on „blagonaklon" ili „pakostan" prema lideru. Kako bi se odredila priroda sledbenika u ovakvim slučajevima definiše se dopunski model. U literaturi se javljaju dva načina da se definiše dopunski model. Prvi je da se dopunski model definiše kao odvojen model, pa se problem svodi na tronivovski model, dok je drugi pristup da se donji problem predefiniše tako da bude bikriterijumski u kojem se poželjna rešenja dobijaju a priori leksikografskom metodom. Pokazalo se da je rešavanje dopunskog modela vremenski zahtevno, kao i da ne pravi značajne promene kada je reč o strukturi rešenja. Primer navedenih tvrdnji mogu se videti u radu [7].

Izložen Model 1 nije model sume nula niti model k sume. U cilju sagledavanja prirode ponašanja sledbenika može se razviti dopunski model. Međutim, kako je u ovom radu akcenat na rešavanju dvonivovskog modela koristeći metaheurstičke metode, a rešavanje dopunskog modela vremenski zahtevno, pri čemu se pokazalo da dopunski model ne pravi značajne promene u rešenjima, u ovom radu nije razvijen dopunski model, već je fokus na rešavanju dvonivovskog problema koristeći metaheurstičke metode.

3.3 Matematička formulacija Modela 2

U nastavku se nalazi matematička formulacija lokacijskog problema ograničenih kapaciteta sa nadmetanjem sa delimičnim zadovoljenjem potražnje. Optimalno rešenje sledbenika označeno je promenljivama $\tilde{Z}_{is}$ $\tilde{z_{ij}}$ $\tilde{\gamma_{ij}}$. Međutim, nije potrebno eksplicitno definisati promenljive $\tilde{Z}_{is}$ i $\tilde{z_{ij}}$, jer one utiču na ciljnu funkciju lidera (3.17) kroz promenljivu $\tilde{\gamma_{ij}}$.

L (Lider):

$$max \quad \sum_{j\in J}\Big(\sum_{i\in I^L}(p_{ij}-dc_{ij})\,x_{ij}\Big)\,\Big(1-\sum_{i\in I^F}\tilde{\gamma}_{ij}\Big)\;-\sum_{i\in I^L,s\in S_i} f_{is}X_{is} \tag{3.17}$$

Pri ograničenjima:

$$\sum_{i\in I^L} x_{ij} \leq 1, \qquad \forall j \in J \tag{3.18}$$

$$\sum_{s \in S_i} X_{is} \leq 1, \qquad \forall i \in I^L \tag{3.19}$$

$$\sum_{j \in J} w_j x_{ij} \leq \sum_{s \in S_i} lc_{is} X_{is}, \qquad \forall i \in I^L \tag{3.20}$$

$$X_{is}, x_{ij} \in \{0, 1\}, \qquad \forall i \in I^L, \forall j \in J, \forall s \in S_i \tag{3.21}$$

S (Sledbenik):

$$max \quad \sum_{i \in I^F, j \in J} (q_{ij} - dc_{ij}) \gamma_{ij} - \sum_{i \in I^F, s \in S_i} g_{is} Z_{is} \tag{3.22}$$

Pri ograničenjima:

$$\sum_{i \in I^F} z_{ij} \leq 1, \quad \forall j \in J \tag{3.23}$$

$$\sum_{j \in J} w_j \gamma_{ij} \leq \sum_{s \in S_i} fc_{is} Z_{is}, \quad \forall i \in I^F \tag{3.24}$$

$$\sum_{s \in S_i} Z_{is} \leq 1, \quad \forall i \in I^F \backslash (I^L \cap I^F) \tag{3.25}$$

$$\sum_{s \in S_i} X_{is} + \sum_{s \in S_i} Z_{is} \leq 1, \quad \forall i \in (I^L \cap I^F) \tag{3.26}$$

$$d_{ij} z_{ij} \leq \sum_{i' \in I^L} d_{i'j} x_{i'j} + M(1 - \sum_{i' \in I^L} x_{i'j}), \quad \forall i \in I^F, \forall j \in J \tag{3.27}$$

$$\gamma_{ij} \leq z_{ij}, \quad \forall i \in I^F, \forall j \in J \tag{3.28}$$

$$Z_{is}, z_{ij} \in \{0, 1\}, \qquad \forall i \in I^F, \forall j \in J, \forall s \in S_i \tag{3.29}$$

$$\gamma_{ij} \geq 0, \quad \forall i \in I^F, \forall j \in J \tag{3.30}$$

Funkcija cilja lidera (3.17) predstavlja profit koji lider ostvaruje od korisnika koji su pridruženi njegovim objektima od čega se oduzimaju troškovi uspostavljanja objekata i troškovi transporta. Uslovi (3.18) obezbeđuju da se svaki kupac može pridružiti najviše jednom objektu lidera. Na nekoj lokaciji može se uspostaviti objekat samo jednog od ponuđenih kapaciteta, što se vidi u uslovima (3.19). Za svaki

objekat, uslovi (3.20) garantuju da potražnja kupaca koji su pridruženi objektu koji je uspostavio lider budu manji ili jednaki od kapaciteta navedenog objekta, kako bi se obezbedilo zadovoljenje potražnje kupaca. Promenljive X_{is} i x_{ij} su binarne, što se vidi u uslovima (3.21).

Funkcija cilja sledbenika (pratioca) (3.22) predstavlja profit koji sledbenik ostvaruje od korisnika koji su pridruženi njegovim objektima od čega se oduzimaju troškovi uspostavljanja objekata i troškovi transporta. Uslovi (3.23) obezbeđuju da svaki kupac može da se pridruži najviše jednom objektu koji je uspostavio sledbenik. Za svaki objekat potražnja ili deo potražnje svih korisnika koji su vezani za sledbenikov objekat ne sme biti veća od kapaciteta navedenog objekta, što je uslovljeno nejednakostima (3.24). Uslovi (3.25) garantuju da se na lokaciji i može uspostaviti objekat tačno jednog kapaciteta. Uslovi (3.26) omogućavaju da se na lokaciji i može uspostaviti objekat najviše od strane jednog od igrača, ili lidera ili sledbenika. S obzirom da prvo igra lider, tj. lider prvi izabere lokacije na kojima će uspostaviti objekte, uslovi (3.27) obezbeđuju da ukoliko sledbenik izabere da uspostavi objekat na lokaciji koja je korisniku bliža od liderovih uspostavljenih objekata ili jednako udaljena od liderovih objekata, korisnik može dealocirati deo ili celu svoju potražnju sa liderovog objekta na sledbenikov objekat. Kako promenljiva z_{ij} govori o tome da li je kupac j vezan za sledbenikov objekat i, a promenljiva γ_{ij} označava deo potražnje kupca j koji je zadovoljen sledbenikovim objektom i, uslovi (3.28) obezbeđuju da deo potražnje kupca koji se zadovolji određenim objektom ne bude veći od cele potražnje datog kupca. Promenljive Z_{is} i z_{ij} su binarne, a promenljiva γ_{ij} je realan broj veći ili jednak nuli, što obezbeđuju uslovi (3.29) i (3.30). Za model 2 prostor dopustivih rešenja čine ona rešenja kod kojih su svi korisnici zadovoljili celu svoju potražnju kod lidera ili bar deo svoje potražnje kod sledbenika.

Ni Model 2 nije model sume nula niti model k sume, pa se u cilju sagledavanja prirode ponašanja sledbenika može razviti dopunski model. Kao što je već rečeno, pokazalo se da je dopunski model vremenski zahtevan za rešavanje, a da ne pravi značajne razlike kada je reč o rešenjima, tako da se u ovom radu, kao i za Model 1, neće razvijati dopunski model, već će fokus biti na rešavanju dvonivovskog problema metaheurističkim metodama.

U sekvencijalnim modelima, kao što su Model 1 i Model 2, može se primeniti strategija odvraćanja od ulaska (*eng. Entry Deterrance*). Navedena strategija podrazumeva da kompanija koja se nalazi na tržištu deluje na način da obeshrabri ulazak novih potencijalnih kompanija na tržište. U izloženim modelima strategija

odvraćanja od ulaska stvara prepreke za sledbenika da uđe na tržište, a lideru omogućava da zadrži veliki udeo tržišta, ili pak celo tržište za sebe. Neke od strategija odvraćanja mogu biti uključenje viška kapaciteta ili smanjenje cene proizvoda. Na primer, u izloženim Modelima 1 i 2 moguće je primeniti strategiju odvraćanja ulaska povećanjem kapaciteta. Strateški višak kapaciteta može se uspostaviti kako bi se smanjila održivost ulaska potencijalnih firmi. Do viška kapaciteta dolazi kada firma koja se nalazi na tržištu preti ostalim kompanijama (u ovom slučaju sledbeniku) da će povećati svoju proizvodnju, time uspostaviti višak ponude, a zatim smanjiti cenu proizvoda tako da konkurentska kompanija ne može da se bori sa tim. Iako bi kratkoročno gledano ove strategije mogle dovesti do neefikasnog poslovanja, dugoročno gledano lider će na ovaj način imati veću korist, jer će imati veći udeo na tržištu. Više o strategiji odvraćanja od ulaska i njenoj primeni može se naći u radovima [48] i [27].

U ovom radu korišćene su pretpostavke da je skup potencijalnih lokacija za uspostavljanje objekata i skup lokacija korisnika isti, tj. $I = J$, zatim i da je skup lokacija na kojima lider može uspostaviti svoje objekte isti kao skup lokacija na kojima sledbenik može uspostaviti svoje objekte, tj. $I = I^F = I^L$. Takođe, korišćena je pretpostavka da je broj ponuđenih kapaciteta isti za svaki objekat, tj. $k_i = k, \forall i \in I$, gde je k broj ponuđenih kapaciteta.

3.4 Primeri

Izloženi Modeli 1 i 2 biće ilustrovani na primerima instanci malih dimenzija. Neka se na tržištu nalaze dva konkurenta, lider i sledbenik. Na raspolaganju su im 4 potencijalne lokacije za uspostavljanje svojih objekata. Na tržištu je potrebno zadovoljiti 4 korisnika. Lokacije korisnika i potencijalne lokacije za uspostavljanje objekata su iste. Instanca dimezije |I| = |J| = 4 prikazane su na slici 3.1.

p_{ij}	j=0	j=1	j=2	j=3
i=0	22	20	17	13
i=1	22	20	17	13
i=2	22	20	17	13
i=3	22	20	17	13

f_{is}	s=0	s=1
i=0	21	33
i=1	23	35
i=2	25	35
i=3	21	30

q_{ij}	j=0	j=1	j=2	j=3
i=0	22	20	17	13
i=1	22	20	17	13
i=2	22	20	17	13
i=3	22	20	17	13

g_{is}	s=0	s=1
i=0	17	30
i=1	17	26
i=2	19	29
i=3	20	29

w_j	j=0	j=1	j=2	j=3
	5	9	5	8

l_{cis}	s=0	s=1
i=0	25	30
i=1	22	30
i=2	22	31
i=3	21	34

d_{ij}	j=0	j=1	j=2	j=3
i=0	0	4	6	5
i=1	4	0	2	6
i=2	6	2	0	4
i=3	5	6	4	0

f_{cis}	s=0	s=1
i=0	9	20
i=1	11	17
i=2	11	16
i=3	12	15

Slika 3.1: Instanca dimezije $n = m = 4, k = 2$

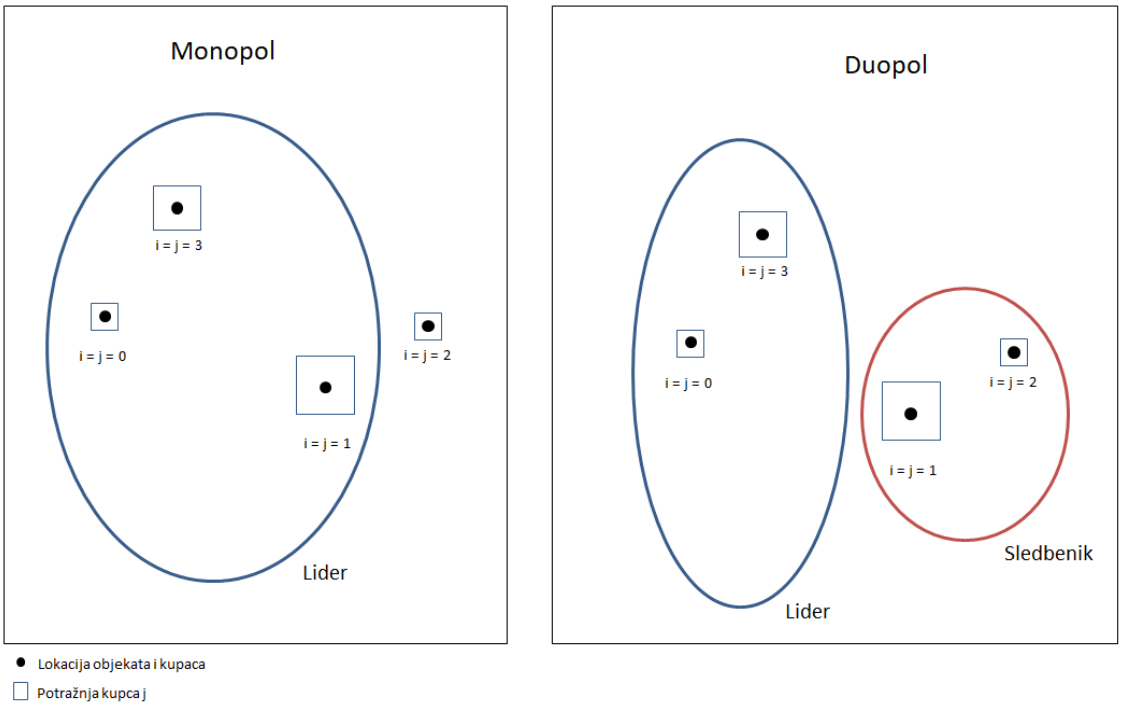
U radu važi jednakost $d_{c_{ij}} = \alpha \cdot d_{ij}$, gde je $\alpha = 0.01$. U nastavku će biti izloženo optimalno rešenje Modala 1 i Modela 2 razmatranog problema za prikazanu instancu.

Model 1

U slučaju da se lider nalazi sam na tržištu i uspostavi objekat kapaciteta $s = 0$ na lokaciji $i = 0$, tada bi se za njega vezali korisnici $j = 0, j = 1, j = 3$, što je prikazano na slici 3.2 levo. Kapacitet ovako uspostavljenog objekta iznosi $lc_{00} = 25$, a potražnja korisnika iznosi $w_0 = 5$, $w_1 = 9$ i $w_3 = 8$. Zbir potražnje ova tri korisnika iznosi 22, tako da se ona mogu vezati za liderov objekat $i = 0$. Preostali kapacitet objekta $i = 0$ iznosi 3. Kako potražnja korisnika $j = 2$ iznosi $w_2 = 5$, uočava se da liderov objekat nema dovoljno kapaciteta da zadovolji potražnju kupca $j = 2$.

Kada na tržište dođe i konkurentska firma i uspostavi objekat kapaciteta $s = 1$ na lokaciji $i = 1$, situacija na tržištu se menja. Kapacitet ovako uspostavljenog

objekta iznosi $fc_{11} = 17$. Kako je potražnja korisnika $j = 2$ iznosi $w_2 = 5$, korisnik $j = 2$ se alocira sledbenikovom objektu, jer ima dovoljno kapaciteta da zadovolji celu potražnju korisnika. Preostali kapacitet sledbenikovog objekta iznosi 12. Ukoliko su korisnici koji su vezani za lidera bliži sledbenikovom objektu, a sledbenikov objekat ima dovoljan kapacitet da zadovolji **celu potražnju** korisnika, korisnik će se delocirati sa liderovog objekta i alocirati sledbenikovom objektu. Imajući u vidu da je korisnik $j = 1$ bliži sledbenikovom objektu, i da njegova potražnja iznosi $w_1 = 9$ sledbenik će preuzeti datog korisnika s obzirom da ima dovoljan kapacitet da zadovolji celu potražnju korisnika. Situacija na tržištu na kom vlada duopol prikazana je desno na slici 3.2. Optimalna vrednost funkcija cilja liderovog i sledbenikovog optimizacionog problema data je parom $(L, S) = (13.95, 10.98)$.



