Univerzitet u Beogradu
Matematički fakultet

Nina Radojičić

REŠAVANJE NEKIH NP-TEŠKIH PROBLEMA DISKRETNE OPTIMIZACIJE

Master rad

B e o g r a d

2011.

Mentor:	**Doc. dr Miroslav Marić** Matematički fakultet u Beogradu
Članovi komisije:	**Prof. dr Gordana Pavlović-Lažetić** Matematički fakultet u Beogradu
	Doc. dr Zorica Stanimirović Matematički fakultet u Beogradu

Datum odbrane: ____________________________________

Rešavanje nekih NP-teških problema diskretne optimizacije

Rezime

U ovom radu su opisani različiti algoritmi za rešavanje dva NP-teška problema diskretne optimizacije: problem izbalansiranosti lokacija (*The Load Balance problem* - LOBA) i problem maksimizacije minimalnog rastojanja (*Max-Min Diversity Problem* - MMDP).

Uzimajući u obzir široku oblast primene rešavanja ova dva problema sa instancama većih dimenzija, korišćenje metaheuristika predstavlja značajan doprinos u realnim aplikacijama. Oba problema rešavana su primenom i genetskih algoritama (*Genetic algorithms* - GA) i metodom promenljivih okolina (*Variable Neighborhood Search* - VNS), čije su performanse poređene sa rezultatima koji su dobijeni korišćenjem *IBM ILOG CPLEX* softverskog paketa nad matematičkim modelima razvijenim za oba problema. Za potrebe rada, za oba razmatrana problema, implementirana je i hibridizacija opisane dve metaheuristike: GA i VNS.

Metode opisane u radu su implementirane i testirane na odgovarajućim instancama. Eksperimentalno dobijeni rezultati predstavljaju optimalna rešenja u slučajevima uporedivih manjih dimenzija, ali daju kvalitetna rešenja i za razmatrane probleme većih dimenzija. Predložene metode su pokazale dobre performance u primenama na razmatranim problemima, pa se mogu koristiti za rešavanje nekih sličnih diskretnih lokacijskih problema.

Ključne reči: *Diskretna optimizacija, Heurističke metode, NP-teški problemi, Genetski algoritmi, Metoda promenljivih okolina, Hibridizacija*

Solving some NP-hard discrete optimization problems

Abstract

In this paper various algorithms for solving two NP-hard discrete optimization problems are described: The Load Balance problem (LOBA) and the Max-Min Problem Diversity (MMDP).

Taking in consideration the wide area of applications of solver for larger instances of these two problems, the use of metaheuristics gives a significant contribution to a real application. Both problems are solved using the Genetic Algorithms (GA) and the Variable Neighborhood Search (VNS), whose performance were compared with results obtained using the IBM ILOG CPLEX software package on mathematical models developed for both problems. For the purpose of this paper, a Hybrid heuristic method, based on the incorporation of the Genetic Algorithms (GA) into modification of the Variable Neighborhood Search (VNS), is presented as well.

The methods described in this paper have been implemented and tested on appropriate instances. The obtained experimenatal results, in comparable cases of smaller dimensions, are optimal solutions, but also good quality solutions for larger dimensions (without proving exactness) are presented. The presented methods have good performance in solving two condidered problems and could be also applied for solving similar discrete location problems.

Key words: *Discrete optimization, Huristic methods, NP-hard problems, Genetic algorithms, Variable Neighborhood Search, Hybridization*

PREDGOVOR

U ovom radu razmatra se značajan problem rešavanja dva NP-teška problema diskretne optimizacije.

U uvodnom poglavlju najpre su date osnovne informacije o diskretnoj optimizaciji, o lokacijskim problemima, kao i o NP-teškim problemima, tj. problemima za čije rešavanje nisu poznati algoritmi čija se složenost može izraziti polinomijalnom funkcijom. Nakon toga, takođe u uvodnom poglavlju, daju se osnovne informacije o heuristikama, odnosno o dve metaheuristike koje se koriste za ovaj rad i to: genetski algoritmi (njihov osnovni koncept i njihova primena) i metoda promenljivih okolina okolina (*Variable neighborhood search* - VNS).

Drugo poglavlje je posvećeno problemu izbalansiranosti lokacija (*The Load Balance problem* – LOBA). Dat je opis LOBA problema, matematička formulacija problema, navedene neke primene tog problema. Dalje, u istom poglavlju, se opisuje kako je problem LOBA rešavan primenom genetskih algoritama, dat je opis kodiranja, načina kako su realzovani genetski operatori i navedeni su i ostali koncepti genetskih algoritama koji su bili važni za implementaciju GA za problem LOBA. Zatim, u nastavku drugog poglavlja prikazano je rešavanje problema LOBA primenom VNS-a, osnovni koncepti VNS-a, kodiranje prostora rešenja problema LOBA za VNS i koncept okolina, kao i uprošćeni VNS koji ima za cilj poboljšanje efikasnosti i koji je implementiran za potrebe ovog rada. Nakon uspešnosti primene prethodno navedenih metoda, genetskog algoritma i VNS-a, na instance problema LOBA, usledila je i njihova hibridizacija. Tako da je i način njihove hibridizacije takođe opisan u poglavlju dva. Zatim su prikazani eksperimentalni rezultati na osnovu kojih se može steći uvid u kvalitet predloženih metoda, a opisane su i instance na kojima je vršeno testiranje.

Problem maksimizacije minimalnog rastojanja (MMDP) prikazan je u trećem poglavlju. Dat je opis i matematička formulacija MMDP problema. Za rešavanje problema MMDP korišćene su iste tehnike kao i za rešavanje problema LOBA, uz odgovarajuće modifikacije, pa je ovo poglavlje znatno kraće. Dalje, u ovom poglavlju su prikazani eksperimentalni rezultati rešavanje problema MMDP primenom genetskih algoritama, VNS-a kao i njihove hibridizacije.

Zaključna razmatranja izložena su u četvrtom poglavlju, gde je dat osvrt na postignute rezultate, najznačajnije doprinose u ovom radu, kao i ideje za dalja proširenja i unapređenja razmatrane problematike.

Želim da se zahvalim svima koji su mi pomogli da aktivnosti oko pripreme i izrade ovog rada realizijem. Posebno bih želela da se zahvalim mentoru dr Miroslavu Mariću na korisnim savetima, podršci i razumevanju toku izrade ovog rada.

Želim da istaknem zahvalnost dr Zorici Stanimirović koja me je zainteresovala za ovu problematiku i značajno pomagla u realizaciji, kao i konačnom oblikovanju, ovog rada.

Zahvaljujem se prof**.** dr Gordani Pavlović-Lažetić na korisnim sugestijama koje su doprinele da se kvalitet rada poboljša.

S obzirom da je za implementaciju genetskog algoritma u ovom radu korišćenja open source biblioteka funkcija GAFramework, koju su implementirali dr Miroslav Marić i Srđan Božović, zahvaljujem im se na instrukcijama i korisnim sugestijama prilikom implementacije genetskog algoritma za probleme koje sam rešavala.

Posebno se zahvaljujem svojoj porodici i prijateljima na razumevanju i podršci tokom izrade ovog rada.

Kandidat

Beograd, 2011. Nina Radojičić

Sadržaj

1. UVOD

Termin optimizacija ima široku upotrebu u različitim oblastima kao što su prirodne, tehničke, društveno-ekonomske nauke. Optimizacija se definiše kao nauka koja određuje "najbolje" rešenje određenog matematički definisanog problema [Opr92]. Matematički gledano, problem optimizacije se svodi na određivanje ekstremne vrednosti funkcije cilja.

Različite metode optimizacije pojavljuju se i razvijaju kao implikacija sve većeg interesa ljudi da brojne probleme reše na najbolji mogući način. Mnogi realni problemi u praksi imaju osobinu da je jednostavno naći neko dopustivo rešenje, ali je teško naći optimalno rešenje, tj. najbolje koje se može postići. Želja za traženjem najboljih rešenja tokom vremena je postajala sve intenzivnija i opravdanija. Posedovanjem najboljeg rešenja ili barem njemu bliskog rešenja sve više je značilo, u različitim sferama ljudske delatnost, ogromne uštede u vremenu, materijalu, energiji, finansijske efekte, kao i postizanje veće pouzdanosti, sigurnosti itd.

Veliki broj praktičnih problema mogu se okarakterisati kao optimizacioni problemi koji podrazumevaju funkciju cilja, uslove i sistem ograničenja. Takva funkcija cilja $f(x)$ preslikava skup mogućih rešenja S u realne vrednosti, tako da je $f(x): S \to \mathbb{R}$. Optimizacija je zapravo globalna pretraga ekstremuma (maksimuma ili minimuma) u odnosu na funkciju cilja.

Optimizacioni problem se može formulisati kao:

$$\min\{f(x) | x \in X, X \subseteq S\}$$

Gde su S; X ; x i f prostor rešenja, dopustiv skup, dopustivo rešenje i realno vrednosna funkcija, redom.

Ako je S konačan ili beskonačno prebrojiv skup onda je definisan problem diskretne optimizacije. Ako je, $S = \mathbb{R}^n$ onda je reč o kontinualnoj optimizaciji.

1.1. Diskretna optimizacija

Problem diskretne optimizacije se može formulisati na sledeći način:

Dat je konačan ili prebrojivo beskonačan, diskretan, skup S i funkcija $f: S \to \mathbb{R}$. Naći minimum funkcije f na skupu S, tj. rešiti zadatak

$$\min_{x \in S} f(x). \qquad (1)$$

Skup S se naziva *dopustivi skup*, a funkcija f *funkcija cilja.* Za tačku $x \in S$ se kaže da je *dopustivo rešenje* navedenog problema. Podrazumeva se da je potrebno naći sva dopustiva rešenja x^* (ili bar neko od njih) takva da je $f(x^*) = \min_{x \in S} f(x)$. Takva rešenja se nazivaju *optimalna rešenja*.

Kako je $\max_{x\in S} f(x) = -\min_{x\in S}(-f(x))$, problem maksimizacije se svodi na problem minimalizacije, pa je dovoljno razmatrati samo jedan od ova dva problema.

Ovim opštim modelom se može predstaviti široka lepeza problema koji su izvorno iz različitih oblasti. U matematici se pomoću njega mogu modelirati problemi koji se javljaju u kombinatorici, teoriji grafova i logici.

Kada govorimo o primeni diskretne optimizacije svakako treba napomenuti one koji se odnose na cilj povećavanja produktivnosti u oblasti upravljanja resursima. Tu spadaju problemi raspoređivanja poslova i mašina, optimalnog transporta, optimalnog dizajna itd. Razvijaju se i primene diskretne optimizacije u oblastima molekularne biologije, fizike i kristalografije.

Primetimo da se zbog diskretnosti dopustivog skupa S za rešavanje problema (1) ne mogu primeniti numeričke metode zasnovane na klasičnoj matematičkoj analizi. Međutim, jasno je da za takav problem sa konačnim skupom S uvek postoje algoritmi potpune pretrage. Pak, ukoliko je kardinalnost skupa S velika, potunu pretragu nije pogodno primeniti.

Opširnije u diskretnoj optimizaciji može se naći u [Cve96], [Cook97], [Law01], [Cve04], [Lee04], [Sch10], [Kah11] itd. Sveobuhvatan prikaz istorijata kombinatorne optimizacije daje Schrijver u navedenoj referenci [SchA].

1.2. O lokacijskim problemima

Matematičko rešavanje lokacijskih problema privuklo je veliki broj istraživača diskretne i kontinualne optimizacije skoro više od četiri decenje. Pod diskretnim lokacijskim problemom se podrazumeva problem odlučivanja gde da se lociraju objekti iz konačnog skupa, tako da na taj način određeni kriterijum je optimizovan. Istraživači su se fokusirali kako na algoritme tako i na formulacije za različite postavke u privatnim sektorima (npr. industrijska postrojenja, banke, maloprodajni objekti, itd.), i u javnom sektoru (npr. ambulante, klinike, itd.). Svaka formulacija ima sličnosti i razlike u odnosu na ostale, ali osobenosti svakog problema omogućavaju mogućnost za rad stotinama istraživača. Prisustvo nelinearnosti i zahtevi da su vrednosti promenljivih 0 i 1 okupirali su istraživače u ovoj oblasti koja se sve više razvija. U radu [ReV05] se može pronaći pregled većine poznatih lokacijskih modela i metoda za njihovo rešavanje.

Tradicionalno, u većini radova iz literature koji se bave determinističkom teorijom lokacijskih problema u mrežama, posmatrani su medijana i centri kao načini merenja, kao i varijacije ta dva pristupa. Lokacijski problemi defisani na mrežama doveli su u pitanje objektivnost prilikom odabira lokacija na takve načine. Zato su u radu [CLM01] uvedeni novi lokacijski problemi u kojima se kriterijum bazira na disperziji distribucije rastojanja od snabdevača do svih korisnika. Posebno u radu [CLM01] se razmatra merenje maksimuma apsolutne devijacije kod lokacijskih problema na mrežama.

Lokacijski problemi sa jednostrukim alokacijama sa objektivnim merenjem, definisani na mrežama razmatraju se u radu [Mes03]. U ovom radu dat je pregled poznatih algoritamskih rešenja i prezentovani su unapređeni algoritmi za neke od sledećih modela: varijansa, suma težina apsolutne devijacije, maksimum težina apsolutne devijacije, rang, i Lorencova mera.

Diskretni problem lokacije objekata sa novom formom kriterijuma objektivnosti razmatra se u radu [Esp09]. Predloženi model se odnosi na slučaj kada tačke potražnje imaju strogu prednost kada je lociranje snabdevača u pitanju.

U radu [Hua11], dati su modeli za poboljšanje rutiranja i razmatra se njihova objektivnost, efikasnost i efektivnost. Slična rutiranja i odluke raspodele se izučavaju u komercijalnim svrhama, gde je efikasnost, u smislu minimiziranja troškova, primarni cilj. Ativnosti koje su razmatrane u radu se usložnjavaju prisustvom višestrukih ciljeva (osim minimiziranja troškova). Međutim, kvantifikovanje tih ciljeva može biti izazov. Tako su u pomenutom radu, definisane i formulisne metrike i performanse za poboljšanje distribucije. Posebna pažnja je posvećena efikasnosti (tj. u kojoj meri su ciljevi distribucije brzo i adekvatno ispunjeni) i adekvatnosti (tj. u kojoj meri svi korisnci dobijaju potrebne usluge). Istraživali su kako efektivnost, efikasnost i objektivnost utiču na strukturu rutiranja vozila i raspodelu resursa. Identifikovali su trendove i principe rutiranja na osnovu analitičkih karakteristika analiziranog problema i nizom računarskih testova.

Lokacijski problemi se mogu, kako se ističe u radu [Ogr00], razmatrati kao višekriterijumski modeli gde se za svakog klijenta definiše individualna funkcija cilja, koja meri stanje među lokacijama uzimajući u obzir zadovoljenje potreba klijenata (npr. računa rastojanje između klijenta i proizvođača kome je on pridružen). Posebna pažnja stavljena je na rešavanje koncepata baziranim na dvokriterijumskoj optimizaciji.

U radu [Cha01] su proučavana različita uopštenja lokacijskih problema, kada i iz skupa korisnika treba izdvojiti podskup, a ne samo za skup snabdevača.

Bibliografija novijih radova iz široke oblasti lokacijskih modela snabdevača data je u radu [ReV08]. O lokacijskim problemima se može naći u brojnim radovima:[Tch95], [Bar97], [Top05], [Kra06], [Kra07], [Cor10] i mnogim drugim.

1.3. NP-teški problemi

NP-teški problemi predstavljaju probleme za čije rešavanje nisu poznati algoritmi čija se složenost može izraziti polinomijalnom funkcijom (npr. linearnom, kvadratnom, kubnom…). Klasa NP problema definiše se pomoću problema zadovoljivosti (*The Propositional Satisfiability Problem* SAT). Istorijski posmatrano SAT problem je bio prvi problem za koje je dokazana NP-potpunost [Cook71].

U [Cve96] se navodi da, intuitivno, pod algoritmom za rešavanje nekog problema (iz unapred fiksirane klase partikularnih rešenja) podrazumevamo konačan spisak pravila postupajući po kojima dolazimo do rešenja bilo kog partikularnog problema (iz zadate klase) za konačno mnogo koraka.

Glavne ideje teorije algoritama se često izlažu u vezi sa problemima odlučivanja. To su problemi kod kojih se rešenje sastoji u utvrđivanju ili opovrgavanju odgovora: “da” ili “ne”, i na koje se i druge vrste problema mogu svesti. Teorija algoritama omogućava da se napravi jasna granica između problema za čije rešavanje postoje algoritmi (odlučivi problem) i onih za koje to nije slučaj (neodlučivi problem). Detaljna klasifikacija složenosti odlučivih problema se može pronaći u [Ognj04], a u ovom poglavlju su navedeni samo osnovni pojmovi.

Deterministički algoritam je onaj algoritam koji će nam za zadati ulaz uvek dati istu izlaznu vrednost (prolazak uvek kroz tačno definisana stanja), za razliku od nedeterminističkog algoritma koji za isti ulaz može proizvesti različita dešavanja.

P je skup svih problema odlučivanja koji su rešivi determinističkim algoritmom u polinomnom vremenu. NP je skup svih problema odlučivanja rešivih nedeterminističkim algoritmom u polinomnom vremenu. Budući da su deterministički algoritmi specijalni slučaj nedeterminističkih sledi da je $P \subset NP$. Čuveni problem glasi: Da li je $P = NP$? Pretpostavlja se da je $P \neq NP$, iako za to nema dokaza.

Neka su L_1 i L_2 problemi. Kažemo da se L_1 svodi na L_2 ako postoji deterministički polinomni algoritam za L_1 koji poziva pretpostavljeni algoritam L_2, s time da se svaki takav poziv broji kao jedan osnovni korak. Detaljnije o lemi o svodljivosti (kao i njen dokaz) se može pronaći u [Živ00].

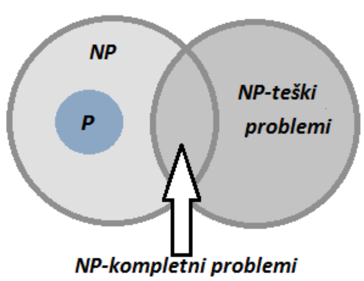
NP – težak problem

Problem L je NP - težak ako se problem zadovoljivosti svodi na L.

NP – kompletan problem

Problem L je NP - kompletan ako je L NP-težak i L $\in$ NP .

Iako nikad nije dokazano, veruje se da odnos skupova NP, P, NP-teških i NP-kompletnih problema, izgleda kao što je prikazano na slici 1.1. Optimizacijski problem može biti NP-težak, a da pritom nije i NP-kompletan.



Slika 1.1 Grafički prikaz odnosa skupova NP, P , NP-teških i NP- kompletnih problema

Problem je nedetermnistički polinomski (NP), što podrazumeva da se za predloženo rešenje u polinomskom vremenu može utvrditi (verifikovati) da li ono pripada ili ne pripada skupu rešenja datog problema. Za deterministički polinomski problem (P) sa druge strane, u polinomskom vremenu moguće je i pronaći rešenje, a ne samo izvršiti njegovu verifikaciju.

1.4. Heuristike

Pod heuristikom se podrazumeva tehnika kojom se traži dobro rešenje zadatka za relativno kratko vreme, bez mogućnosti garantovanja njegove ni dopustivosti ni optimalnosti, ili čak ni njegove bliskosti optimalnom rešenju.

Modeliranje različitih problema iz prakse često dovodi do NP-teških problema, pa se egzaktne metode, iako najpouzdanije, retko koriste za njihovo rešavanje. Zapravo, malo je verovatno da, za probleme iz klase NP-teških, postoji brz ezgzaktan algoritam.

Heuristike se u širokoj upotrebi za rešavanje optimizacionih problema, barem za velike instance ili kao inicijalno, tj. početno rešenje neke određene procedure. Heuristike primenjene na probleme velikih dimenzija daju u razumnom vremenu dobra rešenja, koja su često optimalna iako se optimalnost ne može dokazati.

Klasične heurističke metode se razvijaju u cilju rešavanja jednog i samo jednog problema, ali od 90-ih godina metaheuristički pristup, razvijanja pravila i načela koji se mogu primeniti za rešavanje velikog broja različitih problema, je preovladao.

Za neke od takvih razvijanih metoda, inspiracija je potekla iz prirode. Tako npr. u radovima [Dor99] i [Lia04] prikazuje se primeni mravljih algoritama za problem diskretne optimizacije. Više o mravlim algoritmima može se naći i u [T'ki02] i [Dor06].

U radu [Ary04] su analizirane različite heuristike lokalne pretrage za nekoliko diskretnih lokacijskih problema. U poslednje 3 decenije, lokalna pretragaje sazrelila od jednostavne heurističke ideje u široku oblast istraživanja u okviru diskretne optimiracije. Omogućava osnove za mnoge novije razvijene algoritamske pristupe npr. simulirano kaljenje, genetske algoritme. U tezi [Pan04], predloženi su aproksimativni algoritami za rešavanje lokacijskih problema koje se baziraju tehnika lokalnog pretraživanja.

U istraživanjima za ovaj rad koriste se dve metaheuristike: genetski algoritmi i metoda promenljivih okolina, kao i njihova hibridizacija.

1.4.1. Genetski algoritmi - Osnovni koncept

Početkom 60-tih godina prošlog veka javile su se ideje da se iskoristi princip prirodne evolucije kod izrade algoritma. Kao jedno od ostvarenja te ideje, javlja i genetski algoritam. Teorija genetskih algoritama svoje utemeljenje pronalazi kao abstrakcija biološke evolucije. U cilju detaljnijeg opisa osnova genetskog algoritma, u daljem tekstu rada se nalazi deo biološko računarske terminologije vezane za temu biološke evolucije.

Živi organizmi se sastoje od ćelija, a svaka ćelija sadrži skup hromozoma, koji su izgrađeni od lanaca DNK (eng. DNA - Deoxyribonucleic acid) koji kodiraju informacije o odgovarajućem organizmu. Geni su zapravo funkcionalni blokovi DNK i svaki gen predstavlja odgovarajuće svojstvo organizma. U toku reprodukcije odabrani roditelji rekombinuju (ukrštaju) gene i time se formiraju novi hromozomi. Takođe, mutacija dovodi do malih promena DNK elemenata. Kvalitet (prilagođenost) novonastalog organizma se meri njegovim uspehom u životu. Upravo, na ovim principima izgrađeni su i genetski algoritmi.

Genetski algoritmi predloženi su od strane Johna Hollanda još u ranim sedamdesetim. Holland je u svojoj knjizi [Hll75] predložio (jednostavni) genetski algoritam kao računarski proces koji imitira evolucijski proces u prirodi i primenjuje ga na apstraktne jedinke. Polazni cilj razvoja genetskog algoritma nije bila praktična primena zarad rešavanja nekog specifičnog problema, već formalna studija o evoluciji i adaptaciji u prirodi i načinima na koje je tu logiku moguće ubaciti u računarstvo.