Slika 3.2: Tržište - Model 1

Model 2

U slučaju da se lider nalazi sam na tržištu i uspostavi objekat kapaciteta $s = 0$ na lokaciji $i = 0$, tada bi se za njega vezali korisnici $j = 0, j = 1, j = 3$, što je prikazano levo na slici 3.3. Kapacitet ovako uspostavljenog objekta iznosi $lc_{00} = 25$, a potražnja korisnika iznosi $w_0 = 5$, $w_1 = 9$ i $w_3 = 8$. Zbir potražnje ova tri korisnika iznosi 22, tako da se ona mogu vezati za liderov objekat $i = 0$. Preostali kapacitet

objekta $i = 0$ iznosi 3. Kako potražnja korisnika $j = 2$ iznosi $w_2 = 5$, uočava se da liderov objekat nema dovoljno kapaciteta da zadovolji potražnju kupca $j = 2$.

Kada na tržište dođe i konkurentska firma i uspostavi objekat kapaciteta $s = 0$ (manjeg kapaciteta nego u primeru u Modelu 1) na lokaciji $i = 1$, situacija na tržištu se menja. Kapacitet ovako uspostavljenog objekta iznosi $fc_{10} = 11$. Kako je potražnja korisnika $j = 2$ iznosi $w_2 = 5$, korisnik $j = 2$ se alocira sledbenikovom objektu, jer ima dovoljno kapaciteta da zadovolji celu potražnju korisnika. Preostali kapacitet objekta $i = 1$ iznosi 6. Ukoliko su korisnici koji su vezani za lidera bliži sledbenikovom objektu, a sledbenikov objekat ima dovoljan kapacitet da zadovolji makar **deo potražnje** korisnika, korisnik će delocirati deo svoje potražnje sa liderovog objekta i alocirati taj deo sledbenikovom objektu. Imajući u vidu da je korisnik $j = 1$ bliži sledbenikovom objektu, on ima tendenciju da pređe kod sledbenika kao što je opisano u Modelu 1. Ako sledbenik nema dovoljan kapacitet da zadovolji celu potražnju korisnika, on će zadovoljiti samo deo potražnje korisnika, a ostatak će ostati kod lidera. U navedenom primeru, potražnja korisnika $j = 1$ je 9, tako da će korisnik $3/9$ svoje potražnje zadovoljiti kod lidera, a $6/9$ kod sledbenika. Navedeni korisnik će sledbeniku doneti profit proporcionalan količini potražnje korisnika koju je sledbenik zadovoljio. Analogno važi i za lidera. Situacija na tržištu na kom vlada duopol prikazana je desno na slici 3.3.

Optimalna vrednost funkcije cilja lidera data je sledećom formulom.

$$(p_{0,0} - d_{C_{0,0}}) \cdot (1 - \gamma_{0,0}) + (p_{0,1} - d_{C_{0,1}}) \cdot (1 - \gamma_{0,1}) + (p_{0,3} - d_{C_{0,3}}) \cdot (1 - \gamma_{0,3}) - f_{0,0} =$$

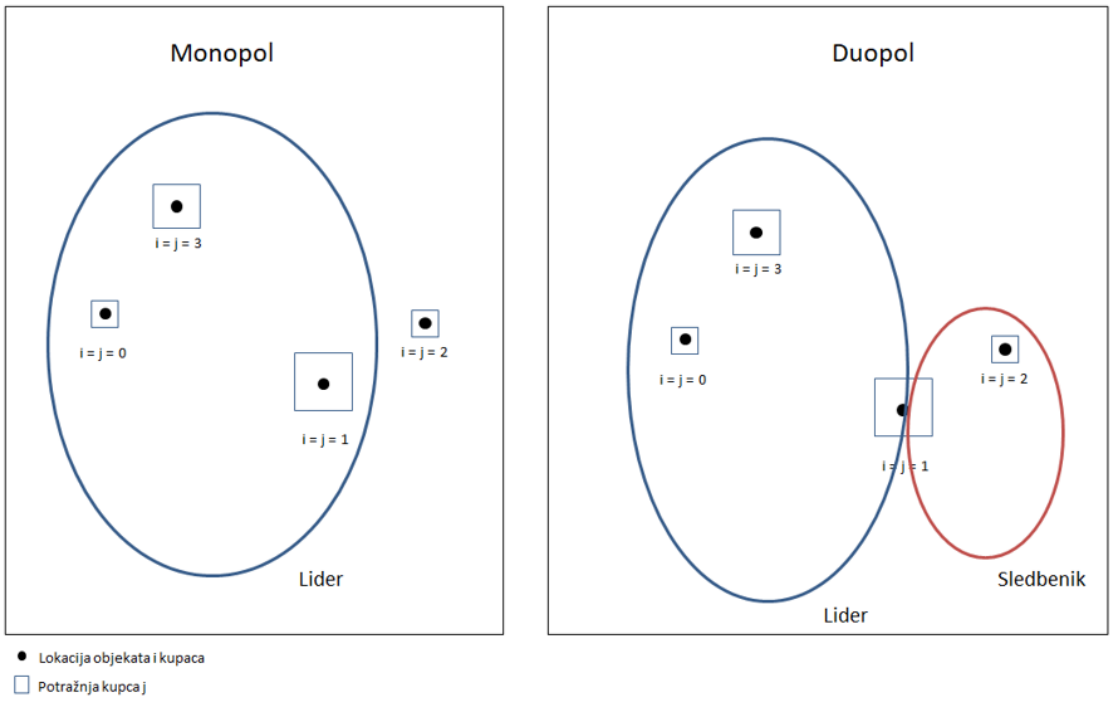
$$= (22 - 0) \cdot 1 + (20 - 0.04) \cdot 3/9 + (13 - 0.05) \cdot 1 - 21 = 20.6$$

Optimalna vrednost funkcije cilja sledbenika data je sledećom formulom.

$$(q_{1,1} - d_{C_{1,1}}) \cdot \gamma_{1,1} + (q_{1,2} - d_{C_{1,2}}) \cdot \gamma_{1,2} - g_{1,0}$$

$$= (20 - 0) \cdot 6/9 + (17 - 0.02) \cdot 1 - 17 = 13.31$$

Optimalna vrednost funkcija cilja liderovog i sledbenikovog optimizacionog problema data je parom $(L, F) = (20.6, 13.31)$.



Slika 3.3: Tržište - Model 2

Uvođenjem uslova da se može zadovoljiti samo deo potražnje korisnika, sledbeniku se pozicija na tržištu poboljšava. U primeru koji je naveden, lider ništa ne gubi, čak i dobija ovakvom igrom sledbenika, međutim može se desiti i obrnuti slučaj, što je češći scenario. U Modelu 1, sledbenik ne može preoteti navedenog korisnika lideru, ako ne može da zadovolji celu njegovu potražnju, dok u Modelu 2 može, čime automatski oduzima deo profita koji je lideru donosio korisnik j i preuzima za sebe deo tog profita. Na taj način, lideru se smanjuje očekivani profit, dok se sledbeniku povećava. Zanimljivost ove igre ogleda se tome što lider unapred zna da će na tržište doći konkurentska kompanija i u skladu sa tim bira lokacije na kojima će uspostaviti svoje objekte, kao i njihove kapacitete.

Glava 4

Genetski algoritam za rešavanje CFLP-PDS

4.1 Osnovne postavke GA

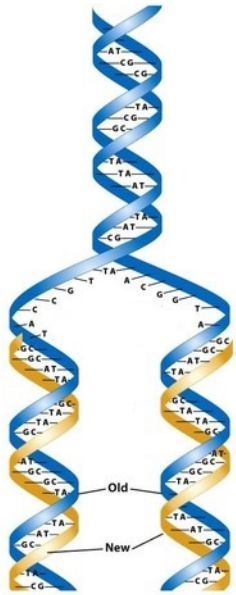
Genetski algoritmi predstavljaju robusne i adaptivne metode koje se između ostalog koriste za rešavanje problema kombinatorne optimizacije. Darvinova [1] teorija o postanku vrsta i prirodnoj evoluciji nastala je krajem 19. veka, a genetski algoritmi (*skr. GA*) zasnovani su na njoj i na činjenici da unutar jedne populacije najčešće opstaju najbolje prilagođene jedinke. Više o teoriji evolucije može se pročitati u radovima [6] i [46]. Prve ideje o ovim algoritmima izložio je Holland u radu [15].

Gen je fizička i funkcionalna jedinica nasleđivanja, koja prenosi naslednu poruku iz generacije u generaciju. Geni su nanizani duž hromozoma, odnosno oni su linearno raspoređeni delovi hromozomske dezoksiribonukleinske kiseline (*skr. DNK*). Njihova veličina (broj nukleotida DNK) i raspored na hromozomima su strogo određeni, a građa gena ogleda se u tačno određenom redosledu nukleotida (A - adenin, G - guanin, C - citozin i T - timin).

Iz postojeće populacije, tokom prirodnog procesa nova jedinka nastaje ukrštanjem gena svojih roditelja, pri čemu postoji verovatnoća mutacije nekih gena. Stvaranje nove jedinke naziva se reprodukcija. Novonastala jedinka nasleđuje neka svojstva jednog roditelja, a neka svojstva drugog roditelja. Promena redosleda nukleoti-

[1] Čarls Robert Darvin (eng. Charles Darwin, 1809 - 1882) bio je britanski biolog, prirodnjak i geolog. Postavio je temelje moderne teorije evolucije po kojoj sve svi životni oblici razvijaju putem prirodne selekcije. Utvrdio je da su sve životne vrste nastale tokom vremena od zajedničkog pretka. Objavio je svoju knjigu o poreklu vrsta 1859. godine. Darvin se smatra jednim od najuticajnijih ličnosti u ljudskoj istoriji.

da, manjak ili višak nukleotida rezultira promenom funkcije gena i naziva se genska mutacija. Ilustracija reprodukcije (replikacije) prikazana je slikom 4.1.



Slika 4.1: Replikacija DNK

Imajući u vidu da je genetski algoritam populaciona heurističa metoda, osnovni koncept ove metode čini populacija određene veličine, odnosno sa određenim brojem jedinki u njoj. Skup određenog broja dopustivih rešenja naziva se populacija, a jedno rešenje predstavlja jedinku unutar te populacije. Kod jedinke predstavlja skup i redosled njenih gena. Svakoj jedinki se dodeljuje vrednost funkcije pogodnosti (prilagođenosti) na osnovu koje se određuje koliko je jedinka prilagođena trenutnom stanju okruženja i ona određuje kvalitet jedinke. Tokom vremena, stvara se nova populacija reprodukovanjem jedinki sa boljom prilagođenošću i na taj način se postiže formiranje jedinki koje su bolje prilagođene trenutnom stanju okruženja. Nakon selekcije roditelja, nove jedinke se reprodukuju primenom osnovnih genetskih operatora: ukrštanjem i mutacijom. Cilj genetskog algoritma je da se, počev od početne populacije, iz generacije u generaciju stvaraju jedinke koje imaju sve bolju prilagođenost, kako bi se u dovoljno velikom broju iteracija došlo do zadovoljavajuće dobrog ili optimalnog rešenja. Više o genetskom algoritmu i njegovim brojnim primenama na rešavanje lokacijskih problema može se naći u radovima [36], [37], [9], [12] i [43].

Neka je X prostor dopustivih rešenja. Svakom rešenju iz prostora X se dodeljuje odgovarajući kod. Kodiranje jedinke zavisi od prirode problema. To može biti niz

binarnih, celobrojnih, realnih brojeva ili niz nekih drugih simbola. Neka je N veličina populacije, tj. broj jedinki u populaciji. Skup $P_n = \{x_{1n}, \ldots, x_{Nn}\}$ predstavlja populaciju u iteraciji n, gde je $x_{in} \in X$, $i = 1, \ldots, N$. x_{in} predstavlja i-tu jedinku u n-toj populaciji. Za svaku jedinku $x_{in} \in P_n$, određuje se vrednost funkcije pogodnosti $F : P_n \to \mathbb{R}$. Prilikom generisanja nove populacije, biraju se jedinke koje imaju bolju prilagođenost, na njih se primenjuju genetski operatori, čime se stvara nova populacija $P_{n+1} = \{x_{1,n+1}, \ldots, x_{N,n+1}\}$.

Selekcija predstavlja izbor jedinki iz trenutne populacije koje će biti korišćene za dobijanje naredne generacije. Najčešće vrste selekcije su *ruletska selekcija* i *turnirska selekcija*. Kod ruletske selekcije izbor jedinki koje će predstavljati roditelje vrši se na osnovu vrednosti funkcije pogodnosti jedinki, pri čemu se veća verovatnoća da budu izabrane daje jedinkama sa boljom prilagođenošću iz razloga što je veća verovatnoća da će roditelji koji imaju dobra svojstva dati potomke sa još boljim svojstvima od svakog pojedinačnog roditelja. Jedinke koje imaju lošiju prilagođenost polako izumiru iz generacije u generaciju. Naziv ove selekcije dolazi otud jer na krugu ruleta veći isečak pripada jedinkama sa boljom prilagođenošću. Kod turnirske selekcije se roditelji biraju tako što se na slučajan način bira nekoliko jedinki, nakon čega se organizuje turnir u kojem od izabranih jedinki pobeđuje ona koja je najbolje prilagođena. Nakon što se odaberu roditelji, potrebno je primeniti određene genetske operatore kako bi se od roditelja formirali potomci.

Ukrštanje predstavlja operator genetskog algoritma. Ono porazumeva kombinovanje gena dva roditelja, pri čemu se dobijaju dva nova potomka. Po ugledu na istoimeni prirodni proces, slučajno se razmenjuju delovi kodova dve jedinke i na taj način nastaju dva potomka. Rezultat ukrštanja jeste razmena genetskog materijala između jedinki pri čemu se povećava mogućnost da dobro prilagođene jedinke generišu još bolje prilagođene jedinke. U literaturi se najčešće javljaju *jednopoziciono, dvopoziciono* i *uniformno ukrštanje*.

Neka je l dužina koda jedinke, i neka se na slučajan način biraju geni i i j, gde je $0 \leq i < j \leq l$. Kod jednopozicionog ukrštanja, prvi potomak nasleđuje gene prvog roditelja do i-tog gena (uključujući i-ti gen), i gene drugog roditelja od i-tog gena, dok drugi potomak nasleđuje gene drugog roditelja do i-tog gena (uključujući i-ti gen) i gene prvog roditelja od i-tog gena. Kod dvopozicionog ukrštanja, prvi potomak nasleđuje gene prvog roditelja do i-tog gena (uključujući i-ti gen) i od j-tog gena (uključujući j-ti gen) i nasleđuje gene drugog roditelja od i-tog do j-tog gena, dok drugi potomak nasleđuje gene drugog roditelja do i-tog gena(uključujući

i-ti gen) i od j-tog gena(uključujući i-ti gen), a gene prvog roditelja od i-tog do j-tog gena. Najviše sličnosti sa prirodnim procesom ima uniformo ukrštanje gde se za svaki gen potomka uniformno bira roditelj, odnosno odabir od kog će roditelja potomak naslediti i-ti gen bira se na slučajan način. Mehanizmom ukrštanja dobro prilagođene jedinke imaju mogućnost da postanu još bolje, a lošije prilagođene jedinke sa nekim dobro prilagođenim genima imaju mogućnost da rekombinacijom dobrih gena proizvedu dobro rešenje.

Drugi operator genetskog algoritma jeste **operator mutacije** kojim se vrši promena nekih gena jedne jedinke sa nekom malom verovatnoćom p_{mut}. Svrha operatora mutacije jeste izbegavanje lokalne konvergencije. Ako je verovatnoća mutacije jako mala ili se izostavi proces mutiranja, može doći do prerane lokalne konvergencije, a ako je verovatnoća mutacije velika može se desiti da algoritam pretražuje prostor rešenja nasumično. Takođe, poželjno je koristiti mutaciju, jer je na ovaj način moguće vraćanje izgubljenog genetskog materijala u populaciju.

U nekim retkim slučajevima, operatorima ukrštanja i mutacije mogu se izgubiti dobre jedinke iz populacije, odnosno može se desiti da se u određenom trenutku dostigne optimalno rešenje, a da se ono potom izgubi primenom operatora ukrštanja i mutacije, i na taj način potomci postanu lošiji od roditelja. Kako bi se sprečilo gubljenje dobrih jedinki, često se određen broj najboljih jedinki direktno prebacuje u narednu generaciju, kako bi se zadržale najbolje jedinke u populaciji, bez rizika da se izgube njihove dobre karakteristike. Ovaj proces naziva se *elitizam*.

Radi formiranja raznovrsnog genetskog materijala, početna populacija se generiše na slučajan način, dok se svakoj jedinki dodeljuje funkcija prilagođenosti, odnosno funkcija pogodnosti koja određuje kvalitet date jedinke. Potom se primenjuju operatori selekcije, ukrštanja i mutacije i na taj način formira nova populacija, koja bi trebalo da bude bolja od prethodne. Osnovna struktura genetskog algoritma prikazana je algoritmom 1.

Algoritam 1 Genetski algoritam

```
Učitavanje parametara;
Generisanje početne populacije;
Određivanje funkcije prilagođenosti populacije;
Određivanje najboljeg rešenja;
while nije ispunjen kriterijum zaustavljanja do
    Postupak elitizma;
    Selekcija jedinki za primenu genetskih operatora;
    Ukrštanje za izabrane parove jedinki;
    Mutacija dobijenih jedinki;
    Određivanje funkcije prilagođenosti populacije;
    Ažuriranje najboljeg rešenja;
end while
return Najbolje rešenje;
```

4.2 Predložena GA implementacija za rešavanje CFLP-PDS

1. **Ulazni parametri**

   Parametri genetskog algoritma implementiranog u radu su: veličina populacije, broj elitnih jedinki, broj jedinki koje učestvuju u turnirskoj selekciji, verovatnoća mutacije jedinki i kriterijumi zaustavljanja - maksimalno vreme izvršavanja i maksimalan broj iteracija.

2. **Reprezentacija rešenja**

   U predloženoj implementaciji genetskog algoritma koristi se dvolinijska reprezentacija rešenja. Jedinka je predstavljena mešovitim vektorom, tj. kodirana je vektorom koji čine celi i realni brojevi. Jedinka je predstavljena vektorom dužine $n + m$, gde je n broj objekata, a m broj korisnika. Prvih n koordinata vektora čine celi brojevi iz segmenta $[0, k]$, koji prikazuju da li je objekat uspostavljen od strane snabdevača na lokaciji i, i ukoliko jeste kog je kapaciteta. Navedene vrednosti predstavljaju status lokacije. Narednih m koordinata vektora čine uniformno raspoređeni različiti realni brojevi iz segmenta $[0, 1]$ i oni određuju redosled pridruživanja korisnika objektima.

   Ukoliko je koordinata i, gde je $0 \leq i \leq n - 1$ jednaka nuli, tada na lokaciji i nije uspostavljen objekat od strane snabdevača, dok ukoliko je koordinata i

jednaka r, $r \in [1, k]$, tada je na lokaciji i uspostavljen objekat, sa statusom r, odnosno kapapaciteta lc_{ir-1}.

Što se tiče dela vektora koji se odnosi na korisnike, vrednost koja se nalazi na koordinati $n + j$ dodeljena je korisniku j. Ako je vrednost dodeljena korisniku j_1 manja od vrednosti dodeljene korisniku j_2, korisnik j_1 ima prvenstvo u odabiru objekta u kojem će zadovoljiti svoju potražnju. Sortiranjem korisnika po vrednosti koje su im dodeljene u reprezentaciji rešenja dobija se redosled pridruživanja korisnika snabdevačima, odnosno njihovim objektima.

U slučaju da je problem koji se razmatra neograničenih kapaciteta prvi deo vektora reprezentacije jedinke bio bi kodiran binarno. Ako je na poziciji i vrednost 1 objekat je uspostavljen, ako je na poziciji i vrednost 0 objekat nije uspostavljen. Imajući u vidu da je problem koji se razmatra ograničenih kapaciteta prvi deo vektora reprezentacije jedinke kodira se celobrojno, kako bi se pored informacije o tome da li je objekat uspostavljen ili ne, dobila i informacija kog je kapaciteta uspostavljeni objekat. Ovaj način reprezentacije rešenja predstavlja unapređenje u odnosu na radove u kojima se razmatraju lokacijski problemi ograničenih kapaciteta.

Kao što je već rečeno, za Model 1 prostor dopustivih rešenja čine jedinke kod kojih su svi korisnici zadovoljili svoju celu potražnju. Za Model 2 prostor dopustivih rešenja čine jedinke kod kojih su svi korisnici zadovoljili celu svoju potražnju kod lidera ili bar deo svoje potražnje kod sledbenika.

Primer

Neka je rešenje lidera za instancu dimenzije $m = n = 4$, $k = 2$ prikazane na slici 3.1 predstavljeno vektorom

$$L = (0, 1, 0, 2, 0.52, 0.34, 0.76, 0.61).$$

U navedenom primeru lider je uspostavio objekat na lokaciji $i = 1$ sa statusom $r = 1$, kapaciteta lc_{10} i objekat na lokaciji $i = 3$ sa statusom $r = 2$, kapaciteta lc_{31}. Drugi deo vektora sadrži realne brojeve uniformno raspoređene u segmentu $[0, 1]$. Najmanju vrednost ima korisnik $j = 1$ i to 0.34, što znači da on prvi bira kom objektu će se pridružiti. Analogno gledano, redosled dodeljivanja korisnika objektima lidera je dat sledećom listom $(1, 0, 3, 2)$. Ilustracija reprezentacije rešenja može se videti na slici 4.2.

Kod jedinke	0	1	0	2	0.52	0.34	0.76	0.61
	i = 0	i = 1	i = 2	i = 3	j = 0	j = 1	j = 2	j = 3

⇩

0.52	0.34	0.76	0.61
j = 0	j = 1	j = 2	j = 3

⇩

Sortiran vektor	0.34	0.52	0.61	0.76
Sortirani indeksi	j = 1	j = 0	j = 3	j = 2

Slika 4.2: Reprezentacija rešenja

3. **Evaluacija rešenja**

   Par vrednosti funkcije cilja lidera i vrednosti funkcije cilja sledbenika (L^*, F^*) smatra se merom kvaliteta rešenja. Imajući u vidu da oba snabdevača žele da maksimizuju svoj profit u razmatranom lokacijskom problemu postoje dve konfliktne funkcije cilja - funkcija cilja lidera i sledbenika.

   Prethodno navedena reprezentacija rešenja odnosi se na optimizacioni problem lidera i kodira rešenje lidera. Nakon formiranja dopustivog rešenja lidera informacije o uspostavljenim objektima lidera i alokaciji korisnika uspostavljenim objektima lidera X_{is} i x_{ij} prosleđuju se sledbenikovom optimizacionom problemu, koji se potom rešava standardnim linearnim rešavačem (CPLEX). Rezultat rešavanja sledbenikovog problema je vrednost funkcije cilja sledbenika i informacije na kojim lokacijama je sledbenik uspostavio objekte Z_{is}, kog kapaciteta $s \in S_i$ i koji korisnici su se vezali za koji objekat z_{ij}. Za računanje funkcije cilja lidera dovoljno je koristiti optimalnu vrednost z_{ij}. Pozivanje CPLEX-a u svakom koraku heurističke metode doprinosi složenosti algoritma.

   Primer

   Neka je dato rešenje iz prethodne tačke $L = (0, 1, 0, 2, 0.52, 0.34, 0.76, 0.61)$. Redosled dodeljivanja korisnika uspostavljenim objektima lidera dat je vektorom $(1, 0, 3, 2)$. Korisnik $j = 1$ ima prvenstvo da bira koji objekat mu najviše

odgovara za zadovoljenje svoje potražnje. Kao što je u radu već navedeno, korisnici biraju najbliži objekat koji zadovoljava njihovu potražnju. Korisniku $j = 1$ najbliži uspostavljen liderov objekat je $i = 1$. Potrebno je proveriti da li objekat zadovoljava potražnju kupca. Kapacitet navedenog objekta je $lc_{10} = 22$, a potražnja korisnika $j = 1$ je $w_1 = 9$, što znači da objekat $i = 1$ zadovoljava potražnju korisnika $j = 1$, i samim tim se vezuje za njega. Nakon pridruživanja potrebno je ažurirati kapacitet liderovog objekta tako što se od njega oduzme deo kapaciteta kojim je zadovoljena potražnja pridruženog kupca, tj. $lc_{10} = lc_{10} - w_1 = 22 - 9 = 13$. Sledeći kupac koji ima prioritet pri odabiru objekta u kojem će zadovoljiti svoju potražnju je $j = 0$. Najbliži objekat kupcu $j = 0$ je $i = 1$. Potražnja kupca $j = 0$ je $w_0 = 5$, a kapacitet objekta $i = 1$ je sada $lc_{10} = 13$. Stoga sledi da se kupac $j = 0$ vezuje za objekat $i = 1$. Ažurira se kapacitet objekta $lc_{10} = lc_{10} - w_0 = 13 - 5 = 8$. Na analogan način se određuje da se kupac $j = 3$ pridružuje objektu $i = 3$, a kupac $j = 2$ objektu $i = 1$.

Kada se ovo rešenje kroz promenljive X_{is} i x_{ij} prosledi sledbenikovom optimizacionom problemu, standardni rešavač pronalazi najbolje rešenje za sledbenika, a to je da sledbenik uspostavi objekat na lokaciji $i = 0$ kapaciteta fc_{00} i da se njemu pridruži korisnik $j = 0$. Standardni rešavač izračunava i vrednost funkcije cilja sledbenika i ona iznosi 5.00. Može se uočiti da je na ovaj način sledbenik preoteo lideru kupca $j = 0$, jer je uspostavio objekat koji zadovoljava potražnju kupca, a bliži je od liderovog objekta kojem je kupac prethodno bio pridružen. Funkcija cilja lidera predstavlja profit koji lider ostvaruje od korisnika koji su pridruženi objektima koje je uspostavio, od čega se oduzimaju troškovi uspostavljanja objekata i troškovi transporta i data je formulom

$$\sum_{j \in J} \Big(\sum_{i \in I^L} (p_{ij} - d_{Cij})\, x_{ij} \Big) \Big(1 - \sum_{i \in I^F} \tilde{z}_{ij} \Big) - \sum_{i \in I^L, s \in S_i} f_{is} X_{is}.$$

Kupac $j = 0$ nije vezan za lidera, kupci $j = 1$ i $j = 2$ su vezani za objekat $i = 1$, a kupac $j = 3$ je vezan za objekat $i = 3$. Troškovi uspostavljanja objekata $i = 1$ kapaciteta lc_{10} i objekta $i = 3$ kapaciteta lc_{31} su 23 i 30, respektivno. Prihod koji kupac $j = 1$ donosi lideru kada se veže za objekat $i = 1$ iznosi $p_{11} = 20$, a trošak transporta od kupca $j = 1$ do objekta $i = 1$ je 0. Prihod koji kupac $j = 2$ donosi lideru kada se veže za objekat $i = 1$ iznosi $p_{12} = 17$, a trošak transporta od kupca $j = 2$ do objekta $i = 1$ je 0.02. Prihod

koji kupac $j = 3$ donosi lideru kada se veže za objekat $i = 3$ iznosi $p_{33} = 13$, a trošak transporta od kupca $j = 3$ do objekta $i = 3$ je 0. Na sledeći način se izračunava vrednost funkcije cilja lidera:

$$\sum_{j\in J}(\sum_{i\in I^L}(p_{ij} - dc_{ij})\,x_{ij})\,(1 - \sum_{i\in I^F}\tilde{z}_{ij}) - \sum_{i\in I^L, s\in S_i} f_{is}X_{is} =$$

$$(p_{10} - dc_{10})(1 - \tilde{z}_{10}) + (p_{11} - dc_{11})(1 - \tilde{z}_{11}) +$$

$$(p_{12} - dc_{12})(1 - \tilde{z}_{12}) + (p_{33} - dc_{33})(1 - \tilde{z}_{33}) - f_{10} - f_{31} =$$

$$(20 - 0) + (17 - 0.02) + (13 - 0) - 23 - 30 = -3.02.$$

Sledi da je za rešenje lidera $L = (0, 1, 0, 2, 0.52, 0.34, 0.76, 0.61)$ optimalna vrednost funkcije cilja lidera $L = -3.02$, a optimalna vrednost funkcije cilja sledbenika $F = 5.00$.

4. **Inicijalizacija populacije**

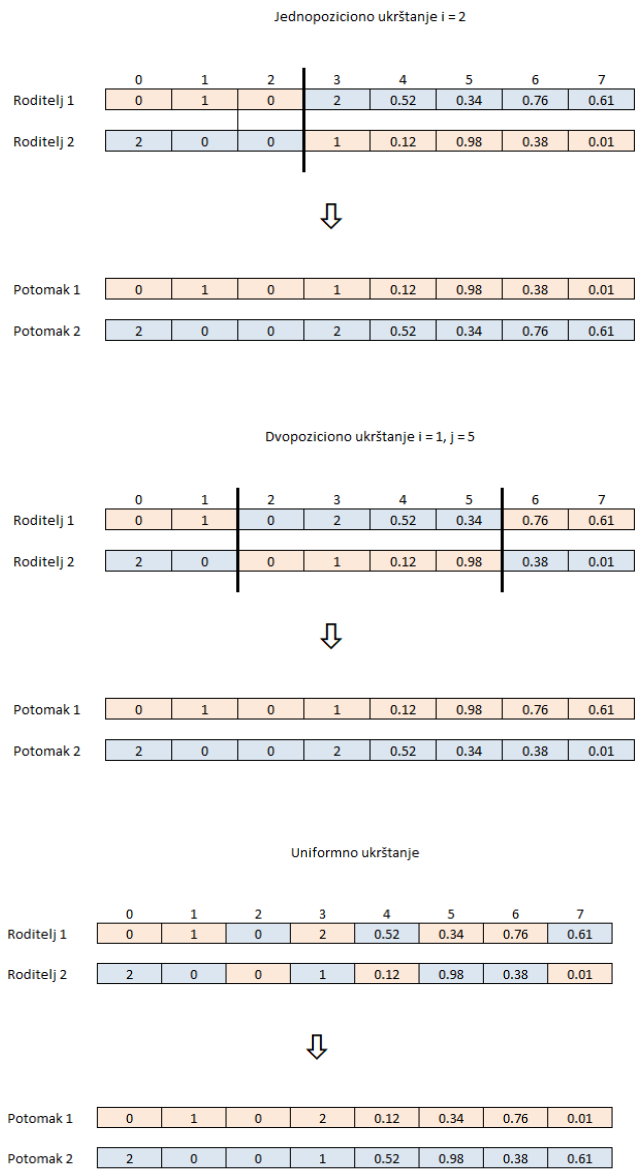
   Na slučajan način bira se $pop_{size} \in \mathbb{Z}$ dopustivih jedinki, gde je pop_{size} veličina populacije.

5. **Selekcija**

   Cilj genetskog algoritma je da svaka naredna generacija bude bolja od prethodne, tj. da potomci imaju bolju funkciju prilagođenosti od svojih roditelja. Da bi bili bolji od svojih roditelja potomci teže da preuzmu dobre gene od svakog roditelja. Iz tog razloga će bolji potomci nastati ako se ukrste dve dobro prilagođene jedinke. Međutim, ne treba zanemariti ni lošije prilagođene jedinke kako ne bi došlo do preuranjene lokalne konvergencije. Za odabir jedinki koje će učestvovati u reprodukciji korišćene su dve strategije. Od ukupnog broja jedinki iz naredne populacije 10% jedinki nastaje od roditelja koji su na slučajan način izabrani među 20% najbolje prilagođenih jedinki iz tekuće populacije. Ostali potomci nastaju od roditelja izabranih turnirskom selekcijom, tj. za roditelja se bira jedinka sa najboljom funkcijom prilagođenosti od t jedinki koje su na slučajan način izabrane iz tekuće populacije. Veličina turnira t predstavlja broj jedinki koji se biraju za takmičenje, i ona takođe predstavlja parametar algoritma. Na ovaj način postignuto je da se za roditelje biraju jedinke koje su dobro prilagođene, ali se i slabije prilagođenim jedinkama daje šansa da uđu u odabir pomoću turnirske selekcije, čime se izbegava preuranjena konvergencija u lokalni optimum.