Genetski algoritam je heuristička metoda optimizacije koja imitira proces prirodne evolucije date populacije jedinki. Živa bića se tokom evolucije prilagođavaju uslovima u prirodi, tj. životnoj okolini. Analogija evolucije kao prirodnog procesa i genetskog algoritma kao metode optimizacije, ogleda se u procesu selekcije i genetskim operatorima. Mehanizam odabira nad nekom vrstom živih bića u evolucijskom procesu čine okolina i uslovi u prirodi. Slično kao što su okolina i uslovi u prirodi ključ selekcije nad nekom vrstom živih bića, tako je i funkcija prilagođenosti ključ selekcije nad populacijom rešenja u genetskom algoritmu. Naime, u prirodi jedinka koja je najbolje prilagođena uslovima i okolini u kojoj živi ima najveću verovatnoću preživljavanja i razmnožavanja, a time i prenošenja svog genetskog materijala na svoje potomke. Taj princip da bolje prilagođene jedinke prolaze u narednu generaciju i prenose svoja osobine na svoje potomke primenjuje se i kod genetskih algoritama i utiče na to da se očekuje da je svaka nova generacija jedinki bolje prilagođena nego prethodna. Za genetski algoritam jedinke su potencijalna rešenja problema koji se rešava, a svojstva jedinki su zapisana u hromozomima pomoću genetskog koda. Selekcijom se biraju dobre jedinke koje se prenose u sledeću populaciju, a ukrštanjem genetskog materijala stvaraju se nove jedinke. Takav ciklus selekcije, ukrštanja i mutacije genetskim materijalom jedinki ponavlja se sve dok nije zadovoljen uslov zaustavljanja evolucijskog procesa.

1.4.1.1 Primena genetskih algoritama

Tokom prethodne tri decenije, a posebno u poslednjih nekoliko godina, genetski algoritmi su se pokazali vrlo moćnim i u isto vreme opštim alatom za rešavanje čitavog niza problema iz prakse. Veliki broj istraživača se bavilo prilagođavanju genetskih algoritama velikom broju problema i povećanju efikasnosti. Paralelno s povećanjem primene povećava se i opseg istraživanja rada i svojstava genetskih algoritama i pokušavaju se svesti njihovi elementi na neke teorijske osnove.

Genetski algoritmi se mogu koristiti za rešavanje velikog broja različitih problema u svim domenima primenjene matematike, pa je široka lepeza oblasti primene genetskih algoritama. U daljem tekstu navedene su samo neke od njih.

O primeni GA u oblasti operacionih istraživanja može se naći u [Dow96]. Kako se navodi u literaturi [Gol10] genetski algoritam se primenjuje i daje dobre rezultate u području učenja kod neuronskih mreža, pri traženju najkraćeg puta, problemu trgovačkog putnika, strategiji igara, problemima sličnim transportnom problemu, problemu raspoređivanja procesa, problemu određivanja parametara sistema, optimizacije upita nad bazom podataka itd. GA se primenjuje prilikom projektovanja komunikacijskih (putnih, vodovodnih, elektroenergetskih...) mreža, kao i za finansijske i ekonomske analize i planiranja.

John Koza je 1992. godine predložio upotrebu genetskog algoritma za razvoj računarskih programa [Koz92] za izvršavanje određenih zadataka i otvorio područje genetskog programiranja.

Mogućnosti korišćenja genetskih algoritama u mašinskom učenju prikazane su u radu [Vug96]. Primena genetskih algoritama u nalaženju minimalnog Steinerovog stable se može naći u radu [Lju98]. Specifično je korištenje genetskih algoritama u homogenizaciji klimatoloških zapisa [Jnn06], gde je svrha određivanje vremenskih tačaka kada se javlja nagli skok vrednosti klimatske promenljive. U monografiji [Mil07] predstavljen je razvoj sistema za pripremu i optimizaciju procesa rezanja pri obradi materijala i optimizaciju reznih parametara korišćenjem genetskih algoritama. Rešavanje različitih problema korišćenjem genetskih algoritama razmatra se i u literaturi: [Šeš02], [Tal03], [Čuš03], [Toš04], [Rad10], [MM10] itd.

1.4.2. Metoda promenljivih okolina (VNS)

Metoda promenljivih okolina (*Variable neighborhood search* - VNS) je metaheuristika za rešavanje problema kombinatorne i globalne optimizacije čija je osnovna ideja sistematska promena okolina sa lokalnom pretragom [Han01a]. Lokalno pretraživanje je jedna od najviše korišćenih heuristika. Krenuvši od početnog rešenja, nizom uzastopnih lokalnih promena kojima se svaki put poboljšava vrednost funkcije cilja, dostiže lokalni optimum.

Mladenović i Hansen u radu [Mla95] su predstavili osnovnu šemu VNS-a koja se, kako navode, lako može implementirati pomoću bilo kog algoritam a lokalnog pretraživanja kao potprograma. Ukazali su na efikasnost i ilustrovali poboljšanja na primeru problema trgovačkog putnika.

Metoda promenljivih okolina ima brojne primene u rešavanju raznih problema optimizacije [Mla95], [Mla97], [Han99], [Han01a], [Lib05], [Han07], [Mla11].

Prilikom rešavanja kompleksnijih primera različitih problema, efikasnost VNS-a može biti poboljšana kroz dekompoziciju. Tako se u radu [Han01b] navodi prosirenje: VNDS (Variable Neighborhood Decomposition Search), i ilustrovana je primena tog metoda na problem p-medijane.

2. PROBLEM IZBALANSIRANOSTI LOKACIJA (LOBA)

Problem izbalansiranosti lokacija (*The Load Balance problem* – LOBA) je predstavio A. Marín u radu [Mar11]. Marín je izložio dve formulacije ovog problema celobrojnog programiranja i razvio nekoliko familija validnih nejednakosti i tehnika preprocesiranja koje omogućavaju da se smanji dimenzija (broj promenljivih i uslova) najveće formulacije. Autor je predložio metodu grananja i sečenja (*Branch-and-Cut algorithm* - BnC) sa poboljšanjima i obe formulacije je testirao na instancama sa do 50 potencijalnih snabevača i 100 korisnika. U slučaju najvećih instanci za koje su dobijena optimalna rešenja prezentovana u Marínovom radu, rešenje je dobijeno u oviru jednog sata CPU vremena.

Prva formulacija [Mar11] je nazvana standardna s obzirom da potiče od standardne formulacije za problem p-medijane s tim što su dodate dve promenljive (maksimalni i minimalni broj korisnika dodeljenih nekom snabdevaču kao i ograničenja da se korisnika pridružuje najbližem snadbevaču. Druga formulacija problema LOBA deli osnovnu strukturu sa nekim formulacijama diskretno uređenog problema medijane (*The discrete ordered median problem* – DOMP). Problem DOMP je prvi definisao S. Nickel [Nic01] i predstavlja uopštenje problema p-medijane, čija je NP-kompletnost dokazana u [KH79]. Problem DOMP je NP-težak [Dom03], a više o formulaciji i rešavanju DOMP problema se može videti i u [Sta04] i [Sta07a].

2.1. Opis problema

Jedna od formulacija koju je Marín predložio u referenci [Mar11] korišćena je i u ovom radu. Neka je $\mathrm{A} = \{1,\dots,\mathrm{n}\}$ skup koji se koristi da predstavi korisnike (potrošače) i neka je $\mathrm{B} = \{1,\dots,\mathrm{m}\}$ skup koji predstavlja potencijalne snabdevače (npr. fabrike, postrojenja). Neka je $C = (c_{ij}), i \in A, j \in B$, matrica troškova (cena transporta, razdaljina ili slično). Nisu nametnuti dodatni uslovi za matricu C (elementi matrice mogu biti i negativni brojevi, ne moraju da zadovolje nejednakost trougla ili simetriju).

Rešenje razmatranog problema je dato kao skup snadbevača X ($\mathrm{X} \subseteq \mathrm{B}$), $|\mathrm{X}| = \mathrm{p}$, tako da svaki korisnik se pridružuje snabdevaču koji mu je najbliži. Svaki korisnik $i \in A$ će biti snadbeven od strane snadbevača $j \in X$ takvog da je $c_{ij} = \min_{k \in X} c_{ik}$. U slučaju nerešenog rezultata korisnik može biti dodeljen bilo kojem od njemu najbližih snadbevača.

Zadatak je minimalizovati razliku između maksimuma i minimuma broja korisnika dodeljenih bilo kom snadbevaču iz X.

Primer 1:

Razmatramo $\{0, 3, 4, 10\}$ četiri tačke u $\mathbb{R}$ koje odgovaraju i skupu korisnika i snadbevača. Euklidska razdaljina između korisnika i i potencijalnog snadbevača j je označena sa c_{ij} i to su elementi matrice C. Neka je $p = 2$ broj snabdevača koje treba uspostaviti. Dakle, matrica C izgleda ovako:

$$C = \begin{pmatrix} 0 & 3 & 4 & 10 \\ 3 & 0 & 1 & 7 \\ 4 & 1 & 0 & 6 \\ 10 & 7 & 6 & 0 \end{pmatrix}$$

U ovom slučaju postoji 6 mogućih izbora za $\mathrm{X}(\binom{4}{2} = 6)$. Lako se uočava da pet od šest mogućih rezultata (tabela 2.1) daju da su 3 korisnika pridružena jednom snabdevaču i jedan drugom snabdevaču, što daje vrednost funkcije cilja $3 - 1 = 2$, a preostala mogućnost (da se odabere (2, 3) tj. da se snabdevači lociraju u tačkama 3 i 4) daje vrednost funkcije cilja 0, pošto oba snabdevača prihvataju po dva korisnika. Rešenje je savršeno izbalansirano.

Tabela 2.1 Prikaz pridruživanja snabdevača korisnicima za sva moguća rešenja problema datog u primeru

X	Kom snadbevaču su pridruženi:				Broj pridruženih korisnika	Funkcija cilja
(1,2)	1	2	2	2	(1,3)	2
(1,3)	1	3	3	3	(1,3)	2
(1,4)	1	1	1	4	(3,1)	2
(2,3)	2	2	3	3	(2,2)	0
(2,4)	2	2	2	4	(3,1)	2
(3,4)	3	3	3	4	(3,1)	2

Primer 1 daje optimalnu vrednost 0 jer postoji rešenje koje daje isti broj korisnika svakom od uspostavljenih snabdevača. U sledećem primeru takva alokacija ne postoji.

Primer 2:

Razmatrajmo slučaj $n = m = 3$ i matricu troškova C:

$$C = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}.$$

Bez obrira na broj p, svi korisnici će biti dodeljeni istom snabdevaču (onom sa najmanjim indeksom). Ako je $p \geq 2$ jedan snabdevač prihvata sve korisnike, a ostali snabdevači nijednog. Dakle optimalna vrednost za rešenje ovog problema je 3 (tj. broj korisnika).

Ovaj primer 2 se lako može proširiti na bilo koji broj korisnika i potencijalnih snabdevača da se pokaže da optimalno rešenje može biti ekstremno neizbalansirano.

2.2. Matematička formulacija problema

Ova formulacija koristi sledeće promenljive: u i l, gde je:

u = maksimum broja korisnika dodeljenih nekoj uspostavljenom snabdevaču

l = minimum broja korisnika dodeljenih nekom uspostavljenom snabdevaču

Koristimo 2 skupa promenljivih za lociranje snabdevača i pridruživanje korisnika uspostavljenim snabdevačima:

$$y_j = \begin{cases} 1, ako\ je\ snabdeva\check{c}\ j\ uspostavljen \\ 0, ina\check{c}e \end{cases}, j \epsilon B$$

$$x_{ij} = \begin{cases} 1, ako\ je\ korisnik\ i\ dodeljen\ snabdeva\check{c}u\ j \\ 0, ina\check{c}e \end{cases}, i \epsilon A, j \epsilon B$$

$$\min \quad u - l \qquad (1)$$

$$\sum_{j=1}^{m} y_j = p \qquad (2)$$

$$x_{ij} \leq y_j \quad \forall i \epsilon A, j \epsilon B \qquad (3)$$

$$\sum_{j=1}^{m} x_{ij} = 1 \quad \forall i \epsilon A \qquad (4)$$

$$u \geq \sum_{i=1}^{n} x_{ij} \quad \forall j \epsilon B \qquad (5)$$

$$l \leq \sum_{i=1}^{n} x_{ij} + n(1 - y_j) \quad \forall j \epsilon B \qquad (6)$$

$$\sum_{a \in B} c_{ia} x_{ia} + (M_i - c_{ij}) y_j \leq M_i \quad \forall i \epsilon A, j \epsilon B \qquad (7)$$

$$M_i = \max_{j \in B}\{c_{ij}\}, \forall i \epsilon A$$

$$y_j \in \{0, 1\} \quad \forall j \epsilon B \qquad (8)$$

$$x_{ij} \in \{0, 1\} \quad \forall i \epsilon A, j \epsilon B \qquad (9)$$

Cilj je minimizovati razliku između u i l (1). Uslov (2) omogućava da je broj uspostavljenih snadbevača tačno p. A kako se korisnici mogu dodeliti tačno jednom i to već uspostavljenom snadbevaču, naveden je i uslov (3) i (4). Ograničenja (5) - (6) predstavljaju gornju i donju granicu za promenljive u i l, respektivno. Svaki korisnik je dodeljen njegovom najbližem snadbevaču - ograničenje (7). Na kraju, (8) - (9) ukazuju na binarnu prirodu promenljivih y_j i x_{ij} .

2.3. Primene problema izbalansiranosti lokacija

Neke primene matematičkog modela predstavljenog u poglavlju 2.2 mogu se pronaći u oblasti dizajna teritorije (videti [Kal05]), gde mala geografska područja moraju biti grupisana u veće geografske klastere prema nekim kriterijumima planiranja (na primer političke izborne jedinice, sakupljanje čvrstog otpada, školski okrug). Druga primena je lokacija antena za mobilne telefone . U principu, predstavljeni model će biti od koristi kada se apsolutna rastojanja između korisnika i snabdevača mogu zanemariti, ali klijenti neće prihvatiti drugog provajdera od onog koji im je najbliži (škole, jedinice za glasanje itd.) ili klijenti moraju da se dodele svom najbližem provajderu iz tehničkih razloga (slučaj antena). Može se primetiti da samo relativne razdaljine imaju smisla u modelu. Cilj je postići da korisnici budu što je moguće ravnomernije raspoređeni po uspostavljenim snabdevačima.

2.4. Rešavanje problema LOBA primenom genetskih algoritama

Genetski algoritmi sprovode proces evolucije skupa jedinki koje predstavljaju moguća rešenja nekoga problema. Skup rešenja se po uzoru na evoluciju u prirodi naziva i populacija. Jedinka je obično predstavljena nizom simbola dekadnog jezika, načešće bitova (ukoliko je odabran binarni prikaz rešenja), što podseća na strukturu DNK koja predstavlja genetski kôd svih živih bića. Proces pretraživanja odvija se iterativnom primenom genetskih operatora, kao što su operatori selekcije, ukrštanja i mutacije, na članove populacije. Način primene operatora zavisi od ocene kvaliteta pojedinih rešenja, koja se još naziva i prilagođenost ili dobrota (engl. *fitness value*), dok se ocenjivanje kvaliteta jedinke proverava primenom funkcije prilagođenosti ili funkcije dobrote (engl. *fitness function*). Veliki broj problema može se rešavati korišćenjem koncepta klasične genetske paradigme, razvijene od strane Hollanda.

Prilikom rešavanja problema minimizacije tj. $\min\ \{f(x) | x \in X, X \subseteq S\}$, prost genetski algoritam se može prikazati na sledeći način [Cve96]:

Inicijalizacija

Definisati prostor kodiranih rešenja X' i funkciju prilagođenosti $F(x)$.

Izabrati početnu populaciju $P_1 = \{x_1^1, \dots, x_N^1\} \subseteq X'$.

$$f^* = \min\{ f\left(d(x_i^1)\right), \quad i = 1, \dots, N \}, \quad x^* = arg\, f^*$$

Iterativni korak

Za $n = 1, 2 \ldots$

Odrediti prilagođenost $F(x_i^n)$ za svaku tačku $x_i^n, i = 1, \ldots, N$ iz trenutne populacije P_n.

Generisati $P_{n+1} = \{x_1^{n+1}, \ldots, x_N^{n+1}\} \subseteq X'$ slučajnim delovanjem genetskih operatora na slučajno izabrane tačke iz P_n koje imaju bolju prilagođenost.

$f_{min} = \min \ \{f(d(x_i^{n+1})),\ i = 1, \ldots, N\}$ Ako je $f_{min} < f^*$, tada je $f^* = f_{min}$ i $x^* = argf_{min}$

Kraj

Ako je ispunjen kriterijum zaustavljanja, staje se, a x^* se uzima za aproksimaciju optimalnog rešenja.

Ipak, složeniji problemi zahtevaju korišćenje poboljšanih verzija GA. Razvijeniji koncepti GA uključuju različite vrste kodiranja i funkcija prilagođenosti, a razvijen je i širok spektar različitih varijanti genetskih operatora selekcije, ukrštanja i mutacije koji se koriste u zavisnosti od karakteristika rešavanog problema.

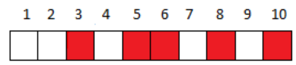
Za implementaciju genetskog algoritma u ovom radu je korišćenja open source biblioteka funkcija GAFramework, koju su implementirali dr Miroslav Marić i Srđan Božović u računarskoj laboratoriji Matematičkog fakulteta u Beogradu.

2.4.1. Definisanje prostora kodiranih rešenja

Definisanje prostora kodiranih rešenja X' zavisi u opštem slučaju od konkretnog oblika problema koji se rešava. Način kodiranja dopustivih rešenja, kao i azbuka simbola nad kojom se ono vrši trebalo bi da omoguće da se, sa što manjim brojem simbola i što prirodnije, izraze glavne karakteristike ovih rešenja.

U osnovnoj verziji GA korisiti se tzv. binarno kodiranje u kome se svakoj tački iz X dodeljuje kôd unapred zadate dužine nad binarnom azbukom $\{0,1\}$. Pri tome vrednosti simbola u kôdu, kao i njihove pozicije, zavise od tačke čiji je to kôd. Ovakav način kodiranja se prirodno nameće kada se problem koji se rešava formuliše kao zadatak matematičkog programiranja sa binarnim promenljivim i takvi problemi se najčešće veoma usupešno rešavaju genetskim algoritmima, što je slučaj i kod problema LOBA. Za neke probleme dobro su se pokazala i kodiranja nad azbukama veće kardinalnosti, o čemu se može naći i u radu [Fie95a] i [Fie95b]. Odabranom načinu kodiranja treba prilagoditi i genetske operatore. O korišćenju tradicionalnih genetskih operatora i za neka ne-binarna kodiranja razmatrano je i u radu [Bud96].

Kodiranje jedinki je, za problem LOBA, je binarno, gde se svakom rešenju dodeljuje kôd unapred zadate dužine m, gde je m broj potencijalnih snabdevača. Jedinica na i-tom mestu podrazumeva da je i-ti snabdevač odabran da bude uspostavljen. Na slici 2.1 je prikazan primer za $m = 10$, crveno su označeni bitovi koji su jedinice, a belo nule. U primeru su alocirani snabdevači sa rednim brojevima 3, 5, 6, 8 i 10.



Slika 2.1 Primer kodiranja jednike

Za GA je najpogodnije uspostaviti bijektivnu vezu između rešenja problema i njihovih genetskih kodova. Međutim ukoliko to nije moguće, kodiranje ne mora biti bijektivno (šta više ne mora biti preslikavanje). Tada postoji mogućnost da se tokom izvršavanja GA generišu kodovi koji ne odgovaraju nijednom rešenju, tzv. nekorektne jedinke. Nekorektne jedinke mogu se izbaciti iz populacije postavljajući im vrednost funkcije prilagođenosti na nulu.

Za problem LOBA korektni kodovi jedinki su oni koji sadrže tačno p jedinica, s obzirom da treba uspostaviti tačno p snabdevača. Kako se primenom genetskih operetora mogu dobiti i neke nekorektne jedinke, nakon svake promene, ukoliko broj jedinica nije p vrši se popravka genetskog koda. Tako, ukoliko je broj jedinica $p+1$, popravka se vrši izmenom proizvoljnog bita iz 1 u 0 ukoliko je broj jedinica. Analogno za ostale slučajeve.

2.4.2. Definisanje početne populacije

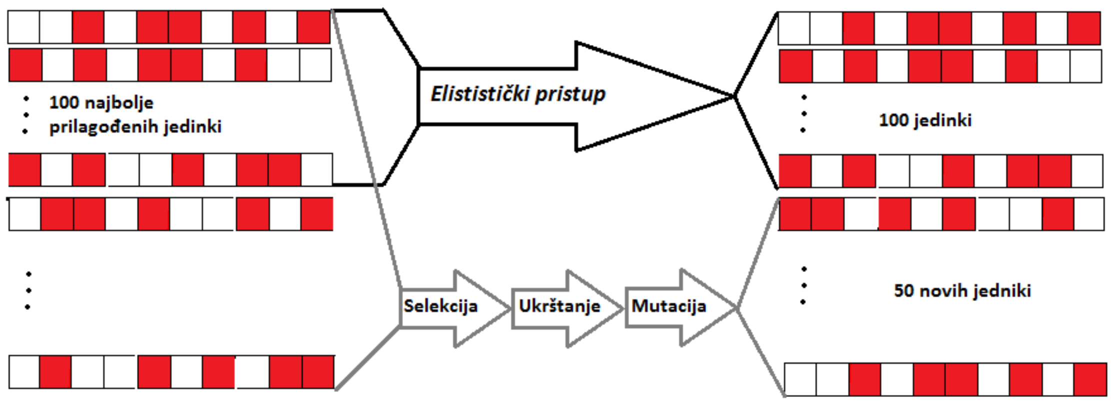
Početna populacija se može u celosti dobiti na slučajan način ili pomoću neke specijalne heuristike za rešavanje razmatranog optimizacionog problema. Da bi se obezbedila mogućnost GA da ispita različite delove prostora rešenja X', početna populacija treba da sadrži kodove što raznovrsnijih rešenja a to se najčešće postiže tako što se populacija generiše na slučajan način. Nekada je pogodno koristiti i heuristike za generisanje dela (ili cele) početne populacije, ali onda ona mora biti dobro izabrana i da je vreme izvršavanja relativno kratko. Za potrebe implementacije problema LOBA početna populacija je slučajno generisana, sa verovatnoćom p/m da vrednost bita bude 1 (inače 0).

Broj članova populacije bi trebalo da bude dovoljno veliki da omogući kreiranje ovakve jedne raznovrsne populacije, ali ne isuviše veliki, jer bi u tom slučaju njeno generisanje zahtevalo mnogo vremena. U implemtaciji genetskog algoritma za problem LOBA se dobro pokazala veličina populacije od 150 jedinki.

Jedan od bitnih aspekata prilikom implementacije GA je i način na koji se vrši zamena generacija za koju se najčešće koristi neki od sledećih pristupa:

- Generacijski: u svakoj generaciji menjati sve jedinke u populaciji.
- Stacionarni: u svakoj generaciji generisati samo deo populacije, dok se preostale jedinke prenose iz prethodne populacije.
- Elitistički: u svaku generaciju propustiti određen broj elitnih jedinki na koje se ne primenjuju genetski operatori.

U implementaciji za problem LOBA se primenjuje stacionarni genetski algoritam sa elitističkom strategijom. Na slici 2.2 je shematski prikazan korišćeni pristup. Dakle, trećina populacije se zamenjuje u svakoj generaciji, i oni nastaju primenom operatora selekcije, ukrštanja i mutacije na sve jedinke iz prethodne populacije. Dve trećine tj. 100 najbolje prilagođenih jedinki se direktno propusta u sledeću generaciju, čime se skraćuje vreme izvršavanja algoritma i obezbeđuje čuvanje dobrih rešenja. Prednost elitističkog pristupa, pored uštede procesorskog vremena, je i to što se kvalitet rešenja može bolje predvideti.



Slika 2.2 Prikaz korišćene strategije

2.4.3. Funkcija prilagođenosti

Različiti načini računanja funkcije prilagođenosti se mogu pronaći u literaturi. Odgovarajući način ili kombinovanje više njih se određuju na osnovu specifičnosti samog problema koji se rešava. Kako GA ne zahteva neprekidnost i glatkost funkcije prilagođenosti, on se može primeniti i u slučajevima kada nisu ispunjeni uslovi za primenu nekih klasičnih metoda.