6. **Operator ukrštanja**

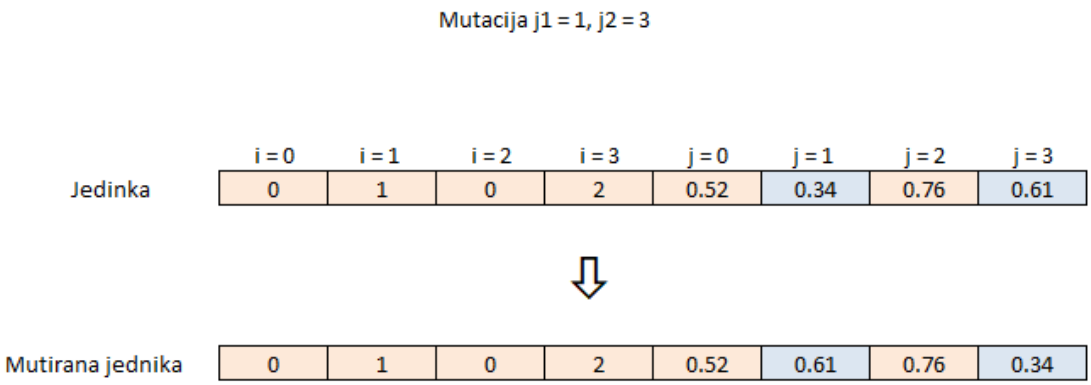
   Ukrštanje predstavlja kombinovanje gena dva roditelja, pri čemu se dobijaju dva nova potomka. U radu su korišćena tri tipa ukrštanja: jednopoziciono, dvopoziciono i uniformno. Za svaki par roditelja na slučajan način se (sa istom verovatnoćom) bira koji operator ukrštanja će se primeniti na roditelje. Ilustracija navedene tri metode ukrštanja može se videti na slici 4.3.



Slika 4.3: Operator ukrštanja

7. **Operator mutacije**

   Uobičajan pristup podrazumevao bi da se sa verovatnoćom p_{mut} na slučajan način izabere lokacija i i promeni joj se status. Međutim kako je problem CFLP-PDS lokacijsko-alokacijski problem pa od redosleda dodeljivanja korisnika objektima umnogome zavisi vrednost rešenja, kako u Modelu 1, tako i u Modelu 2, ovde je korišćen drugačiji pristup. Operatorom mutacije se na slučajan način biraju dva korisnika i međusobno im se menjaju vrednosti dodeljene u reprezentaciji rešenja. Na taj način menja se redosled dodeljivanja korisnika objektima. Operator mutacije se primenjuje na svaki potomak sa verovatnoćom p_{mut}. Verovatnoća mutacije predstavlja parametar algoritma. Ilustracija mutacije rešenja može se videti na slici 4.4.



Slika 4.4: Operator mutacije

8. **Kreiranje nove generacije**

   Iako je cilj genetskog algoritma da se pri reprodukciji stvore potomci koji imaju bolje karakteristike od svojih roditelja, može se desiti da se prilikom reprodukcije izgube jedinke koje su dobre ili pak najbolje. Da bi se sprečio takav scenario, u radu je primenjen princip elitizma, odnosno određen broj najboljih jedinki se automatski prenosi u narednu generaciju. Broj jedinki koji se na ovaj način prenosi u sledeću generaciju $elite_{size}$ predstavlja parametar algoritma. Naredne jedinke nove populacije nastaju primenom navedenih operatora ukrštanja i mutacije na tekuću populaciju. Prilikom reprodukcije na slučajan način se bira tip ukrštanja koji će se koristiti za formiranje potomaka, a potom se sa verovatnoćom p_{mut} na potomke primenjuje operator mutacije.

9. **Kriterijum zaustavljanja**

   Zbog velike složenosti algoritma, koriste se dva kriterijuma zaustavljanja: maksimalni broj iteracija i maksimalno vreme izvršavanja algoritma.

Glava 5

Metoda promenljivih okolina za rešavanje CFLP-PDS

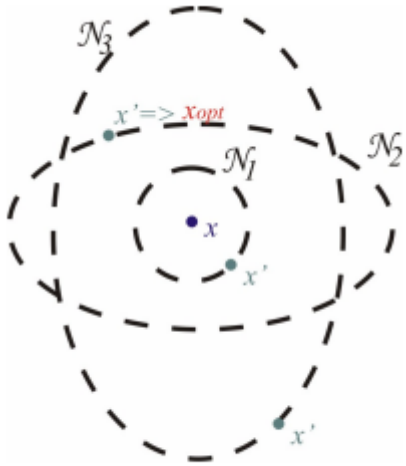
5.1 Osnovne postavke VNS metode i njenih varijanti

Metoda promenljivih okolina (VNS) predstavlja metaheuristiku zasnovanu na iterativnom poboljšanju jednog rešenja. Ona je zasnovana na lokalnom pretraživanju (LS) koje predstavlja najosnovniju verziju i najstariju S - metaheuristiku. Veliki nedostatak LS metaheuristike jeste u tome što algoritam često nalazi lokalni optimum, koji nije globalni. Usled svojih mana, retko se koristi samostalno, već u kombinaciji sa nekim drugim metaheuristikama. Metoda lokalnog pretraživanja se može prikazati algoritmom 2.

Algoritam 2 Lokalna pretraga

```
Generisati početno rešenje x;
Definisati okolinu N rešenja x
Izračunati vrednost funkcije cilja f ← f(x)
while nije ispunjen kriterijum zaustavljanja do
    Izabrati proizvoljno rešenje x' iz okoline rešenja x, x' ∈ N(x)
    if f(x') bolje od f(x) then
        x ← x'
        f ← f(x')
    end if
end while
Ispisati (x, f)
```

U cilju prevazilaženja navedenog nedostatka nastale su unapređene S - metaheuristike zasnovane na lokalnom pretraživanju, a jedna od njih je i metoda promenljivih okolina. VNS metoda bazirana je na iterativnom poboljšanju jednog rešenja x i pretraživanju t različitih okolina $N_t(x)$ tog rešenja, što se može videti na slici 5.1. Kao što joj samo ime kaže, osnovni koncept VNS metode jeste sistemska promena unapred definisanih okolina, koje se pretražuju sa ciljem izbegavanja zaglavljivanja algoritma u lokalni optimum.



Slika 5.1: Okoline

Metoda promenljivih okolina zasniva se na tri postulata:

1. Lokalni optimum u odnosu na jednu okolinu ne mora biti lokalni optimum u odnosu na drugu okolinu.

2. Globalni optimum je lokalni optimum u odnosu na sve okoline.

3. Za većinu problema lokalni optimumi u odnosu na različite okoline su relativno blizu.

VNS metodu predložili su Mladenović i Hansen u radu [25]. U međuvremenu su nastale razne determinističke, stohastičke i kombinovane varijante VNS metode, pri čemu je VNS metoda načelno nastala zbog potrebe rešavanja problema kombinatorne optimizacije, ali je kasnije proširila svoju primenu i na kontinualne optimizacione probleme.

Najčešće korišćene varijante metode promeljivih okolina u literaturi su Metoda promenljivog spusta (*eng. Variable Neighborhood Descent, VND*), Redukovana metoda promeljivih okolina (*eng. Reduced Variable Neighborhood Search, RVNS*), Osnovna metoda promenljivih okolina (*eng. Basic Variable Neighborhood Search, BVNS*) i Opšta metoda promenljivih okolina (*eng. General Variable Neighborhood Search, GVNS*). Neke od brojnih primena VNS metode na rešavanje lokacijskih problema mogu se naći u radovima [10], [38] i [29].

Metoda promenljivog spusta - VND

Koncept VND metode je da se bolje rešenje traži u više unapred definisanih okolina N_t tekućeg rešenja. Metodom lokalnog pretraživanja traži se najbolji sused rešenja x iz okoline N_t, $t \leq t_{max}$, gde je t_{max} broj definisanih okolina. Sve dok postoji bolje rešenje od tekućeg, pretraga se vrši u okolini N_t. Kada lokalna pretraga trenutne okoline ne dovodi do poboljšanja rešenja, algoritam nastavlja sa lokalnom pretragom u sledećoj okolini N_{t+1}. Broj okolina t_{max} po kojima se pretražuje prostor predstavlja parametar algoritma. Pseudokod VND metode prikazan je algoritmom 3.

Algoritam 3 Metoda promenljivog spusta

```
Definisati tipove okolina N_t, t ⩽ t_max;
Generisati početno rešenje x;
t ← 1;
while t ≤ t_max do
    x' = local_search(x, N_t);
    if f(x') < f(x) then
        x ← x';
        t ← 1;
    else
        t ← t + 1;
    end if
end while
Ispisati (x, f(x));
```

Redukovana metoda promenljivih okolina - RVNS

Koncept redukovane metode promenljivih okolina bazira se na uštedi vremena potrebnog za izvršavanje algoritma izbacivanjem korišćenja lokalne pretrage cele okoline. Naime, u VND metodi pomoću LS algoritma pretražuje se cela okolina tekućeg rešenja $N_t(x)$, pri čemu se dosta vremena gubi. Modifikacija VNS metode koja štedi vreme sastoji se u tome da se izbegava primena lokalne pretrage cele

okoline. Počevši od tekućeg rešenja na slučajan način se bira sused x' iz okoline N_t. Ako je x' bolje rešenje od x, ažurira se najbolje rešenje, i nastavlja se pretraga počev od prve okoline N_1. Ako x' nije bolje rešenje od x, pretraga se nastavlja u okolini N_{t+1}. Na ovaj način se štedi vreme, ali se i gubi na kvalitetu dobijenog rešenja. Pseudokod RVNS metode prikazan je algoritmom 4.

Algoritam 4 Redukovana metoda promenljivih okolina

```
Definisati tipove okolina N_t, t ⩽ t_max;
Generisati početno rešenje x;
while nije ispunjen kriterijum zaustavljanja do
    t ← 1;
    while t ≤ t_max do
        Na slučajan način izabrati x' ∈ N_t(x);
        if f(x') < f(x) then
            x ← x';
            t ← 1;
        else
            t ← t + 1;
        end if
    end while
end while
Ispisati (x, f(x));
```

Osnovna metoda promenljivih okolina - BVNS

Osnovna metoda promenljivih okolina sastoji se iz dve faze: **faze razmrdavanja** i **faze lokalne pretrage**. Najpre se definišu okoline koje će se koristiti u fazi razmrdavanja i u fazi lokalne pretrage. U svakoj iteraciji se naizmenično menjaju okoline u kojima se vrši pretraživanje $N_t, t \leq t_{max}$, a potom primenjuju faza razmrdavanja, faza lokalne pretrage i vrši provera kvaliteta tekućeg rešenja. U fazi razmrdavanja se na slučajan način izabere rešenje x' iz okoline N_t rešenja x, $x' \in N_t(x)$. U fazi lokalne pretrage se na x' primenjuje neka metoda lokalne pretrage unapred definisane okoline u cilju nalaženja boljeg rešenja. Neka je rezultat lokalne pretrage x''. Ako je x'' bolje rešenje od x, ažurira se najbolje rešenje i pretraga se ponovo nastavlja u okolini N_1, tj. t se postavlja na 1. Ako x'' nije bolje rešenje od x, t se postavlja na $t + 1$ i ponovo se primenjuje faza razmrdavanja u kojoj se koristi okolina N_{t+1}, a potom i lokalna pretraga. Faza razmrdavanja ima ulogu da spreči konvergenciju metode ka lokalnom optimumu koji nije globalni, dok faza lokalne pretrage ima ulogu da smanji nepotrebno širenje pretraživačke oblasti. Postoje varijante metode promenljivih okolina u kojima figuriše samo jedna od ovih

faza. Važno je napomenuti da okoline koje se primenjuju u fazi razmrdavanja i fazi pretrage ne moraju biti iste. Same okoline se mogu razlikovati na osnovu rastojanja (broja transformacija nad rešenjem) i na osnovu metrike (vrste transformacije nad rešenjem). BVNS metoda je najčešće korišćena metoda VNS-a. Pseudokod osnovne metode promenljivih okolina prikazan je algoritmom 5.

Algoritam 5 Osnovna metoda promenljivih okolina

```
Definisati tipove okolina N_t, t ⩽ t_max;
Generisati početno rešenje x;
while nije ispunjen kriterijum zaustavljanja do
    t ← 1;
    while t ⩽ t_max do
        \\Faza razmrdavanja
        Izabrati proizvoljno rešenje x' iz okoline N_t(x);
        \\Faza lokalne pretrage
        x'' ← local_search(x');
        if f(x'') < f(x) then
            x ← x'';
            t ← 1;
        else
            t ← t + 1;
        end if
    end while
end while
Ispisati vrednost (x, f(x));
```

Uopštena metoda promenljivih okolina - GVNS

Uopštena metoda promenljivih okolina predstavlja modifikaciju osnovne metode promenljivih okolina. Ako se umesto faze lokalne pretrage u BVNS primeni metoda promenljivog spusta dobija se GVNS metoda. Skup okolina koje se koriste u VND metodi može biti isti kao skup okolina koji se koriste u fazi razmrdavanja, ali i ne mora. Od prirode problema zavisi da li će algoritam biti efikasniji ako se u fazi ramrdavanja i u VND metodi koriste isti tipovi okolina ili ne.

Pseudokod uopštene metode promenljivih okolina prikazan je algoritmom 6.

Algoritam 6 Uopštena metoda promenljivih okolina

```
Definisati tipove okolina N_t, t ⩽ t_max za fazu razmrdavanja;
Definisati tipove okolina M_l, l ⩽ l_max za VND;
Generisati početno rešenje x;
while nije ispunjen kriterijum zaustavljanja do
    t ← 1;
    while t ⩽ t_max do
        \\Faza razmrdavanja;
        Izabrati proizvoljno rešenje x' iz okoline N_t(x);
        \\Faza VND;
        l ← 1;
        while l ≤ l_max do
            x'' = local_search(x', M_l);
            if f(x'') < f(x') then
                x' ← x'';
                l ← 1;
            else
                l ← l + 1;
            end if
        end while
        if f(x'') < f(x) then
            x ← x'';
            t ← 1;
        else
            t ← t + 1;
        end if
    end while
end while
Ispisati vrednost (x, f(x));
```

5.2 Predložena BVNS implementacija za rešavanje CFLP-PDS

1. **Ulazni parametri**

   Ulazni parametri BVNS algoritma implementiranog u radu su: broj okolina za pretraživanje i kriterijumi zaustavljanja - maksimalno vreme izvršavanja i maksimalan broj iteracija (za BVNS algoritam i za LS algoritam implementiran u fazi lokalnog pretraživanja BVNS alogoritma). Parametar t_{max} označava broj okolina koje se koriste u fazi razmrdavanja. Ovaj parametar postavljen je

na $[n/3]+1$, gde je n broj lokacija na kojima se mogu uspostaviti objekti, čime se postiže da sused na približno trećini lokacija zameni status. Povećavanjem ovog parametra okolina rešenja teži da postane čitav pretraživački prostor, čime se gubi svrha algoritma, dok se smanjenjem parametra t_{max} sužava okolina i povećava opasnost da se algoritam zaglavi u lokalnom optimumu koji nije globalni.

2. **Reprezentacija rešenja**

   Rešenje u BVNS algoritmu kodirano je na isti način kao jedna jedinka populacije u implementaciji genetskog algoritma.

3. **Evaluacija rešenja**

   Imajući u vidu da je reprezentacija rešenja u BVNS algoritmu ista kao u predloženom genetskog algoritmu, vrednost funkcije cilja lidera i sledbenika računa se na isti način kao u genetskom algoritmu, koji je izložen ranije u radu.

4. **Generisanje početnog rešenja**

   Početno rešenje generisano je na slučajan način iz dopustivog prostora rešenja. Za prvih n koordinata biraju se celi brojevi iz segmenta $[0, k]$ na slučajan način, za drugih m koordinata biraju se različiti realni brojevi iz segmenta $[0, 1]$, takođe na slučajan način. Kako bi algoritam imao informaciju koji korisnici su se vezali za lidera, a koji za sledbenika, u ovoj fazi je potrebno standardnom rešavaču (u ovom radu korišćen je CPLEX) proslediti početno rešenje lidera kako bi on pronašao najbolje rešenje sledbenika za tako definisano rešenje lidera. Proces se ponavlja sve dok se ne dobije dopustivo rešenje. Naime, dopustivo rešenje je ono rešenje kod koga važi samo to da su zadovoljeni svi korisnici, bez obzira na profit koji ostvaruju igrači (igrač može da bude na gubitku). Zbog toga se retko dešava da se kao početno rešenje definiše nedopustivo rešenje, odnosno retko postoji potreba za generisanjem novog rešenja koje će predstavljati početno.