Funkcija prilagođenosti, u slučaju problema LOBA, je konceptirana tako da one jedinke koje imaju najbolju vrednost funkcije cilja (za problem LOBA to je najmanja vrednost, poželjno 0) slika u 1, a one koji imaju najlošiju vrednost funkcije cilja slika u 0. Skaliranje u jedinični interval podrazumeva da funkcija prilagođenosti uzima vrednosti iz intervala [0,1].

Nije poželjno u populaciji imati dve identične jedinke. To se reguliše, tako što se ukoliko se neka jedinka već pojavila, svakoj takvoj se vrednost funkcije cilja postavlja na nulu i na taj način se smanjuje verovatnoća prelaska takve jedinke u narednu populaciju. Na isti način se postupa i ukoliko se u populaciji javlja više od 10 jedinki sa istom vrednošću funkcije cilja.

2.4.4. Operator selekcije

Ovaj operator bira jedinke iz trenutne populacije koje će ostaviti potomke za sledeću generaciju i obično se definiše tako da one jedinke, koje imaju bolju prilagođenost, imaju i veće šanse.

Pristup koji je najjednostavniji i za shvatanje i za implementaciju je da se selekcijom biraju jedinke sa najvećom vrednošću funkcije prilagođenosti, nije uvek najbolji izbor jer često može da uzrokuje završavanje algoritma u lokalnom optimumu. Kako i loše prilagođene jedinke mogu imati neke dobre gene, ne treba u potpunosti eliminisati već im omogućiti šansu da rekombinacijom njihovih dobrih gena sa drugim jedinkama proizvedu dobro prilagođene jedinke.

Metod koji se češće koristi je selekcija koja je zasnovana na slučajnosti, ali jedinke sa većom vrednošću funkcije prilagođenosti imaju veće šanse da budu izabrane. Implementira se korišćenjem ruleta, na kom svaka jedinka dobija svoj odsečak veličine koja je propocionalna vrednosti funkcije prilagođenosti za tu jedinku. Nedostatak proste rulet selekcije je mogućnost preuranjene konvergencije usled postepenog preovlađivanja visoko prilagođenih jedinki u populaciji koje ne odgovaraju globalnom optimumu. Ipak, često se koristi i kombinacija prethodno navedena dva metoda, prema kojem određeni broj najboljih jedinki garantovano prelazi u narednu generaciju zarad ubrzavanja algoritma, dok se ostale jedinke biraju ruletnom selekcijom.

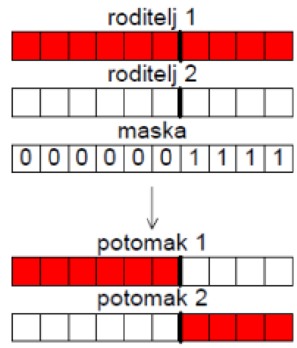
Jedna od popularnijih tehnika je i turnirska selekcija, kod koje se populacija na slučajan način deli u grupe od po n jedinki, koje se zatim “nadmeću” radi preživljavanja i prelaska u narednu generaciju [Fil98]. Broj jedinki koje učestvuju na turniru je *veličina turnira* (u oznaci n), koja predstavlja parametar turnirske selekcije (koji se najčešće unapred zadaje). Ponekad je problem odabrati odgovarajući broj n, te postoji i metod tzv. fino gradirane turnirske selekcije (FGTS), koji omogućava da u proseku taj broj ne bude ceo. Više o fino gradiranoj turnirskoj selekciji, primeni ali i poređenju u praksi sa drugim operatorima selekcije dato je u radovima V. Filipovića [Fil00], [Fil01] i [Fil03].

U genetskom algoritmu implementiranom za problem LOBA korišćena je fino gradirana turnirska selekcija sa parametrom 5.4. Što znači da je u 60% slučajeva veličina turnira 5, a inače 6.

2.4.5. Ukrštanje

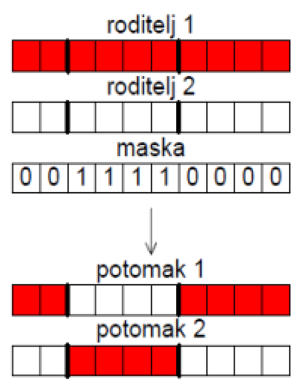
Operator ukrštanja se po ugledu na istoimeni genetski proces, definiše u najopštijem slučaju kao postupak u kome se slučajno uzajamno razmenjuju delovi kodova dva rešenja (roditelja) i tako dobijaju kodovi 2 nova rešenja (potomka). Razmena genetskog materijala jedinki-roditelja može se sprovesti pomoću jednopozicionog, dvopozicionog, višepozicionog ili uniformnog ukrštanja, ali postoje i drugi, složeniji, vidovi ovog genetskog operatora.

U osnovnoj verziji GA se primenjuje tzv. *jednopozicioni operator ukrštanja*: Na slučajan način se bira broj $k, k \in \{0, 1, \ldots, N-1\}$, koji predstavlja tzv. tačku ukrštanja tj. poziciju u kodovima roditelja u odnosu na koju će razmeniti sadržaj. Zatim se simboli u ova dva koda, od pozicije $k+1$ do poslednje pozicije N, uzajamno razmenjuju mesta (slika 2.3).



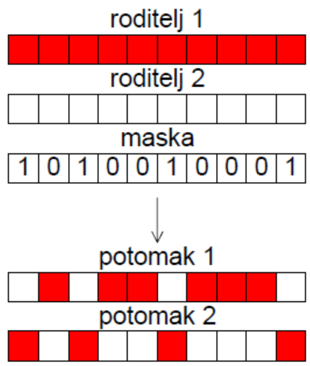
Slika 2.3 Jednopoziciono ukrštanje

Operator ukrštanja može biti definisan i u odnosu na 2 tačke ukrštanja. Kod *dvopozicionog ukrštanja* slučajno se biraju dve tačke ukrštanja $k_1, k_2 \in \{0, 1, \ldots, N-1\}$ i uzajamno se razmenjuju delovi kodova roditelja od pozicije $k_1 + 1$ do pozicije k_2. Npr. za $k_1 = 2$, a $k_2 = 6$ $(N = 10)$ razmena izgleda kao na slici 2.4.



Slika 2.4 Dvopoziciono ukrštanje

Operator *uniformnog ukrštanja* za svaki roditeljski par na slučajan način generiše binarni niz iste dužine kao genetski kod jedinki, tzv. "masku". Jedinke-roditelji razmenjuju gene na svim pozicijama na kojima maska ima vrednost 1, dok na mestima gde maska uzima vrednost 0 roditelji zadržavaju svoje gene kao što je prikazano na slici 2.5.



Slika 2.5 Uniformno ukrštanje

Koji operator ukrštanja koristimo zavisi od samog problema koji rešavamo, pa tako ukoliko ako su geni međusobno zavisni, treba se opredeliti za jednopoziciono ukrštanje koje će delimično sačuvati strukturu genetskog koda. Na suprot tome, kada su geni nezavisni među sobom najbolje je koristiti unifomno ukrštanje.

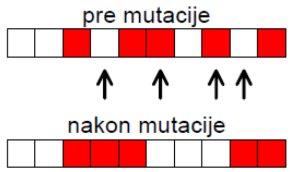
Za rešavanje problema LOBA najbolje se pokazao operator dvopozicionog ukrštanja sa verovatnoćom ukrštanja 0,85. Dakle, 85% roditalja razmenjuje genetski materijal pod dejstvom navedenog načina ukrštanja, a u ostalim slučajevima odabrani roditelji nepromenjeni prelaze u sledeću generaciju.

2.4.6. Operator mutacije

Operator mutacije je unarni operator koji se koristi da bi se povremeno unela raznovrsnost među jedinkama jedne populacije (posebno u slučaju kada su ove jedinke veoma slične), kao i da bi se sprečio neopravdan gubitak pojedinih simbola na nekim od pozicija kodova populacije. Tokom rada GA može se očekivati da će, usled uzajamnog ukrštanja kodova jedne generacije, doći do „konvergencije“ u okviru trenutne populacije, tj. da će iz iteracije u iteraciju doći do sve veće i veće sličnosti između njenih kodova. To znači da vremenom dolazi do grupisanja populacije u jednoj uskoj oblasti prostora X'. Takva populacija može, usled forsiranja jedinki koje su bolje prilagođene, sadržati kodove visoko kvalitetnih rešenja i lokalizovati oblast koja verovatno sadrži kôd nekog lokalnog minimuma problema.

Korišćenjem samo selekcije i ukrštanja, moglo bi doći do prerane konvergencije, tj. do preranog izjednačavanja kodova jedne populacije, čime bi se pretraživanje moglo „zaglaviti“ u nekoj oblasti od X' koja ne sadrži kôd globalnog minimuma. Da bi se donekle sprečila ovakva prerana konvergencija koristi se mutacija (sa malom verovatnoćom p_m) koja omogućava povratak raznolikosti u populaciju, a time i dalje rasejavanje po prostoru X'.

Operator mutacije u opštem slučaju vrši, kao i odgovarajući genetski proces, promenu sadržaja koda (hromozoma) nekog rešenja (jedinke) slučajnom zamenom pojedinih simbola ovog koda sa nekim drugim simbolima iz azbuke simbola. Ovakav operator se primenjuje na svaki potomak a unapred zadatom verovatnoćom mutacije. Mutacija, u slučaju binarno kodiranih rešenja, podrazumeva promenu jednog bita iz 1 u 0, i obrnuto (slika 2.6).



Slika 2.6 Primer mutacije

Primenom selekcije i ukrštanja, može se desiti da na određenoj poziciji sve (ili skoro sve) jedinke u populaciji imaju istu fiksiranu vrednost bita, čime je prostor pretrage smanjen. Strategija koja se može primeniti je da se mutacija u slučaju takvih tzv. "zaleđenih" bitova vrši sa verovatnoćom nekoliko puta većom nego što je to slučaj za ostale bitove [Sta07a]. Na taj način se vraćaju delovi prostora pretrage koji su bili izgubljeni usled zaleđenog bita.

Za implementaciju GA za rešavani problem LOBA dobro se pokazala mutacija sa zaleđenim bitovima. Verovatnoća mutacije je $p_{mut} = 0.4/m$ (gde je m dužina binarnog koda svake jedinke). Posebno, u slučaju zaleđenih bitova, verovatnoća mutacije se množi dodatno sa odabranim faktorom 3.5.

2.4.7. Kriterijum zaustavljanja

Iterativno se smenjuju generacije jedinki u GA sve dok se ne ispuni kriterijum zaustavljanja. Mogu se koristiti različiti kriterijumi da bi se prekinulo izvršavanje koraka GA, a najčešće se koriste:

- dostignut maksimalni, unapred zadati, broj generacija,
- sličnost jedinki u populaciji,
- ponavljanje najbolje jedinke maksimalni, unapred zadati, broj puta,
- dostizanje optimalnog rešenja (ako je ono unapred poznato),
- dokazana optimalnost najbolje jedinke (ukoliko je to moguće),
- ograničeno vreme izvršavanja genetskog algoritma i
- prekid od strane korisnika.

U različitim implementacijama koriste se različiti kriterijumi, a najčešće njihovo kombinovanje čime se smanjuje mogućnost da se loše proceni kada treba prekinuti GA.

Za potrebe rešavanja problema LOBA korišćeni su kriterijumi:

- dostignuti maksimalni broj generacija: 10000

- ponavljanje najbolje vrednosti funkcije cilja u 5000 uzastopnih generacija.

2.5. Rešavanje problema LOBA primenom VNS-a

Kao što je napomenuto u prvom poglavlju, metoda promenljivih okolina se bazira na principu lokalnog pretraživanja, tako što se pri svakoj iteraciji može vršiti prestrukturiranje okoline trenutnog rešenja. Polazna ideja VNS-a je da se u trenutku kada se dođe do nekog lokalnog optimuma izvrši slučajno pomeranje u tekućoj okolini do nekog rešenja (bez obzira i da je to rešenje loše), pa se iz tog rešenja lokalnim pretraživanjem pokuša pronalaženje nekog drugog lokalnog ekstrmuma. Na taj način se ovim pomeranjem, do rešenja koje se nalazi relativno daleko od trenutnog, postiže se sistematično pretraživanje prostora rešenja i sprečava konvergencija metode ka lošijem lokalnom ekstremumu. U slučajevima kada pomeranje nije dovelo do boljeg rešenja, zadržavanje u trenutno najboljem rešenju smanjuje mogućnost nepotrebnog širenja pretraživanja na nove oblasti prostora dopustivih rešenja.

2.5.1. Osnovni koncepti VNS-a

Neka je $N_k(k = 1, \ldots, k_{max})$ je konačan skup prethodno određenih okolina, a $N_k(x)$ skup rešenja koji se nalaze u k-oj okolini rešenja x. Kako se navodi u [Han01a] većina heuristika, koje koriste lokalnu pretragu, koristi jednu okolinu, tj. $k_{max} = 1$. U VNS-u se u fazi mešanja (*Shaking*) izlazi i u k-tu okolinu od polazne tačke x, gde k može biti veće od 1. Zatim se vrši lokalna pretraga (*Local search*) u okolini tako nađenog rešenja. Nakon čega, u zavisnosti od toga da li je rešenje pronađeno lokalnom pretrgom bolje od aktuelnog optimumam, ili se vrši ostatak algoritma sa tom novom početnom tačkom, ili se povećava k i pretražuju udaljenije okoline prethodne tačke x (*Move or not*).

Prema Hansenu i Mladenoviću [Han01a], osnovni koraci VNS-a su sledeći:

Podrazumeva se da je odabrano inicijalno rešenje x, okoline N_k kao i kriterijum zaustavljanja.

Dakle, dok se ne ispuni kriterijum zaustavljanja vrše se sledeći koraci:

1) k se postavlja na 1 (što znači pretragu prve okoline)
2) sve dok k ne postane k_{max} ponavljaju se sledeći koraci:
    - Mešanje: Generiše se tačka $x^{'}$ iz k-te okoline rešenje x (tj. $x^{'} \in N_k(x)$)
    - Lokalna pretraga: Primenjuje se metod lokalne pretrage koji počinje od $x^{'}$ kao inicijalnog rešenja i dolazi do lokalnog optimuma $x^{''}$

- Eventualno pomeranje: Ukoliko je upravo pronađeni lokalni optimum x'' bolji od aktuelnog optimuma, pretraga se pomera i nastavlja od tačke x'' tako što se pretražuje N_1 (k se postavlja na 1). U suprotnom, samo se k uvećava za 1.

Kriterijum za zaustavljanje može biti dostignuto najveće zadato vreme, maksimalni broj iteracija ili maksimalni broj iteracija između dva poboljšanja. Kao i kod genetskog algoritma, i ovde se mogu koristiti različiti kriterijumi zaustavljanja. U implementaciji korišćen je uslov unapred zadatog vremena i maksimalnog broja generacija. Parametar k_{max} je podešavan kao parameter prilikom poziva programa, a najbolje se pokazalo računanje k_{max} po formuli $\min\ \{p,\ m-p\}$, gde je p broj snabdevača koji treba uspostaviti, a $m-p$ broj preostalih snabdevača.

2.5.2. Kodiranje prostora rešenja problema LOBA za VNS

Rešenje je predstavljeno u obliku permutacije brojeva skupa $\{1, 2, \dots, m\}$ potencijalnih snabdevača, tako da prvih p u nizu su upravo oni koji su odabrani. Tako npr. za $m = 10$ i $p = 4$ kodiranje jednog rešenja može biti kao na slici 2.7. Rešenje koje je prikazano na slici podrazumeva da su odabrani snabdevači upravo 1, 2, 3 i 4.

Slika 2.7 Primer kodiranja za VNS

Princip na kom funkcioniše VNS podrazumeva da lokalni optimum često daje neke informacije o globalnom optimumu. Na primer, može biti više promenljivih sa istim vrednostima u oba rešenja. Međutim, obično se ne zna koje vrednosti su takve. Vrši se pretraga okolina ovog lokalnog optimimu je sve dok se ne nađe bolji.

Koncept okoline je važan prilikom implementacije VNS-a. Tako se, u skladu sa ovakvim kodiranjem rešenja problema LOBA, u okolini nekog rešenja nalazi rešenje dobijeno tako što se proizvoljni uspostavljeni (odabrani) snadbevač izbaci iz tog skupa tako što zameni mesto sa proizvoljnim snabdevačem koji do tada nije bio uspostavljen. Na primer: u okolini rešenja sa slike 2.7 nalazi se i rešenje dobijeno kada snabdevači 2 i 7 zamene mesta u nizu kojim je rešenje predstavljeno (slika 2.8).

Slika 2.8 Primer kodiranja rešenja u okolini

Na taj način, rešenja koja se razlikuju po jednom alociranom snabdevaču se nalaze jedno drugom u N_1 oklini. Tako, rešenje x sa slike 2.7. se pripada $N_1(y)$, gde je y rešenje sa slike 2.8. Slično, okolina N_2 podrazumeva da su u promenjena dva alocirana snabdevača itd.

2.5.3. Uprošćeni VNS

Jedno poboljšanje bazničnog VNS-a je implementirano i u slučaju rešavanja problema LOBA. Uprošćeni VNS (Reduced VNS – RVNS) ima za cilj poboljšanje efikasnosti. Najčešće ono što crpi najviše vremena u bazičnom VNS algoritmu je lokalna pretraga koja se koristi. Zato se u RVNS-u sve dok kriterijum zaustavljanja ne postane ispunjen ponavlja sledeći postupak: Za slučajno odabranu tačku rezultata bira se najbolja tačka da one zamene mesta, ukoliko je rezultat poboljšan novo rešenje se pamti.

Koraci od kojih se sastoji algoritam uprošćenog VNS-a su sledeći:

- Podrazumeva se da je odabrano inicijalno rešenje x, okoline N_k kao i kriterijum zaustavljanja.
- Dok se ne ispuni kriterijum zaustavljanja vrše se sledeća 2 koraka:
  1) k se postavlja na 1 (što znači pretragu prve okoline)
  2) sve dok k ne postane k_{max} ponavljaju se sledeći koraci:
     - *Mešanje*: Generiše se tačka x' iz k-te okoline rešenje x (tj. $x' \in N_k(x)$)
     - *Eventualno pomeranje*: Ukoliko je upravo pronađeni lokalni optimum x' bolji od aktuelnog optimuma, pretraga se pomera i nastavlja od tačke x' tako što se pretražuje N_1 (k se postavlja na 1). U suprotnom, samo se k uvećava za 1.

Kriterijumi zaustavljanja koji su korišćeni za implementaciju VNS-a za problem LOBA su:

- Isteklo ukupno vreme
- Izvršen broj RVNS iteracija koji je dat kao parameter
- Izvršen broj VNS iteracija koji je dat kao parameter.

Inicijalno rešenje od koga kreće RVNS se bira slučajno (proizvoljna permutacija brojeva od 1 do m) i RVNS se izvršava unapred zadat broj iteracija. Nakon toga kreće klasičan VNS algoritam za koji se kao početno rešenje x uzima ovo dobijeno primenom RVNS-a.

2.6. Predložena hibridizacija GA i VNS-a

Dodatna pogodnost prilikom rada sa heuristikama je i mogućnost njihove hibridizacije čime se mogu koristiti dobre strane svake od korišćenih heuristika. Nakon uspešnosti primene prethodno navedenih metoda, genetskog algoritma i VNS-a, na instance problema LOBA, sledio i pokušaj njihove hibridizacije.

Umesto faze uprošćenog VNS-a, za dobijanje početnih rešenja za glavni deo algoritma VNS-a korišćen je upravo implementirani GA. Dizajnirano je tako da se rešenje dobijeno genetskim samo prekodira u odgovarajuće rešenje kodirano za VNS. Vreme dolaska do rešenja se računa kao zbir vremena potrebnog i genetskom algoritmu i VNS-u. Kriterijumi zaustavljanja su podrazumevali zaustavljanje svakog od ova dva metoda pojedinačno. Unapred je određen broj generacija GA koji se izvršava, kao i broj iteracija VNS-a, a dodatni kriterijum zaustavljanja je i ukupno vreme izvršavanja.

2.7. Eksperimentalni rezultati

Testiranje predloženih algoritama je izvršeno na PC računaru sa Intel Core i7-860 procesorom sa radnim taktom 2.8 GHz pod Windows 7 Professional operativnim sistemom. Algoritmi su implementirani na C# programskom jeziku.

Test instance:

1. Test instance dizajnirane za problem LOBA i predstavljene u [Mar11]. U ovim instancama elementi matrice troškova postavljeni su tako da je $c_{ij} = j$, $i \in A$, $j \in B$. Da bi se postiglo da instance budu različitog nivoa težine za iste A, B i matrice troškova C, autor fiksira prvi parameter pl i onda bira slučajno izabrane pl parove u $A \times B$ (i_1, j_1) i (i_1, j_2), i na kraju menja $c_{i_1 j_1}$ i $c_{i_1 j_2}$. Ispostavlja se da niske vrednosti parametra pl proizvode komplikovane instance sa manjim optimalnim vrednostima, dok ako je pl veliko, dobijaju se manje optimalne vrednosti. Ukupno 58 instanci su generisane kombinovanjem različitih vrednosti za $m(20, 30, 50)$, $n(20, 30, 50, 100)$, $p(3, 4, 6, 10)$ i parameter $pl(10, 30, 50, 100, 200, 400)$. Dobijene instane su označene sa $m - n - p - pl$. Npr., $50 - 100 - 6 - 100$ znači da imamo 50 potencijalnih snadbevača, 100 korisnika, $p = 6$ snadbevača treba da bude alocirano i $pl = 100$ parova alokacijskih parova koje su izmenjene u matrici C.
2. Instance $G1 - G8$ kreirali su Galvão and ReVelle [Gal96]. One odgovaraju mrežama i slučajno su generisane. Instance $G1 - G4$ uključuju $n = 100$ čvorova, a instance $5 - G8$ $n = 150$. Autori su izračunali rastojanja između čvorova pomoću Floyd-Warshall-ovog algoritma vodeći računa o dužinama slučajno generisanih lukova. Elementi matrice rastojanja date u $G1 - G8$ su uzeti kao elementi matrice troškova C za problem LOBA. Maksimum prekrivajućeg rastojanja i zahtevi za čvorovima su bili ignorisani iz skupa podataka $G1 - G8$. Dimenzije (n, p) ovih instanci su: $(100, 3)$, $(100, 5)$, $(100, 7)$, $(100, 10)$, $(150, 5)$, $(150, 7)$, $(150, 10)$ i $(150, 15)$.
3. Skup instance koji su kreirali Lorena i Senne i koristili u [Lor03] i [Lor04] za problem p-medijane ograničenih kapaciteta. Ovi podaci se odnose na realne podatke iz grada São José dos Campos u Brazilu (http://www.lac.inpe.br/lorena/instancias.html). Ove instance su označene sa: $SJC1$, $SJC2$, $SJC3a$, $SJC3b$, $SJC4a$, $SJC4b$. Za date vrednosti za n: 100, 200, 300 i 402 korišćene su sledeći skupovi vrednosti za p: $\{3, 5, 7, 10\}$, $\{5, 10, 15, 20\}$, $\{10, 15, 20, 30\}$ i $\{10, 20, 30, 40\}$, redom. Skup korisnika je jednak skupu mogućih lokacija, tj. $A = B$ $(m = n)$. Zanemaruju se zahtevi i kapaciteti kod ovih instanci i koristi se Euklidska razdaljina između čvorova kao elemenata matrice troškova, tj. $c_{ij} = d(i, j)$.