5. **Faza razmrdavanja**

   Neka je x tekuće rešenje, a N_t tekuća okolina u fazi razmrdavanja. Sused rešenja x iz okoline N_t jeste rešenje x' kojem je na t lokacija zamenjen status lokacije u odnosu na rešenje x. Faza razmrdavanja se odnosi na lociranje i

dealociranje objekata, pri čemu redosled dodeljivanja korisnika ostaje nepromenjen. Naime, na slučajan način se bira t lokacija, te im se potom menja status. Kao što je već rečeno, status lokacije jeste ceo broj iz segmenta $[0, k]$. Ako je status lokacije i jednak 0 tada na navedenoj lokaciji nije uspostavljen objekat, dok ako je status lokacije i jednak r, $r \in [1, k]$, tada je na navedenoj lokaciji uspostavljen objekat kapaciteta lc_{ir-1}.

Menjanjem statusa se uspostavlja ili uklanja objekat sa lokacije ili se menja kapacitet već uspostavljenog objekta. Ako na lokaciji i nije uspostavljen objekat, menjanje tekućeg statusa (koji je jednak nuli) statusom $r \in [1, k]$, koji se bira na slučajan način, podrazumeva uspostavljanje objekta kapaciteta lc_{ir-1}. Ako je na lokaciji i uspostavljen objekat sa statusom $r \in [1, k]$ menjanjem postojećeg statusa lokacije statusom r^*, $r^* \in [1, k] \backslash \{r\}$, koji se bira na slučajan način, može se promeniti kapacitet uspostavljenog objekta na lc_{ir^*-1} ili se može ukloniti objekat sa te lokacije postavljanjem statusa r na nulu.

Primer

Neka je $m = 4, n = 4, k = 2$ i $t = 2$ i rešenje $x = (0, 1, 0, 2, 0.52, 0.34, 0.76, 0.61)$. S obzirom da je $t = 2$ na slučajan način biraju se dva cela broja $i_1 \neq i_2$ iz segmenta $[0, n-1]$, tj. $[0, 3]$. Neka je, na primer, $i_1 = 1$ i $i_2 = 2$. Status lokacije i_1 je $r_1 = 1$, što znači da je na lokaciji $i = 1$ uspostavljen objekat kapaciteta lc_{10}. Na slučajan način se bira novi status $r_1^* \in [0, k] \backslash \{r_1\}$. Neka je na slučajan način izabrano $r_1^* = 2$. To znači da će se na lokaciji i_1 status lokacije promeniti sa 1 na 0, tj. kapacitet već uspostavljenog objekta će promeniti sa lc_{i0} na lc_{i1}. Status lokacije i_2 je $r_2 = 0$, što znači da na lokaciji 2 nije uspostavljen objekat. Na slučajan način se bira novi status $r_2^* \in [0, k] \backslash \{r_2\}$. Neka je na slučajan način izabrano $r_2^* = 1$. To znači da će se na lokaciji i_2 uspostaviti objekat kapaciteta $lc_{i_2 0}$. Najzad sused x' je $(0, 2, 1, 2, 0.52, 0.34, 0.76, 0.61)$. Sused u fazi razmrdavanja menja statuse objekatima, pri čemu redosled raspoređivanja korisnika objektima ostaje isti.

6. **Lokalna pretraga**

U fazi razmrdavanja okolina se definiše tako da pravi „grublje", krupnije promene u rešenju, kako bi se raširio prostor pretrage i na taj način izbeglo zaglavljivanje u lokalnom optimumu koji nije globalni. U fazi lokalne pretrage okolina se definiše tako da pravi finije promene u rešenju sa ciljem intenzifikacije rešenja. S tim u vezi okolina definisana u lokalnoj pretrazi definisana

je na sledeći način. Neka je x' tekuće rešenje i M okolina rešenja x'. Susedi rešenja x' jesu rešenja kojima statusi lokacija ostaju nepromenjeni, ali se menja redosled dodeljivanja klijentima. Na slučajan način biraju se dva korisnika j_1 i j_2, gde je $j_1 \neq j_2$, kojima se međusobno zamenjuje redosled dodeljivanja korisnika objektima.

Od redosleda dodeljivanja korisnika objektima značajno zavisi vrednost funkcije cilja optimalnog rešenja. Korisnici imaju pravo da biraju između objekata lidera i sledbenika i kao što je navedeno ranije u radu, dodeljuju se najbližem objektu koji ima dovoljan kapacitet da zadovolji celu (Model 1) ili delimičnu potražnju korisnika (Model 2).

Od redosleda dodeljivanja korisnika objektima zavisi sama vrednost funkcije cilja lidera i sledbenika. Drugim rečima, kada lider i sledbenik odaberu lokacije na kojima će uspostaviti svoje objekte, korisnici biraju u kojim objektima će zadovoljiti svoju potražnju, a sam redosled kojim korisnici biraju navedene objekte utiče na vrednost funkcije cilja optimizacionih problema lidera i sledbenika. Iz tog razloga na izloženim modelima se ne može primeniti reprezentacija rešenja u kojem će figurisati samo lokacije za uspostavljanje objekata, već je neophodno da u reprezentaciji rešenja figuriše i redosled pridruživanja korisnika objektima.

Primer

Neka je $m = 4, n = 4$ i rešenje $x' = (0, 2, 1, 2, 0.52, 0.34, 0.76, 0.61)$. Redosled dodeljivanja korisnika objektima dat je sledećom listom $(1, 0, 3, 2)$. Da bi se kreirao sused iz okoline $M(x')$ na slučajan način se biraju dva cela broja $j_1 \neq j_2$ iz segmenta $[0, m-1]$, tj. segmenta $[0, 3]$. Neka je na slučajan način izabrano $j_1 = 1$ i $j_2 = 3$. Vrednosti dodeljene korisnicima j_1 i j_2 međusobno se zamenjuju, pri čemu se dobija sused x'' je $(0, 2, 1, 2, 0.52, 0.61, 0.76, 0.34)$. U ovom rešenju redosled dodeljivanja korisnika objektima dat je sledećom listom $(3, 0, 1, 2)$.

7. **Kriterijum zaustavljanja**

   Kriterijumi zaustavljanja su maksimalno vreme izvršavanja i maksimalan broj iteracija BVNS algoritma.

Glava 6

Memetski algoritam za rešavanje CFLP-PDS

6.1 Osnovne postavke MA

Heuristike zasnovane na iterativnom poboljšanju jednog rešenja imaju izvesne vrline i mane, baš kao i populacijske metaheuristike. S-heuristike imaju osobinu da brzo konvergiraju, ali se često dešava da je taj optimum lokalnog, a ne globalnog karaktera. Populacijske metaheuristike imaju dobre mehanizme da izađu iz lokalnog optimuma, ali su znatno sporije. Takođe, porastom dimenzije problema raste i vreme izvršavanja koje može postati nedopustivo veliko. U cilju prevazilaženja nedostataka ovih metoda dolazi se do njihove hibridizacije. Hibridni algoritmi kombinuju dve heuristike, na način svojstven problemu koji se razmatra. Uglavnom se kobinuju jedna P-heuristika i jedna S-heuristika. Populacijska heuristika služi da diverzifikuje pretraživački prostor, a pomoću S-heuristike ubrzava se konvergencija i detaljnije se pretražuje prostor u okolini potencijalno dobrog rešenja čime se izbegava slučaj da se algoritam približi jako blizu globalnog optimuma, ali ipak ne stigne do njega.

Jedan od mnoštva hibridnih algoritama jeste memetski algoritam (MA) koji hibridizuje jednu populacijsku heuristiku i neku metodu lokalne pretrage. Srž memetskog algoritma jeste mema, koja predstavlja jedinicu prenosa kulture. Termin „memetski" izveo je Čarls Darvin u teoriji o memama, kao kulturološkim i sociološkim ekvivalentima gena. Naime, meme se mogu ukrštati i menjati u okviru populacije. Na taj način kultura evoluira i teži napretku. Kao prirodan proces, dobre ideje (meme) teže da opstanu i prošire se u okruženju, dok loše ideje okruženje ne prepoznaje i one polako izumiru. Više o memetskim algoritmima i njihovim primenama

može se pročitati u radovima [11], [26], [8], [32] i [33].

Memetski algoritam se u poslednje dve decenije koristi za rešavanje brojnih problema optimizacije. Neke od oblasti u kojima se primenjuju memetski algoritmi su mašinsko učenje, planiranje, problemi raspoređivanja, lokacijslo planiranje, bioinformatika, elektronika, telekomunikacije i slično. Najčešća primena memetskih algoritama hibridizuje genetski algoritam i neku od metoda lokalnih pretraga, stoga se vremenom za memetski algoritam i podrazumeva da je on hibridizacija konkretno genetskog algoritma i neke verzije lokalne pretrage.

Posebna pažnja pridaje se odabiru jedinki na koje će se primeniti metoda lokalne pretrage. Rešenja na koje se primenjuje LS prolaze kroz proces usavršavanja što odgovara paradigmi učenja tokom celog života. Moguće je u svakoj iteraciji GA na svaku jedinku primeniti lokalnu pretragu, čime bi se dobila populacija bogata lokalnim optimumima, međutim na taj način se dosta povećava vreme izvršavanja algoritma, a GA će biti potrebna jaka mutacija kako bi izašao iz navedenih lokalnih optimuma i dostigao globalni. Najčešća metoda je da se samo na najbolju jedinku iz populacije primeni metoda lokalne pretrage i to samo u slučaju da se ta jedinka promenila u poslednjoj iteraciji genetskog algoritma, čime se izbegava primena lokalne pretrage na isto rešenje više puta. Drugi predlog je da se na slučajan način izabere k jedinki na koje će se primeniti lokalna pretraga. Treći predlog je da se populacija podeli na k nivoa, a da se potom iz svakog nivoa izabere jedinka sa najboljom funkcijom pogodnosti i da se na nju primeni metoda lokalne pretrage.

U radu je predložen memetski algoritam koji hibridizuje GA i BVNS metodu. Algoritmom 7 je prikazana struktura predloženog algoritma.

Algoritam 7 Memetski algoritam GA + BVNS

```
Učitavanje parametara;
Generisanje početne populacije;
Određivanje funkcije prilagođenosti populacije;
Određivanje najboljeg rešenja;
while nije ispunjen kriterijum zaustavljanja do
    Postupak elitizma;
    Selekcija jedinki za primenu genetskih operatora;
    Ukrštanje za izabrane parove jedinki;
    Mutacija izabranih jedinki;
    Primena BVNS algoritma na određene jedinke iz populacije;
    Određivanje funkcije prilagođenosti populacije;
    Ažuriranje najboljeg rešenja;
end while
return Najbolje rešenje;
```

6.2 Predložena implementacija MA za rešavanje CFLP-PDS

U implementaciji memetskog algoritma korišćen je genetski algoritam predložen u sekciji 4.2 ovog rada i osnovna metoda promenljivih okolina predložena u sekciji 5.2 ovog rada.

Početna ideja je da se BVNS algoritam primeni na sledeći način. Populacija se sortira opadajuće u odnosu na vrednost funkcije cilja. Sortiranu populaciju podeliti na b slojeva, a potom BVNS metodu primeniti na prvu jedinku iz svakog sloja. Prva jedinka će ujedno biti i jedinka sa najboljom funkcijom pogodnosti iz svoje grupe, jer je populacija sortirana. Međutim, ovakav algoritam odnosi mnogo vremena, posebno kada su u pitanju instance velike dimenzije. Iz navedenog razloga u radu je implementiran sledeći metod. U svakoj iteraciji genetskog algoritma, BVNS se primenjuje samo na najbolju jedinku iz populacije, pod uslovom da se najbolja jedinka promenila u odnosu na prethodni korak GA.

Prilikom predložene hibridizacije GA i BVNS algoritma, vreme izvršavanja BVNS algoritma se ograničava kako ne bi došlo do značajnog povećanja vremena izvršavanja celokupnog algoritma.

Glava 7

Eksperimentalni rezultati i analiza

Implementacija predloženih heuristika implementirana je u programskom jeziku *C++*. Sva testiranja izvršena su na računaru sa 64-bitnim procesorom *Intel(R) Core(TM) i5-3337U CPU @ 1.80 GHz 1.80 GHz* i 8.00 *GB* radne memorije (RAM memorije). Za rešavanje optimizacionog problema sledbenika korišćen je egzaktni rešavač *IBM ILOG CPLEX Optimization Studio (CPLEX) V12.8.0*.

Za potrebe testiranja implementiranih heurstika korišćene su instance malih, srednjih i velikih dimenzija, koje su javno dostupne na adresi https://doi.org/10.6084/m9.figshare.5548594. Instance sadrže redom, broj potencijalnih lokacija $|I|$, broj korisnika $|J|$, broj različitih kapaciteta koji se mogu uspostaviti na određenoj lokaciji k, p_{ij}, q_{ij}, w_j, f_{is}, g_{is}, $l_{C_{is}}$, $f_{C_{is}}$ i d_{ij}. Navedene instance generisane su na slučajan način pod uslovima koji su prikazani u tabeli 7.1, gde U predstavlja uniformnu raspodelu. Rastojanja između korisnika i objekata d_{ij} dobijena su iz cirkularne mreže sa dva prstena i 25 čvorova iz [3]. Prilikom formiranja instanci korišćene su pretpostavke da je skup potencijalnih lokacija za uspostavljanje objekata i skup lokacija korisnika isti, tj. $I = J$, zatim i da je skup lokacija na kojima lider može uspostaviti svoje objekte isti kao skup lokacija na kojima sledbenik može uspostaviti svoje objekte, tj. $I = I^F = I^L$. Takođe, korišćena je pretpostavka da se na svakoj lokaciji može uspostaviti objekat dva različita kapaciteta, tj. $k = 2$.

Tabela 7.1: Vrednosti parametara za generisanje instanci

Vrednosti parametara
$p_{ij} = U(12, 24)$
$q_{ij} = p_{ij}$
$w_j = U(5, 10)$
$f_{i1} = U(20, 25)$
$f_{i2} = U(30, 35)$
$g_{i1} = U(15, 20)$
$g_{i2} = U(25, 30)$
$l_{C_{i1}} = U(20, 25)$
$l_{C_{i2}} = U(30, 35)$
$f_{C_{i1}} = U(7, 12)$
$f_{C_{i2}} = U(15, 20)$
$d_{ij} \leftarrow$ *Class R2d from Beresnev(2016)*
$d_{C_{ij}} = 0.01 * d_{ij}$
$S_i = \{1, 2\}$

Na test primeru dimenzije $|I| = |J| = 4$, tj. $m = n = 4$ i $k = 2$, ulazni podaci u odgovarajućem formatu prikazani su na slici 7.1. U prvom redu nalaze se brojevi koji označavaju dimenziju problema, redom n, m i k. Nakon toga sledi matrica prihoda lidera $[p_{ij}]_{i,j=\{1,\ldots,n\}}$ dimenzije mxn, a potom matrica prihoda sledbenika $[q_{ij}]_{i,j=\{1,\ldots,n\}}$ dimenzije mxn. U narednom redu nalazi se vektor dimenzije m koji sadrži podatke o potražnji kupaca w_j. Zatim slede matrice dimenzije nxk koje sadrže podatke o fiksnim troškovima potrebnim za uspostavljanje objekata lidera i sledbenika $[f_{is}]_{i=\{1,\ldots,n\},s=\{1,\ldots,k\}}$, $[g_{is}]_{i=\{1,\ldots,n\},s=\{1,\ldots,k\}}$, a potom matrice dimenzije nxk koje sadrže podatke o maksimalnoj količini potražnje koja može biti zadovoljena uspostavljanjem objekata lidera ili sledbenika na određenoj lokaciji $[lc_{is}]_{i=\{1,\ldots,n\},s=\{1,\ldots,k\}}$ i $[fc_{is}]_{i=\{1,\ldots,n\},s=\{1,\ldots,k\}}$. Na kraju nalazi se matrica $[d_{ij}]_{i,j=\{1,\ldots,n\}}$ dimenzije nxm koja predstavlja rastojanja korisnika od objekata. Na osnovu ulaznih podataka, računa se dc_{ij} po formuli $dc_{ij} = 0.01 \cdot d_{ij}$.

```
4 4 2

22      20      17      13
22      20      17      13
22      20      17      13
22      20      17      13

22      20      17      13
22      20      17      13
22      20      17      13
22      20      17      13

5       9       5       8

21      33
23      35
25      35
21      30

17      30
17      26
19      29
20      29

25      30
22      30
22      31
21      34

9       20
11      17
11      16
12      15

0       4       6       5
4       0       2       6
6       2       0       4
5       6       4       0
```

Slika 7.1: Ulazni podaci test instance dimenzije |I|=|J|=4, k=2

Razmatrani problem moguće je rešiti egzaktnim metodama, ali samo za instance male dimenzije, imajući u vidu složenost samog problema. Za instance srednjih i velikih dimenzija, egzaktni rešavač radi nedopustivo dugo. Navedena zapažanja preuzeta su iz rada [35]. Broj promenljivih u modelima 1 i 2 prikazan je u tabeli 7.2. Na ovim instancama se jasno može uočiti značaj heurističkih metoda za rešavanje razmatranog problema. Kao što je već navedeno, optimizacioni problem lidera rešava se heurističkom metodom u kojoj je ugnježden optimizacioni problem sledbe-

nika, koji se rešava standardnim rešavačem CPLEX. Maksimalno vreme izvršavanja optimizacionog problema sledbenika u CPLEX-u ograničeno je na 120 sekundi.

Tabela 7.2: Broj promenljivih Modela 1 i 2

		Lider		Sledbenik			Ukupno
Instanca	$\|I\| = \|J\|$	X_{is}	x_{ij}	Z_{is}	z_{ij}	γ_{ij}	Model1/Model2
small1	4	8	16	8	16	16	48/64
small2	5	10	25	10	25	25	70/95
small3	6	12	36	12	36	36	96/132
medium1	8	16	64	16	64	64	160/224
medium2	10	20	100	20	100	100	240/340
medium3	12	24	144	24	144	144	336/480
large1	16	32	256	32	256	256	576/832
large2	20	40	400	40	400	400	880/1280
large3	25	50	625	50	625	625	1350/1975

Način prezentovanja eksperimentalnih rezultata

Na svakoj instanci male i srednje dimezije implementirana heuristika je pokrenuta 10 puta, dok je za instance velike dimenzije heuristika pokrenuta 5 puta. Neka se metaheuristika izvršava k puta na instanci. Pri svakom izvršavanju pamti se najbolje rešenje koje je postigla sol_i, vreme za koje metaheuristika prvi put dobila najbolje rešenje t_i u sekundama i ukupno vreme i-tog izvršavanja metaheuristike t_tot_i u sekundama, gde je $i = 1, \ldots, k$. Po završetku svih k izvršavanja, odredi se najbolje rešenje lidera dobijeno kroz tih k izvršavanja i označi se sa $Best_{sol}$. S obzirom da je razmatrajući problem optimizacioni problem maksimizacije važi $Best_{sol} = \max_{i=1,\ldots,k}(sol_i)$. Za svako rešenje sol_i izračunato je procentualno odstupanje od najboljeg rešenja dobijenog implementiranom heuristikom po formuli

$$gap_i = 100 \cdot \frac{|sol_i - Best_{sol}|}{|Best_{sol}|}.$$

Prosečno vreme za koje je heuristika prvi put dostigla najbolje rešenje t_{best}, prosečno ukupno vreme izvršavanja heurističke metode t_{tot}, srednje odstupanje $agap$ (u procentima) i standardna devijacija σ (u procentima) računaju se po sledećim formulama:

$$t_{best} = \frac{1}{k}\sum_{i=1}^{k} t_i,$$

$$t_{tot} = \frac{1}{k}\sum_{i=1}^{k} t\_tot_i,$$

$$agap = \frac{1}{k}\sum_{i=1}^{k} gap_i,$$

$$\sigma = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(gap_i - agap)^2}.$$

7.1 Eksperimentalni rezultati dobijeni GA metodom

Parametri implementiranog genetskog algoritma određeni su eksperimentalnim putem. Izbor parametara *pop_size*, *tour_size*, *elite_size*, *mut_prob*, *max_iter* i *max_time* za instance male srednje i velike dimenzije prikazan je u tabeli 7.3.

Tabela 7.3: Vrednost parametara genetskog algoritma

Parametar	small	medium	large
pop_size	50	100	250
tour_size	5	7	20
elite_size	5	7	20
mut_prob	0.5	0.5	0.5
max_iter	100	500	1000
max_time	600s	1200s	1800s

U ovoj sekciji, kao i u narednim sekcijana prikazane su tabele sa eksperimentalnim rezultatima, koje sadrže redom: redni broj instance, naziv instance, dimenziju instance, najbolje rešenje dobijeno heuristikom (L, F), prosečno vreme za koje je heuristika prvi put dostigla najbolje rešenje t_{best} (u sekundama), prosečno ukupno vreme izvršavanja heurističke metode t_{tot} (u sekundama), prosečno odstupanje od najbolje dobijenog rešenja *agap* (u procentima) i standardnu devijaciju σ (u procentima). Iako se rešenja porede po vrednosti funkcije cilja lidera (s obzirom da je lider dominantan u igri) u tabelama je prikazan uređen par (L, F), sa ciljem boljeg sagledavanja i analiziranja rezultata. Rezultati GA metode na instancama malih, srednjih i velikih dimenzija prikazani su u tabelama 7.4 i 7.5 za Model 1 i za Model 2, respektivno.

Tabela 7.4: Rezultati GA za Model 1

Genetski algoritam - Model 1							
R.br.	Instanca	|I|=|J|	(L,F)	t_best (s)	t_tot (s)	agap(%)	σ(%)
1.	small1	4	(13.95, 10.98)	0.38	17.94	0.00	0.00
2.	small2	5	(33.94, 2.00)	0.56	9.41	0.00	0.00
3.	small3	6	(25.95, 15.95)	3.85	16.83	0.40	1.20
4.	medium1	8	(42.94, 0.00)	2.09	70.52	0.00	0.00
5.	medium2	10	(58.80, 0.00)	5.88	309.59	6.24	7.22
6.	medium3	12	(58.85, 9.00)	17.62	233.77	3.57	5.61
7.	large1	16	(73.76, 31.91)	24.85	1800.00	6.97	4.75
8.	large2	20	(127.72, 18.91)	16.62	1800.00	6.07	3.57
9.	large3	25	(137.72, 0.00)	26.62	1800.00	10.16	7.90

Tabela 7.5: Rezultati GA za Model 2

Genetski algoritam - Model 2							
R.br.	Instanca	|I|=|J|	(L,F)	t_best(s)	t_tot(s)	agap(%)	σ(%)
1.	small1	4	(20.60, 13.31)	0.60	23.55	0.00	0.00
2.	small2	5	(33.94, 2.00)	0.59	13.84	0.00	0.00
3.	small3	6	(27.48, 16.34)	8.77	28.51	0.00	0.00
4.	medium1	8	(42.94, 0.00)	3.71	154.73	0.23	0.70
5.	medium2	10	(31.69, 14.15)	27.32	340.29	3.67	6.54
6.	medium3	12	(40.89, 18.98)	71.41	307.55	13.49	11.58
7.	large1	16	(58.82, 25.95)	109.82	1800.00	5.48	3.60
8.	large2	20	(114.19, 21.57)	37.51	1800.00	5.25	5.19
9.	large3	25	(95.42, 39.29)	119.84	1800.00	16.47	9.21

7.2 Eksperimentalni rezultati dobijeni BVNS metodom

Parametri implementirane heuristike osnovne metode promenljivih okolina određeni su eksperimentalnim putem. Izbor parametara k_{max}, max_iter i max_iter_ls za instance male, srednje i velike dimenzije prikazan je u tabeli 7.6. Rezultati BVNS metode na instancama malih, srednjih i velikih dimenzija prikazani su u tabelama 7.7 i 7.8 za Model 1 i za Model 2, respektivno.

Tabela 7.6: Vrednost parametara BVNS algoritma

Parametar	small	medium	large
t_{max}	$[\|I\|/3]+1$	$[\|I\|/3]+1$	$[\|I\|/3]+1$
max_iter	100	500	1000
max_iter_ls	70	500	100
max_time	600s	1200s	1800s

Tabela 7.7: Rezultati BVNS algoritma za Model 1

BVNS algoritam - Model 1							
R.br.	Instanca	|I| = |J|	(L,F)	t_best(s)	t_tot(s)	agap(%)	σ(%)
1.	small1	4	(13.95, 10.98)	2.70	54.85	0.00	0.00
2.	small2	5	(33.94, 2.00)	4.84	62.54	0.00	0.00
3.	small3	6	(25.95, 15.95)	7.34	145.07	0.00	0.00
4.	medium1	8	(42.94, 0.00)	45.63	378.28	0.00	0.00
5.	medium2	10	(62.80, 4.89)	455.42	1200.00	0.00	0.01
6.	medium3	12	(58.85, 9.00)	155.32	1200.00	0.00	0.00
7.	large1	16	(73.76, 31.91)	907.97	1800.00	3.47	4.78
8.	large2	20	(130.71, 7.91)	866.14	1800.00	2.28	3.04
9.	large3	25	(133.74, 20.89)	945.99	1800.00	12.11	6.17

Tabela 7.8: Rezultati BVNS algoritma za Model 2

BVNS algoritam - Model 2							
R.br.	Instanca	|I|=|J|	(L,F)	t_best(s)	t_tot(s)	agap(%)	σ(%)
1.	small1	4	(20.60, 13.31)	4.04	58.14	0.00	0.00
2.	small2	5	(33.94, 2.00)	8.78	96.53	0.00	0.00
3.	small3	6	(27.48, 16.34)	28.17	161.87	0.00	0.00
4.	medium1	8	(42.94, 0.00)	197.33	1200.00	0.00	0.00
5.	medium2	10	(31.69, 14.15)	493.39	1200.00	8.83	13.79
6.	medium3	12	(40.89, 18.98)	388.19	1200.00	0.00	0.01
7.	large1	16	(64.83, 35.85)	880.98	1800.00	6.14	5.56
8.	large2	20	(112.07, 17.69)	1141.39	1800.00	7.03	8.64
9.	large3	25	(80.76, 46.94)	724.75	1800.00	12.43	10.46

7.3 Eksperimentalni rezultati dobijeni MA metodom

Parametri implementiranog memetskog algoritma određeni su eksperimentalnim putem. Izbor parametara pop_size, $tour\_size$, $elite\_size$ i mut_prob, k_{max}, max_iter_mem,

max_iter_bvns i *max_iter_ls*, *max_time* za instance male, srednje i velike dimenzije prikazan je u tabeli 7.9. Rezultati MA metode na instancama malih, srednjih i velikih dimenzija prikazani su u tabelama 7.10 i 7.11 za Model 1 i za Model 2, respektivno.

Tabela 7.9: Vrednost parametara MA algoritma

Parametar	small	medium	large
pop_size	50	100	250
tour_size	5	7	20
elite_size	5	7	20
mut_prob	0.5	0.5	0.5
max_iter_mem	100	500	1000
t_{max}	$[\|I\|/3]+1$	$[\|I\|/3]+1$	$[\|I\|/3]+1$
max_iter_bvns	30	30	30
max_iter_ls	20	20	20
max_time	600s	1200s	1800s

Parametri u memetskom algoritmu su korigovani u odnosu na parametre u GA i BVNS metodama, sa ciljem uštede vremena, kao što je slučaj u većini postojećih radova vezanih za memetski algoritam.

Tabela 7.10: Rezultati MA za Model 1

MA - Model 1							
R.br.	Instanca	|I|=|J|	(L,F)	t_best(s)	t_tot(s)	agap(%)	σ(%)
1.	small1	4	(13.95, 10.98)	3.12	23.58	0.00	0.00
2.	small2	5	(33.94, 2.00)	0.47	12.80	0.00	0.00
3.	small3	6	(25.95, 15.95)	2.58	37.60	0.00	0.00
4.	medium1	8	(42.94, 0.00)	9.12	79.87	0.00	0.00
5.	medium2	10	(62.80, 4.89)	48.59	229.27	0.00	0.00
6.	medium3	12	(58.85, 9.00)	43.25	191.93	0.34	1.02
7.	large1	16	(68.79, 31.91)	148.24	1800.00	0.81	0.66
8.	large2	20	(125.77, 4.00)	137.90	1800.00	3.64	3.34
9.	large3	25	(125.73, 26.94)	341.79	1800.00	0.65	0.79

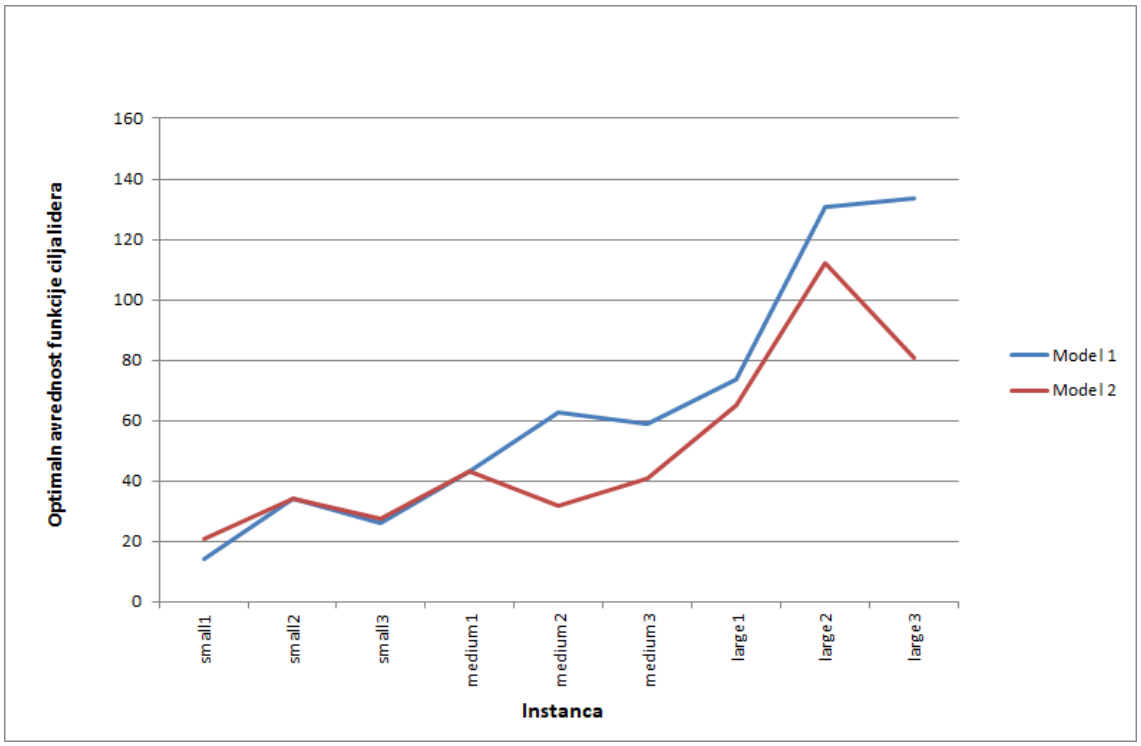
Tabela 7.11: Rezultati MA za Model 2

MA - Model 2							
R.br.	Instanca	|I|=|J|	(L,F)	t_best(s)	t_tot(s)	agap(%)	σ(%)
1.	small1	4	(20.60, 13.31)	0.65	30.91	0.00	0.00
2.	small2	5	(33.94, 2.00)	1.31	22.31	0.00	0.00
3.	small3	6	(27.48, 16.34)	25.41	73.79	0.00	0.00
4.	medium1	8	(42.94, 0.00)	16.99	180.37	0.00	0.00
5.	medium2	10	(31.69, 14.15)	88.26	430.59	3.41	4.95
6.	medium3	12	(40.89, 18.98)	55.79	305.64	0.01	0.02
7.	large1	16	(61.85, 23.94)	278.93	1800.00	7.11	5.99
8.	large2	20	(114.19, 21.57)	220.17	1800.00	1.44	1.32
9.	large3	25	(90.73, 57.95)	223.30	1800.00	15.98	8.01