Verifikacija korektnosti optimizacionih algoritama je važan aspekt. Proveru je moguće uraditi za manje dimenzije instance, za koje su poznata optimalna rešenja. Optimalnost rešenja je verifikovana uz pomoć programa za programski paket IBM ILOG CPLEX ([Cplex]) i tom prilikom je korišćena formulacija matematičkog modela navedenog u poglavlju 2.2. ovog rada, a implementacija urađena na programskom jeziku C. Optimalna rešenja dobijena CPLEX-om su saglasna i sa rezultatima koje je Marin dobio u radu [Mar11], ali su u tom radu razmatrane samo instance manjih dimenzija (do 50 potencijalnih snabdevača i do 100 korisnika). Međutim, provera se za instance većih dimenzija ne može izvršiti zbog ne posedovanja optimalnih rešenja, ali značajno je istaći to da se za većinu instanci velikih dimanzija rezultati dobijeni korišćenjem implementiranih različitih heuristika poklapaju.

Rešavanje problema datog kroz navedenu matematičku formulaciju korišćenjem CPLEX-a, u poređenju sa rezultatima genetskog algoritma (paralelizovana i neparalelizovana varijanta), je prikazano u tabeli 2.2.

Rešenje je traženo u 20 iteracija za svakoj instanci, a za veće dimenzije u 5 iteracija.

U prvoj koloni tabele 2.2 data su imena testiranih instanci (koja sadrže podatke o dimenzijama instance). Druga kolona sadrži optimalno rešenje tekuće instance dobijeno korišćenjem CPLEX-a tj. ako je optimalno rešenje poznato. U suprotnom, "--" je upisano u odgovarajuće polje. Vreme (u sekundama) za koje je CPLEX došao do rešenja se nalazi u trećoj koloni. Najbolje rešenje dobijeno GA dato je u sledećoj koloni. Prosečno vreme (u sekundama) potrebno algoritmu da prvi put dobije najbolju vrednost je dato u *T* koloni, dok T_{ukupno} predstavlja ukupno vreme izvršavanja GA za svih 10000 generacija.

U daljem tekstu koriste se iste formule i oznake kao u radu [Sta07b]. Kvalitet GA rešenja u svih 20 izvršavanja se računa kao srednje procentualno odstupanje $agap = \frac{1}{20}\sum_{i=1}^{20} gap_i$, gde je gap_i procentualno odstupanje dobijenog rešenja u *i*-tom izvršavanju algoritma (i = 1, 2, . . . , 20). Pri tome se $gap_i = 100 * \frac{sol_i - Opt.sol}{Opt.sol}$ računa kao procentualno odstupanje u odnosu na unapred poznato optimalno rešenje *Opt.reš,* ili u odnosu na $GA_{najb.}$ $gap_i = 100 * \frac{sol_i - Best.sol}{Best.sol}$, ukoliko optimalno rešenje za datu instancu nije poznato (sol_i označava GA rešenje dobijeno u *i*-tom izvršavanju algoritma). Standardna devijacija odstupanja $\sigma = \sqrt{\frac{1}{20}\sum_{i=1}^{20}\left(gap_i - agap\right)^2}$ predstavljena je u koloni *sigma*.

Tabela 2.2 Rezultati dobijeni primenom implementiranog genetskog algoritma za instance “datos”

Instance	**Opt**	**T**	**GAbest**	**T**	**T ukupno**	**N gen**	**Agap**	**Sigma**
datos20-20-3-1	1	3.68800	1	0.00	8.99	10000	0	0
datos20-20-3-10	15	3.54700	15	0.04	10.77	10000	0.333333	1.452966
datos20-20-3-50	3	2.27400	3	0.01	7.92	10000	0	0
datos20-20-6-1	1	3.31800	1	0.00	7.64	10000	0	0
datos20-20-6-10	12	1.63100	12	0.01	6.18	10000	0	0
datos20-20-6-50	6	3.16900	6	0.01	6.09	10000	0	0
datos30-30-3-1	0	4.94700	0	0.00	7.85	10000	0	0
datos30-30-3-10	25	38.37600	25	0.01	8.43	10000	0	0
datos30-30-3-30	22	34.01900	22	0.16	7.21	10000	0	0
datos30-30-3-100	9	35.34000	9	0.04	7.45	10000	0	0
datos30-30-4-1	1	30.93500	1	0.00	8.17	10000	0	0
datos30-30-4-10	24	32.64300	24	0.01	8.30	10000	0	0
datos30-30-4-30	17	43.39400	17	0.02	7.31	10000	0	0
datos30-30-4-100	9	74.43300	9	0.54	8.05	10000	1.111111	3.333333
datos30-50-3-10	45	46.64500	45	0.01	10.98	10000	0	0
datos30-50-3-50	34	56.43600	34	0.01	8.62	10000	0	0
datos30-50-3-100	27	79.01000	27	0.01	10.10	10000	0	0
datos30-50-3-200	13	102.22000	13	0.02	9.34	10000	0	0
datos30-50-6-10	45	19.89700	45	0.77	9.03	10000	1.888889	0.793492
datos30-50-6-50	34	9.36600	34	0.03	8.78	10000	0	0
datos30-50-6-100	19	52.99200	19	0.10	12.92	10000	0	0
datos30-50-6-200	10	17.15400	10	1.11	10.56	10000	0	0
datos30-50-10-10	45	6.72800	45	0.09	11.44	10000	0	0
datos30-50-10-50	27	8.03300	27	0.04	9.42	10000	0	0
datos30-50-10-100	18	40.57700	18	0.79	9.57	10000	0	0
datos30-50-10-200	5	44.41200	5	0.99	9.52	10000	0	0
datos30-100-3-10	94	48.34000	94	0.06	10.47	10000	0	0

datos30-100-3-50	85	400.76000	85	0.08	9.60	10000	0	0
datos30-100-3-100	75	412.05600	75	0.01	11.48	10000	0	0
datos30-100-3-200	63	315.69400	63	0.43	16.02	10000	0	0
datos30-100-6-10	92	48.74900	92	1.55	12.36	10000	0.108696	0.326087
datos30-100-6-50	84	874.66700	84	0.04	11.27	10000	0	0
datos30-100-6-100	64	401.65600	64	0.65	14.98	10000	0.3125	1.362156
datos30-100-6-200	48	411.90600	48	0.30	12.36	10000	0.104167	0.454052
datos30-100-10-10	96	32.90200	96	0.01	14.00	10000	0	0
datos30-100-10-50	77	139.70700	77	0.09	13.00	10000	0	0
datos30-100-10-100	63	129.80000	63	0.25	12.85	10000	0	0
datos30-100-10-200	41	75.51300	41	0.06	13.65	10000	0	0
datos50-50-3-10	46	148.62500	46	0.01	10.42	10000	0	0
datos50-50-3-50	39	423.93800	39	0.61	10.91	10000	0	0
datos50-50-3-100	32	463.12500	32	0.02	11.26	10000	0	0
datos50-50-3-400	11	370.74400	11	0.68	14.05	10000	9.545454	13.008261
datos50-50-6-10	43	233.78300	43	2.17	11.44	10000	1.395349	2.790698
datos50-50-6-50	33	3264.00800	33	0.70	13.28	10000	0.151515	0.660439
datos50-50-6-100	24	7926.62600	24	0.57	16.69	10000	0	0
datos50-50-6-400	3	1376.23200	3	0.46	13.60	10000	6.666667	22.607767
datos50-50-10-10	43	223.64200	43	1.16	14.74	10000	1.395349	1.860465
datos50-50-10-50	30	238.75100	30	0.28	11.81	10000	0.166667	0.726483
datos50-50-10-100	20	441.99600	20	1.47	15.14	10000	13	5.567764
datos50-50-10-400	2	1802.68300	2	1.52	14.00	10000	7.5	17.853571
datos50-100-3-10	96	1416.07700	96	0.01	12.51	10000	0	0
datos50-100-3-50	87	1065.98900	87	0.07	13.81	10000	0.114943	0.344828
datos50-100-3-100	79	1126.60100	79	0.02	12.72	10000	0	0
datos50-100-3-400	48	1749.60600	48	0.02	11.57	10000	0	0
datos50-100-6-10	95	1665.54400	95	0.02	12.62	10000	0	0
datos50-100-6-50	83	13273.55600	83	1.18	16.64	10000	0.301205	0.923476

datos50-100-6-100	70	110415.50600	70	0.17	21.46	10000	0	0
datos50-100-6-400	35	7592.40600	35	0.31	15.35	10000	0	0
datos50-100-10-10	96	1056.70300	96	0.02	16.77	10000	0	0
datos50-100-10-50	83	31282.51000	83	0.20	15.85	10000	0	0
datos50-100-10-100	69	11380.78000	69	0.58	20.48	10000	0.434782	1.304348
datos50-100-10-400	18	18080.49500	18	0.77	20.99	10000	1.666667	3.557291
datos100-100-3-1	--	--	1	0.00	16.53	10000	0	0
datos100-100-3-10	96	19296.37200	96	0.16	25.82	10000	0	0
datos100-100-3-50	--	--	91	0.04	17.20	10000	0	0
datos100-100-3-100	--	--	86	1.18	21.72	10000	0	0
datos100-100-3-1000	--	--	41	0.67	27.79	10000	0	0
datos100-100-3-2000	--	--	11	0.22	24.78	10000	0	0
datos100-100-3-2500	--	--	3	1.99	29.50	10000	0	0
datos100-100-3-3000	--	--	1	0.49	29.65	10000	190	43.588989
datos100-100-3-3500	--	--	1	0.12	20.49	10000	5	21.794495
datos100-100-3-5000	--	--	1	0.01	18.44	10000	0	0

Prikazani rezultati su dobijeni korišćenjem neparalizovane verzije GA. Primenom paralelizovane varijante se postižu potpuno isti rezultati, a poređenje vremena potrebnog za izračunavanje ilustrovano je na grafikonu na slici 2.7. Može se primetiti da paralelizovana verzija dolazi do rešenja i do nekoliko puta brže.

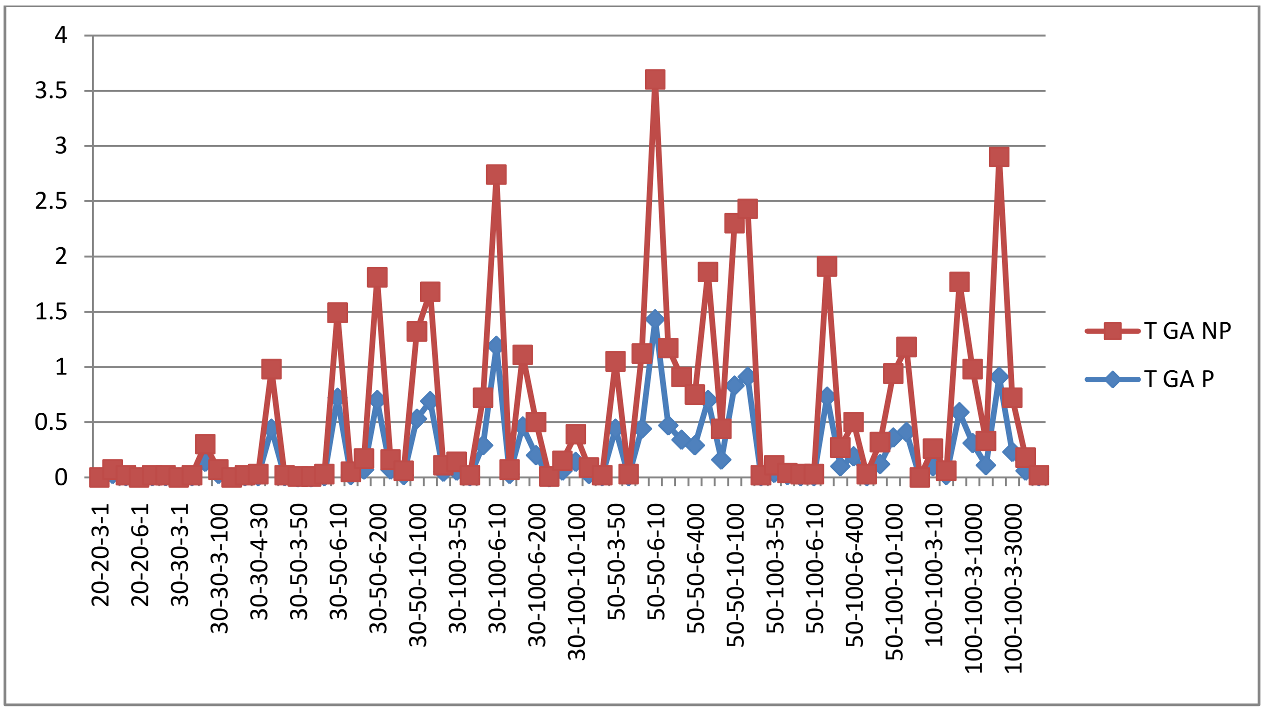
Može se konstatovati da su sve metode dostigle optimalna rešenja (za one instance za koje se raspolagalo rešenjima dobijenim CPLEX-om, pa su poznata optimalna rešenja). Kada se porede prosečna vremena potrebna za izračunavanja rezultata za te instance koje je rešio i CPLEX poređenja se mogu prikazati tabelom 2.3.