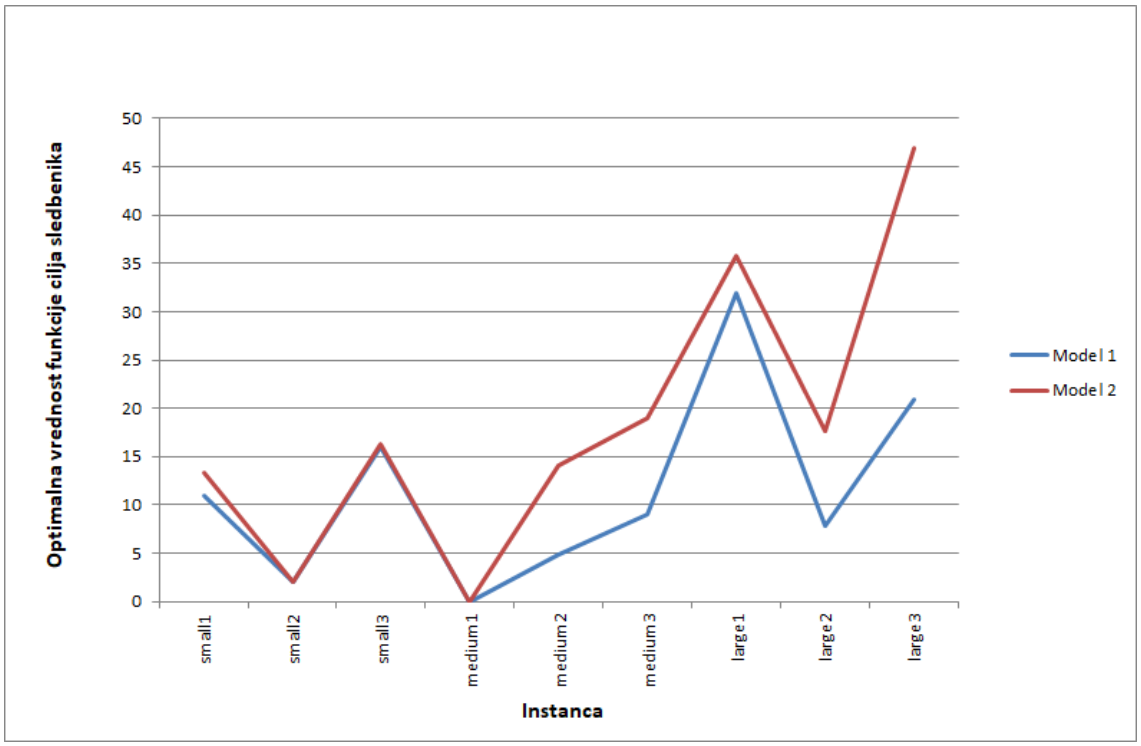
7.4 Analiza uticaja parcijalnog zadovoljenja potražnje

Kao što je ranije rečeno u radu, u Modelu 1 korisnik se može vezati za sledbenika samo u slučaju da sledbenik može da zadovolji celu potražnju korisnika. U Modelu 2, ovaj uslov je oslabljen u smislu da korisnik može da se veže sa sledbenika i u slučaju da sledbenik može da zadovolji samo deo potražnje korisnika, bez obzira da li je navedeni korisnik već vezan za lidera ili ne. Na ovaj način sledbenik poboljšava svoju poziciju u odnosu na Model 1.

Analizirajući rezultate uočava se da se vrednost funkcije cilja sledbenika u Modelu 2 uglavnom povećava u odnosu na Model 1. Takođe, uočeno je da se vrednost funkcije cilja lidera za instance male dimenzije povećava ili ostaje ista u Modelu 2 u odnosu na Model 1, ali se za instance srednje i velike dimenzije ova vrednost smanjuje ili ostaje ista. Na slikama 7.2 i 7.3 prikazani su grafici zavisnosti vrednosti funkcije cilja lidera i vrednosti funkcije cilja sledbenika, respektivno, od dimenzije problema za Model 1 i Model 2 , dobijeni BVNS metodom.



Slika 7.2: Zavisnost vrednosti funkcije cilja lidera od Modela 1 i 2



Slika 7.3: Zavisnost vrednosti funkcije cilja sledbenika od Modela 1 i 2

Primera radi, u nastavku će se analizirati rešenje dobijeno od ulaznih instanci *small3* gde važi $n = m = 6$ i $k = 2$ za oba modela. U Modelu 1 najbolje rešenje je da lider uspostavi objekte na lokaciji $i = 2$ i $i = 4$ kapaciteta $s = 0$, a da sledbenik uspostavi objekat na lokaciji $i = 3$ kapaciteta $s = 1$. Nakon odabira lokacija za uspostavljanje objekata od strane rivala korisnici se pridružuju objektima koji su im najbliži, a zadovoljavaju celu potražnju korisnika. Na taj način, kupci $j = 1$ i $j = 2$ vezuju se za liderov objekat $i = 2$, a kupci $j = 4$ i $j = 5$ vezuju se za liderov objekat $i = 4$. Korisnici $j = 0$ i $j = 3$ vezuju se za sledbenikov objekat $i = 3$. Vrednost funkcije cilja lidera u ovoj igri iznosi 25.95, a vrednost funkcije cilja sledbenika u ovoj igri iznosi 15.95. Rezultat igre prikazan je u tabeli 7.12.

Tabela 7.12: Rešenje n=m=6 Model 1

Korisnik	0	1	2	3	4	5
Liderov objekat		2	2		4	4
Sledbenikov objekat	3			3		

U Modelu 2 najbolje rešenje je da lider uspostavi objekte na lokaciji $i = 3$ i

$i = 4$, kapaciteta $s = 0$, a da sledbenik uspostavi objekat na lokaciji $i = 0$ kapaciteta $s = 1$. Nakon odabira lokacija za uspostavljanje objekata od strane rivala korisnici se pridružuju objektima koji su im najbliži, a zadovoljaju celu potražnju ili samo deo potražnje korisnika. Na taj način, kupci $j = 0$ i $j = 1$ vezuju se za liderov objekat $i = 3$, korisnici $j = 3$, $j = 4$ i $j = 5$ vezuju se za liderov objekat $i = 4$. Korisnici $j = 0$ i $j = 2$ vezuju se za sledbenikov objekat $i = 0$. Kao što se može primetiti, kupac $j = 0$ se vezao i za liderov i za sledbenikov objekat. Korisnik $j = 0$ je kod sledbenika zadovoljio 80% svoje potražnje, a 20% potražnje zadovoljio je kod lidera, što je dato kroz promenljivu $\gamma_{ij} = \gamma_{00}$, koja u ovom slučaju iznosi 0.8. Vrednost funkcije cilja lidera u ovoj igri iznosi 27.48, a vrednost funkcije cilja sledbenika u ovoj igri iznosi 16.34. Rezultat igre prikazan je u tabeli 7.13.

Tabela 7.13: Rešenje n=m=6 - Model 2

Korisnik	0	1	2	3	4	5
Liderov objekat	3	3		4	4	4
Sledbenikov objekat	0		0			
γ_{ij}	0.8					

Analizom navedenog primera uočava se da su se vrednosti funkcije cilja lidera i sledbenika povećavale u Modelu 2 u odnosu na Model 1.

7.5 Analiza rezultata

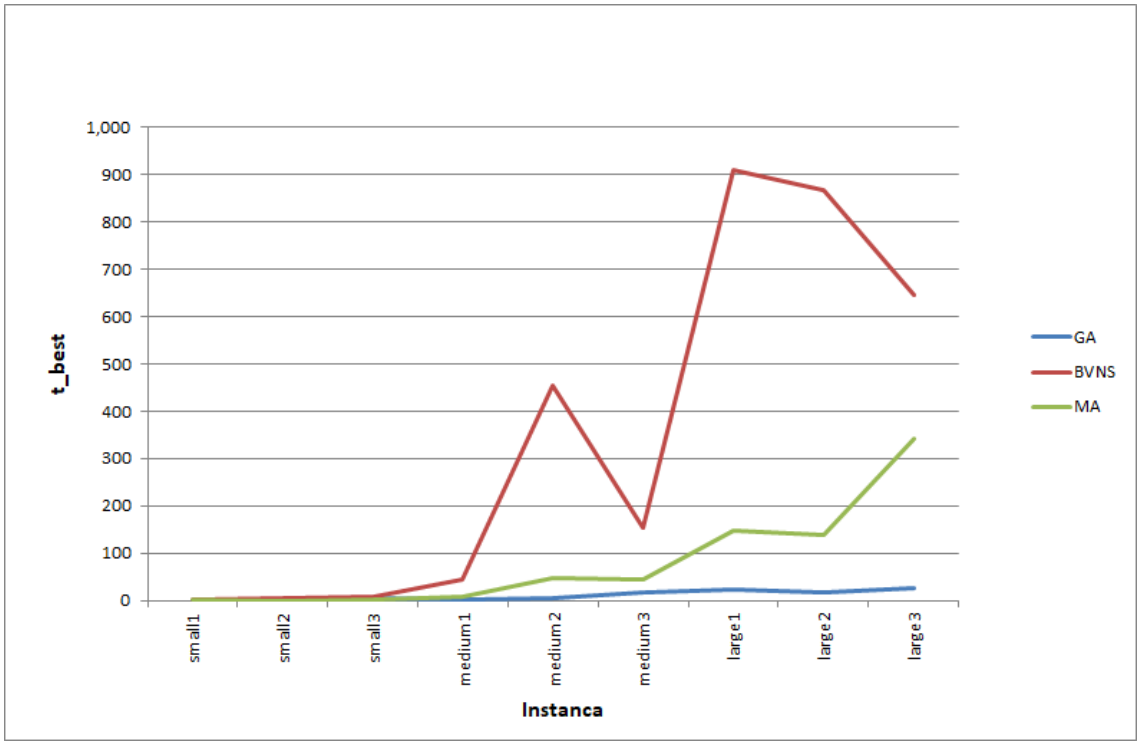
Poređenjem predloženih implementacija prema standardnom odstupanju σ zaključuje se da je najbolja implementirana metoda memetski algoritam, što je i očekivano, a potom BVNS algoritam. Naime, prosečno standardno odstupanje u memetskom algoritmu je za instance male dimenzije iznosi 0.00% (0.00%), za instance srednje dimenzije 0.34% (1,65%) i za instance velike dimenzije 1,59% (5,10%) za Model 1 (Model 2). Prosečno standardno odstupanje u BVNS algoritmu je za instance male dimenzije 0.00% (0.00%), za instance srednje dimenzije 0.00% (4,59%) i za instance velike dimenzije 4,66% (8,22%) za Model 1 (Model 2). Prosečno standardno odstupanje u GA algoritmu je za instance male dimenzije 0.4% (0%), za instance srednje dimenzije 4,27% (6,27%) i za instance velike dimenzije 5,4% (6,00%) za Model 1 (Model 2). Implementirane metode prave veće standardno odstupanje u Modelu 2 u odnosu na Model 1, što je i očekivano s obzirom na složenost navedenih modela.

Za pronalazak najboljeg rešenja prvi put genetskom algoritmu je na instancama male dimezije potrebno u proseku 1,59 sekundi, za instance srednje dimenzije 8.53 sekunde i za instance velike dimenzije 22,69 sekundi, za model 1. Posmatrajući implementirane metode uočava se da više vremena potrebnog da dostignu najbolje rešenje treba algoritmima implementiranim za Model 2. U Modelu 2 za pronalazak najboljeg rešenja prvi put genetskom algoritmu je na instancama male dimezije potrebno u proseku 3,32 sekunde, za instance srednje dimenzije 34.14 sekunde i za instance velike dimenzije 89,07 sekundi. Za instance velike dimezije GA ne zadovolji kriterijum zaustavljanja koji se odnosi na maksimalan broj iteracija, već se zaustavi kada dostigne vreme određeno za maksimalno vreme izvršavanje algoritma, što je u ovom radu 1800 sekundi. Najviše vremena za pronalazak najboljeg rešenja potrebno je BVNS algoritmu. Za instance male dimenzije potrebno je 14.88(13,66) sekunde, za instance srednje dimenzije 218.33(359,25) sekunde i za instance velike dimenzije 906(915) sekundi za Model 1(Model 2). Ukupno vreme izvršavanja programa ograničeno je na 1200 sekundi za instance srednje dimenzije i 1800 sekundi za instance velike dimenzije, što BVNS algoritam dostigne na testiranim instancama. Memetskom algoritmu treba više vremena da dostigne najbolje rešenje od GA, a manje od BVNS algoritma.

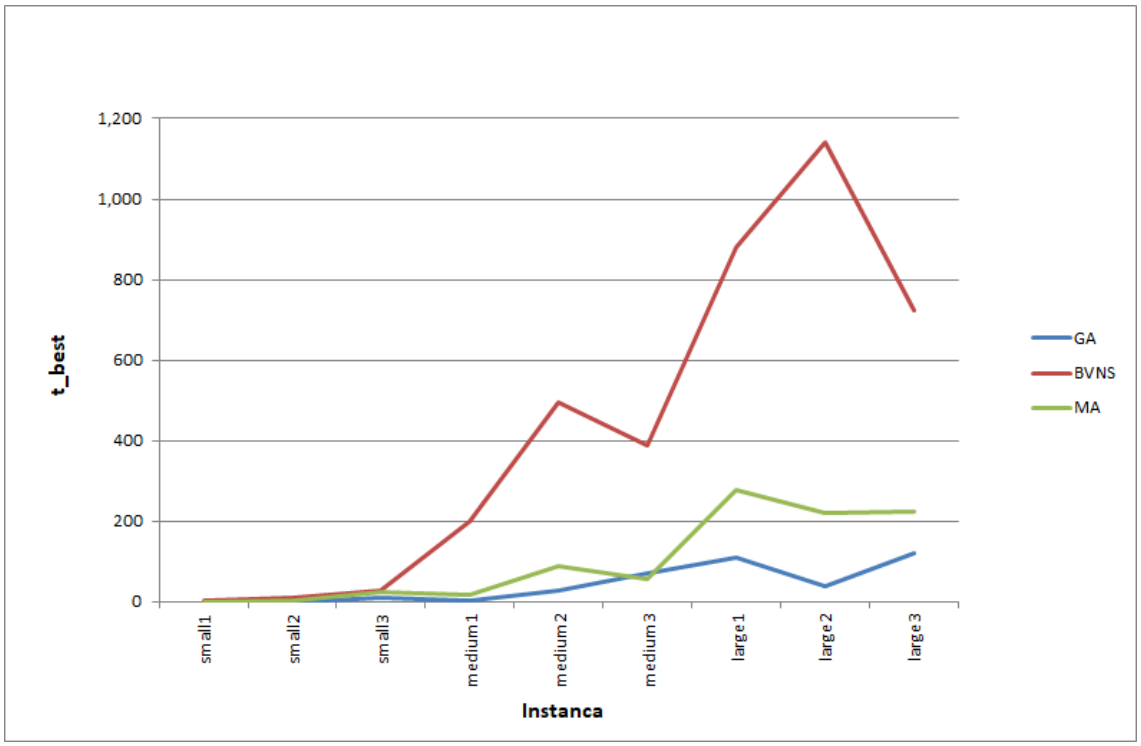
Ukoliko je u tabelama sa eksperimentalnim rezultatima $t_{tot} < t_{max}$ algoritam je prekinut jer je dostigao maksimalan broj iteracija. Ukoliko je $t_{tot} = t_{max}$ algoritam je prekinut zbog vremenskog ograničenja izvršavanja. Na primer, za sve instance velike dimenzije svi algoritmi su prekinuti zbog vremenskog ograničenja, dok je BVNS algoritam prekinut usled vremenskog ograničenja za instance srednje dimenzije medium1 (za Model 2), kao i za instance medium2 i medium3(za oba modela).

Na slikama 7.4 i 7.5 prikazani su grafici zavisnosti prosečnog vremena neophodnom za sva tri algoritma da pronađe najbolje rešenje prvi put za Model 1 i za Model 2, respektivno. Iako važi da su populacijske metaheuristike sporije od heuristika zasnovanih na iterativnom pretraživanju jednog rešenja, u ovom radu to nije slučaj. Razmatrani problem je lokacijski problem sa nadmetanjem, pa se za svako rešenje lidera poziva standardni rešavač CPLEX kako bi se dobilo optimalno rešenje sledbenika. Iako genetski algoritam u svakoj iteraciji posmatra pop_{size} jedinki, standardni rešavač se poziva samo jednom za svaku jedinku u populaciji, odnosno pop_{size} puta u jednoj iteraciji algoritma. U BVNS metodi se CPLEX poziva pri svakoj promeni rešenja u fazi razmrdavanja, kao i pri svakoj promeni rešenja u fazi lokalne pretrage, što rezultuje dužim izvršavanjem algoritma nego što je očekivano,

dužim i od implementirane populacijske heuristike.



Slika 7.4: Prosečno vreme neophodno algoritmima za pronalazak najboljeg rešenja - Model 1



Slika 7.5: Prosečno vreme neophodno algoritmima za pronalazak najboljeg rešenja - Model 2

Što se tiče Modela 1, za instance male dimenzije sve implementirane metaheuristike postižu isto najbolje rešenje. Za instance srednje dimenzije medium1 i medium3 sve metaheuristike nalaze isto rešenje, ali za instancu medium2 BVNS i MA metaheuristike nalaze bolje rešenje od GA. Iz navedenog može se zaključiti da je memetski algoritam iskoristio dobre karakteristike BVNS metode.

Iako je očekivano da memetski algoritam dostigne bolja rešenja od GA i BVNS, za instance velike dimenzije to nije slučaj. Za instance velike dimenzije GA i BVNS algoritmi su našli bolja rešenja od MA algoritma. Razlog tome je to što se u memetskom algoritmu maksimalno vreme izvršavanja BVNS algoritma mora ograničiti, kako MA ne bi previše vremena potrošio na izvršavanje BVNS algoritma. Iz tabela 7.7 i 7.8 može se zaključiti da je BVNS algoritmu potrebno puno vremena da pronađe najbolje rešenje (za instance male dimenzije algoritmu je potrebno da prvi put dostigne najbolje rešenje prosečno za 13.66 sekundi, za instance srednje dimenzije 432 sekunde i za instance velike dimenzije 910 sekundi). Ograničavanjem vremena izvršavanja BVNS algoritma u hibridizaciji dešava se to da BVNS metoda nema vremena da dostigne dovoljno dobro rešenje, a genetskom algoritmu se oduzme vre-

me za izvršavanje. Za instancu large1, najbolje rešenje za lidera koje su našli GA i BVNS algoritam je 73.76, dok je najbolje rešenje za lidera koje je našao MA algoritam 68.79. Za instancu large2, BVNS je našao bolje rešenje od GA, ali mu je bilo potrebno dosta duže vremena za pronalazak najboljeg rešenja. Dok je za instancu large3 GA algoritam našao bolje rešenje od BVNS. Posmatrajući rešenja koja su pronađena u Modelu 1, pokazuje se da je BVNS metoda bolja od GA, ali joj je potrebno više vremena za izvršavanje.

Što se tiče Modela 2, za instance male i srednje dimenzije sve implementirane metaheuristike dostižu ista rešenja, ali je BVNS metodi potrebno najviše vremena za izvršavanje. Za instance velike dimenzije je isti slučaj kao u Modelu 1. GA i BVNS metode dostižu bolje rešenje od MA algoritma iz istog razloga kao u Modelu 1. Za instancu large1 bolje rešenje dostiže BVNS algoritam, a za instance large2 i large3 bolje rešenje dostiže GA.

U Modelu 1 se dešava slučaj da lider izabere lokacije za uspostavljanje objekata tako da se sledbeniku ne isplati da uspostavi objekat ni na jednoj lokaciji, a da pritom profitira. Na taj način sledbenik ostaje bez profita (npr. rezultati GA za instance medium1 i medium2). U ostalim slučajevima lider profitira više od sledbenika prosečno za instance male dimenzije 562.02%, za instance srednje dimenzije 553.89% i za instance velike dimenzije 353.28% (posmatrajući rezultate koji su dobijeni GA metodom). Usled parcijalnog zadovoljenja potražnje u problemu sledbenika, u Modelu 2 se ređe dešava slučaj da sledbeniku ne preostane nijedna strategija kako ne bi ostao bez profita. I u ovom modelu lider profitira više od sledbenika, za instance male dimenzije prosečno za 573.32% više, za instance srednje dimenzije 119.7% i za instance velike dimenzije 232.31%. Na instancama srednje i velike dimenzije uočava se da je razlika u Modelu 2 između profita koji zarađuju lider i sledbenik manja od navedene razlike u Modelu 1.

Glava 8

Zaključak

U ovom radu izložene su dve varijante lokacijskog problema ograničenih kapaciteta sa nadmetanjem. Prva varijanta podrazumava da oba igrača moraju zadovoljiti celu potražnju korisnika, dok se u drugoj varijanti problema ovaj uslov popušta, ali samo za sledbenika s obzirom da on svoj potez povlači drugi. Drugim rečima, sledbenik ima pravo da zadovolji samo deo potražnje korisnika, bez obzira da li je tog korisnika već izabrao lider ili ne. Cilj oba igrača je da maksimizuju svoj profit tako što će uspostaviti objekte na određenim lokacijama, a profit im donose korisnici koji se vezuju za uspostavljene objekte. Lokacije na kojima će se uspostaviti objekti biraju se tako da svi uslovi budu zadovoljeni.

Osnovna varijanta ovog problema primenjena je na princip monopola na tržištu. Međutim, usled razvoja ekonomije, stvorene su potrebe da se razvije model za princip duopola. Problem koji se razmatra u radu ima veliku primenu u prirodi. S obzirom na složenost modela, za probleme velike dimenzije egzaktnim metodama nije moguće rešiti navedeni lokacijski problem, stoga se značaj implementiranih heurističkih metoda razvijenih u radu povećava. U ovom radu predložena je i implementirana jedna populacijska metaheuristika - genetski algoritam i jedna heuristika zasnovana na iterativnom poboljšanju jednog rešenja - osnovna metoda promenljivih okolina. Treća heuristika koja je predložena i implementirana u radu predstavlja hibridizaciju prve dve, čime je stvoren memetski algoritam. Cilj ove hibridizacije je da se iskombinuju dobre strane P i S heuristika, kako bi se dobilo kvalitetnije rešenje ili uštedelo na vremenu izvršavanja algoritma. Predložene heuristike testirane su na javno dostupnim instancama male, srednje i velike dimenzije. Analizirajući rezultate uočava se da je memetski algoritam najpogodniji za rešavanje ovog problema, potom osnovna metoda promenljivih okolina, i naposletku genetski algoritam.

Analizom se naslućuje da su za ovaj problem pogodnije heuristike zasnovane na iterativnom poboljšanju jednog rešenja. Buduća istraživanja mogu se bazirati na analiziranju navedene pretpostavke. U drugoj varijanti problema moguć je slučaj da neki korisnik zadovolji samo deo potražnje kod sledbenika, a ni jedan deo potražnje kod lidera, time korisnik ostaje nezadovoljan, što nije poželjan slučaj. U daljem radu moguće je napraviti treću varijantu problema u kojoj korisnik može da zadovolji deo potražnje kod lidera, a deo kod sledbenika, ali bez mogućnosti da korisnik ostane nezadovoljan na kraju igre. Zbog složenosti algoritma, liderov optimizacioni problem rešavan je predloženim heuristikama, dok je sledbenikov optimizacioni problem rešavan egzaktnim rešavačem. U budućem radu moguće je pokušati rešiti oba problema nekim heurističkim metodama. Za male dimenzije ovo bi bio neefikasniji slučaj u odnosu na trenutni, ali za instance velike dimenzije heuristička metoda bi bila efikasnija od egzaktnog rešavača.

Bibliografija

[1] Weber A. Ueber den standort der industrien. *Ripol Klassik*, Vol. 2, 1909.

[2] Z. Stanimirović i P. Stanojević A. Đenić, M. Marić. A variable neighbourhood search method for solving the long-term care facility location problem. *Journal of Management Mathematics 206*, 2016.

[3] Beresnev, 2016. on-line: http://gcc.gnu.org/.

[4] V.L. Beresnev. Local search algorithms for the problem of competitive location of enterprises. *Automation and Remote Control*, pages 425–539, 2021.

[5] J.A. Moreno-Perez i D.R. Santos-Penate C.C. Rodriguez. Particle swarm optimization with two swarms for the discrete (r|p) centroid problem. *Springer*, 2011.

[6] C. Darwin. On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life. *John Murray*, 1859.

[7] A.V. Plyasunov i A. Savić D.D. Čvokić, Y.A. Kochetov. A variable neighborhood search algorithm for the (r|p) hub-centroid problem under the price war. *Springer*, 2021.

[8] Nebojša V. Bačanin Dzakula. Unapređenje hibridizacijom metaheuristika inteligencije rojeva za rešavanje problema globalne optimizacije. *PhD Thesis*, 2015.

[9] A.A. Freitas i C. Carnieri E. S. Correa, M.T.A. Steiner. A genetic algorithm for solving a capacitated p-median problem. *Numerical Algorithms 35.2*, pages 373–388, 2004.

[10] Aleksandar Đenić. Rešavanje diskretnih lokacijskih problema primenom metode promenljivih okolina. *PhD Thesis*, 2018.

[11] C. Cotta i P. Moscato F. Neri. Handbook of memetic algorithms. *Springer*, 379.

[12] D.E. Goldberg. Genetic algorithms in search, optimization and machine learning. *Addison -Wesley Publishing Company*, 1989.

[13] Hotelling H. Stability in competition. *Economic Journal*, 39.153:41–57, 1929.

[14] N.Aras i K.Altinel H. Kucukaydin. A leader–follower game in competitive facility location. 39:437–448, 2012.

[15] J. H. Holland. Adaption in natural and artificial systems. *The University of Michigan Press*, 1975.

[16] A. Karakitsiou i A.Migdalas. Locating facilities in a competitive environment. *Optimization Letters*, 2016.

[17] D. Serra i C. ReVelle. Competitive location and pricing on networks. *Geographical Analysis 31.2*, pages 109–129, 1995.

[18] D. Serra i C. ReVelle. Competitive location in discrete space. *Springer*, 1995.

[19] C. A. Irawan i D. Jones. Formulation and solution of a two-stage capacitated facility location problem with multilevel capacities. *Springer Science+Business Media, LLC, part of Springer Nature 2018*, pages 41–67, 2019.

[20] E. Mansfield i H. H. Wein. A model for the location of a railroad classification yard. *Management Science 4.3*, pages 292–313, 1958.

[21] Z. Drezner i H. W. Hamacher. Facility location: applications and theory. *Springer Science & Business Media*, 2001.

[22] R.Z.Farahani i M. Hekmatfar. Facility location: concepts, models, algorithms and case studies. *Springer*, 2009.

[23] P. Hansen i N. Mladenović. Variable neighbourhood search: methods and applications. *European journal of Operational Research 130.3*, pages 367–407, 2010.

[24] Y. Gur i N. Stier-Moses. The competitive facility location problem in a duopoly: Advances beyond trees. *Operations Research*, 2018.

[25] N. Mladenović i P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997.

[26] C. Cotta i P. Moscato. A gentle introduction to memetic algorithms.

[27] G. Ellison i S. F. Ellison. Strategic entry deterance and the behavior of pharpaceutical incumbents prior to patent expiration. *Nber workig paper series*, 2017.

[28] M. L. Brandeau i S. S. Chiu. An overview of representative problems in location research. *Management science*, 1989.

[29] R. Todosijević i D. Urošević J. Brimberg, N. Mladenović. General variable neighborhood search for the uncapacitated single allocation p-hub center problem. *Springer - Verlag Berlin Heidelberg*, pages 377 – 388, 2016.

[30] M. Sevaux i F. Glover K. Sorensen. A history of metaheuristics. *Springer*, 2016.

[31] A. Karakitsiou. Modeling discrete competitive facility locaiton. *Springer*, 2015.

[32] A. Đenić i P. Stanojević M. Marić, Z. Stanimirović. Memetic algorithm for solving the multilevel uncapacitated facility location problem. *Informatica 25.3*, pages 439–466, 2014.

[33] Z. Stanimirović i P. Stanojević M. Marić. An efficient memetic algorithm for the uncapacitated single allocation hub location problem. *Soft Computing 17.3*, pages 445–466, 2013.

[34] N. Mladenović. Kontinualni lokacijski problemi. *Matematički institut, SANU.*

[35] A. Rahbari i S. Farahmand M.M. Nasiri, V. Mahmoodian. A modified genetic algorithm for the capacitated competitive facility location problem with the partial demand satisfaction. *Coputers & Industrial Engineering*, 2018. on-line at: https://doi.org/10.1016/j.cie.2018.07.045.

[36] E. Erkut i Z. Drezner O. Alp. An efficient genetic algorithm for the p-median problem. *Annals of Operations research 122.1*, pages 21–42, 2003.

[37] E. Erkut i Z. Drezner O. Alp. Solving the uncapacitated hub location problem using genetic algorithms. *Computers & Operations Research 32.4*, pages 967–984, 2005.

[38] Z. Stanimirović i R. Todosijević O. Janković, S. Mišković. Novel formulations and vns-based heuristics for single and multiple allocation p-hub maximal covering problems. *Springer Science + Business Media New Yourk*, pages 259:191–2016, 2017.

[39] A. Lančinskas i J. Žilinskas P. Fernandez, B. Pelegrin. New heuristic algorithms for discrete competitive location problems with binary and partially binary customer behavior. *Elsevier*, 79:12–18, 2017.

[40] O. Berman i D.Krass R. Aboolian. Competitive facility location model with concave demand. *Elsevier*, 181:598–619, 2007.

[41] R. Leaver i J. Sussams R. Burstall. Evaluation of transport costs for alternative factory sites - a case study. *OR*, pages 345 – 354, 1962.

[42] Y. Rapp. Planning of exchange locations and boundaries. *Ericsson Technics 2*, pages 1–22, 1962.

[43] C.R. Reeves. Genetic algorithms, modern heuristics tehniques for combinatorial problems. *John Willy and Sons*, 1993.

[44] J. Quon i A. Charnes S. Wersan. Systems analysis of refuse collection and disposal practices. *American Public Works Association Yearbook, American Public Works Association*, 1962.

[45] A. Makui i R. Ramezanian s.J. Sadjadi, M. Gorji Ashtiani. A mathematical model for competititve location problem with product selection. *Scientia Iranica*, pages 2157–2176, 2020.

[46] J. M. Smith. The theory of evolution. *Cambridge University Press*, 1993.

[47] P. Stanojević. Egzaktne i metaheurističke metode za rešavanje np - teških lokacijskih problema. *PhD Thesis*, 2016.

[48] V.Y. Suslow. Entry deterrance strategies. *Journal of Industrial Organization Education*, 1, 2006.

[49] Z. Drezner i S. Salhi T. Drezner. Solving the multiple competitive facilities location problem. *Elsevier*, 142:138–151, 2002.

[50] T. Friesz i R.Tobin T. Miller. Heuristic algorithms for delivered price spatially competitive network facility location problems. *Springer, Annals of Operations Research*, 34:177–202, 1992.

[51] E. Talbi. Metaheuristics from design to implementation. *John Wiley & Sons*, 2009.

[52] H. Charkhgard i Y. Zhang V. Mahmoodian. A bi-level branch-and-bound algorithm for capacitated competitive facility location problem. *Department of Industrial and Management Systems Engineering, University of South FLorida*, 2018.

[53] D. Valinsky. Symposium on applications of operations research to urban services - a determination of the optimum location of fire-fighting units in new york city. *Journal of the Operations Research Society of America 3.4*, pages 494–512, 1955.

[54] D. Saban i N.E. Stier-Moses Y. Gur. The competitive facility location problem in duopoly. 2014.

[55] T.K. Rakphs i Z. Xue Y. Zhang, L.V. Snyder. The competitive facility location problem under disruption risks. 93:453–473, 2016.

[56] A. Schobel i O.G. Weselowsky Z. Drezner, K. Klamroth. The weber problem. *Springer*, pages 1–36, 2001.

[57] M. Abedian i S. Sharahi Z.R. Farahani. Dynamic facility location problem. *Physical Verlag HD*, pages 347–372, 2009.