Tabela 2.3 Uporedni prikaz prosečnih vremena potrebnih za izračunavanje instance

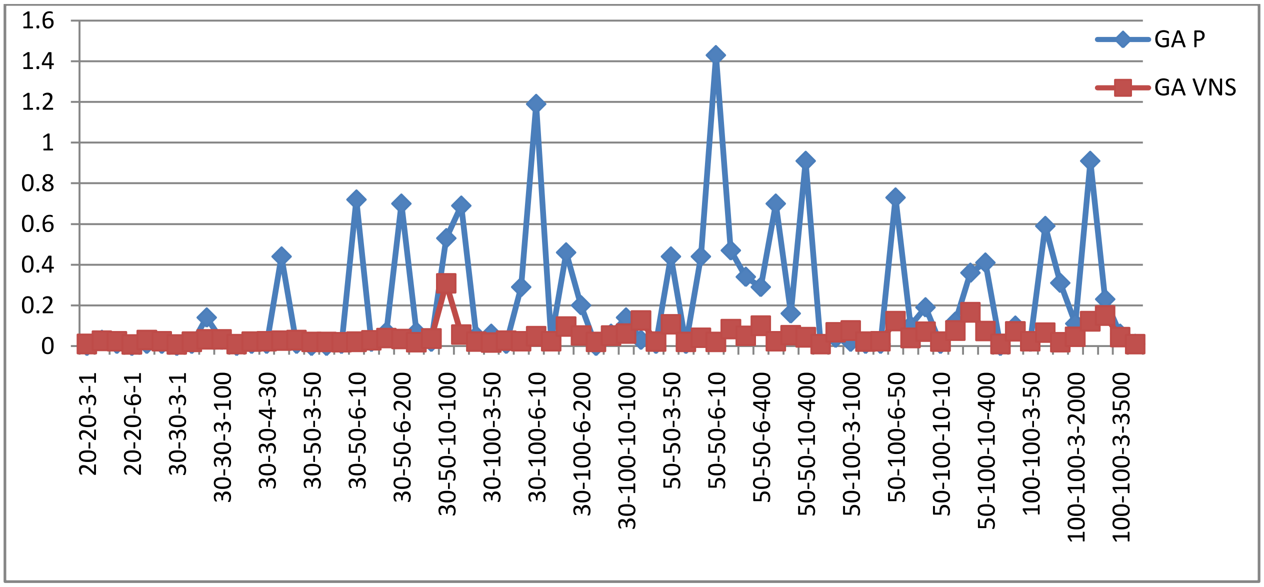
Metoda	**Prosečno vreme u sekundama**
CPLEX	3566.24
Genetski algoritam – neparalelizovana varijanta	0.34
Genetski algoritam – paralelizovana varijanta	0.22

VNS	0.76
Hibridizacija VNS i GA	0.05

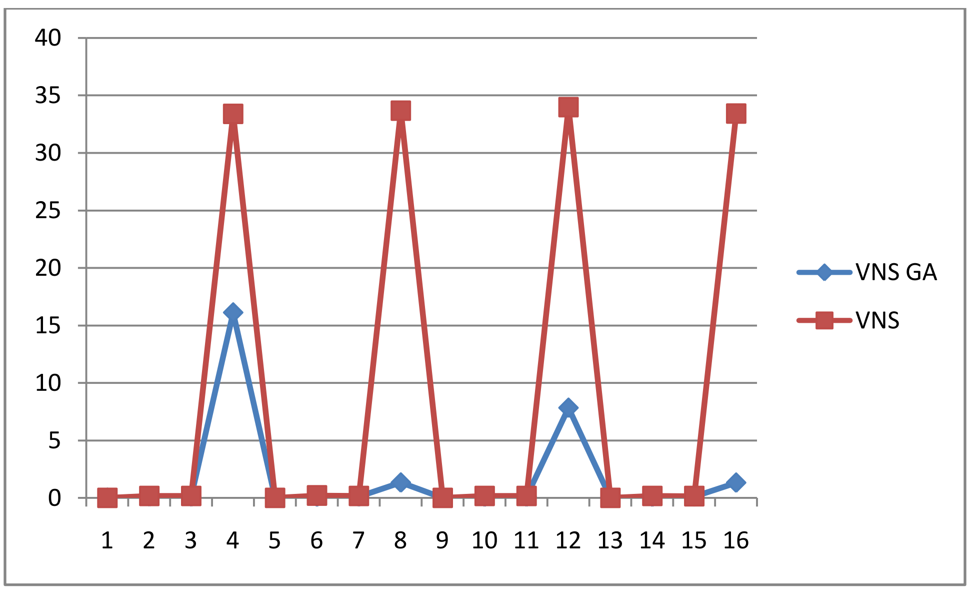
Rezultate nad instancama „datos“ najbrže je dala implementacija hibridizacije VNS-a i GA. Vreme potrebno za izračunavanje na ovaj način uporedno je prikazano sa vremenima vezanim za paralelizovanu verziju GA na grafikonu (slika 2.8). Na ordinati, odnosno y-osi, je označeno vreme koje je izraženo u sekundama.



Slika 2.7 Poređenje vremena u sekundama potrebnim za izračunavanje na instancama „datos“ korišćenjem paralelizovane varijante GA (plavo) i neparalelizovane (crveno)

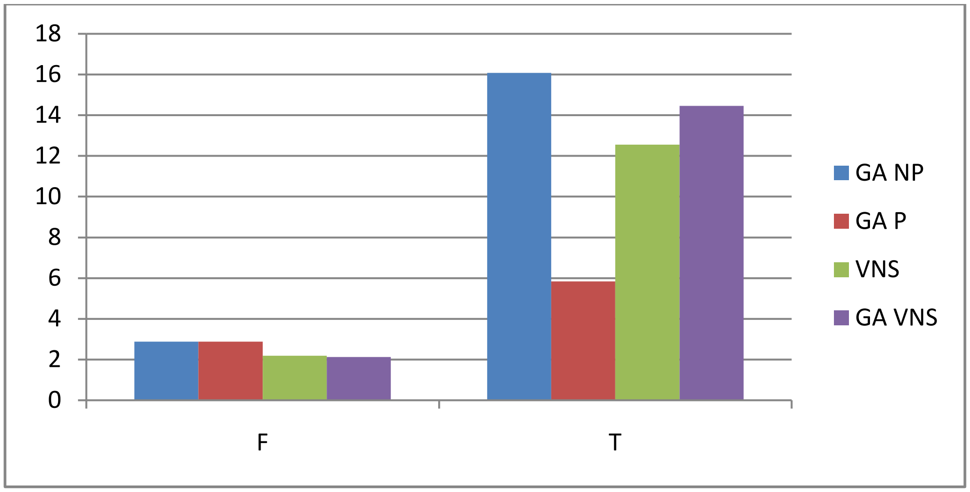


Slika 2.8 Poređenje vremena u sekundama potrebnim za izračunavanje na instancama „datos" korišćenjem paralelizovane varijante GA (plavo) i hibridizacije VNS-a i GA (crveno)



Slika 2.9 Poređenje vremena u sekundama potrebnim za izračunavanje na instancama „Galvao" dimenzija $n = m = 100$ korišćenjem implementirane hibridizacije VNS-a i GA (plavo) i VNS-a (crveno)

Na slici 2.9 su prikazan grafik koliko je za koju od „Galvao" instanci koje su sve istih dimenzija, ali za različite vrednosti za p. Može se primetiti da su instance 4, 8, 12 i 16 zahtevale znatno više vremena za izračunavanje pomoću VNS-a. Zajednično tim instancama je da je $p = 10$. Vidi se da je to vreme kod hibridizacije VNS-a sa GA smanjeno. Važno je napomenuti da su obe verzije dale iste rezultat. Rezultati se poklapaju i sa rezultatima dobijenim pomoću GA, osim na pomenutim spornim instancama, gde su VNS kao i hibrid dali bolje rešenje (najbolje do sada poznato prema dostupim podacima).



Slika 2.10 Poređenje vremena u sekundama potrebnim za izračunavanje na instancama „SCJ“ dimenzija $m = n$ iz skupa $\{100, 200, 300, 402\}$ korišćenjem implementiranih heuristika

Ostaje razmatranje rezultata dobijenih za instance najvećih dimenzija, za koje ne postoje poznata optimalna rešenja. Na slici 2.10 su prikazane prosečne vrednosti funkcije cilja dobijene sa sve četiri heuristike, kao i prosečna vremena izračunavanja. Može se primetiti da je rešenja najbrže dao paralelizovani GA, koji je, kao što je i očekivano, davao identična rešenja kao i neparalelizovani GA. Nešto duže vreme izračunavanja od paralelizovane varijnte GA (ali ne i neparalelitovane) imao je VNS, ali i nešto bolja rešenja. Još precizniji, ali malo i sporiji je bio hibrid VNS-a sa GA. Važno je napomenuti da su sve metode dale dobra rešenja, a detaljnije tabele zainteresovani mogu naći na internet adresi: http://www.math.rs/~nina/master.

3. PROBLEM MAKSIMIZACIJE MINIMALNOG RASTOJANJA (MMDP)

Problem maksimizacije minimalnog rastojanja (*Max-Min Diversity Problem* - MMDP) se sastoji od odabira podskupa elemenata datog skupa na način da najmanje rastojanje među odabranim elementima bude maksimizovano. Problem je NP-težak i može se formulisati kao problem celobrojnog linearnog prograiranja [Res10].

Od 1980-tih, nekoliko metoda za rešavanje ovog problema su razvijeni i primenjeni u različitim oblastima, pogotovu socialnih i bioloških nauka. U radu [Res10] predstavljen je heuristički metod za pronalaženje aproksimativnog rešenja, koji se, za ovaj problem, pokazao bolje nego prethodno korišćene metaheuristike (tabu pretraživanje i simulirano kaljenje).

3.1. Opis i matematička formulacija MMDP problema

MMDP se sastoji iz odabira podskupa M od m elemenata¸ ($|M| = m$) od skupa N od n elemenata tako da je minimalna distanca između odabranih elemenata maksimizovana. Definicija distance između elemenata se može prilagoditi različitim primenama. Kao što je istaknuto u [Kuo93] i [Glo98], MMDP ima primene u raspoređivanju postrojenja, socialnim problemima, kao u ekologiji, itd. U većini ovih primena, podrazumeva se da se svaki element može prestaviti skupom atributa. Neka je s_{ik} stranje ili vrednost k-og atributa elementa i, gde je $k = 1, \dots, K$. Rastojanje između elemenata i i j se može definisati na sledeći način:

$$d_{ij} = \sqrt{\sum_{k=1}^{K} \left(s_{ik} - s_{jk}\right)^2}.$$

U ovom slučaju d_{ij} je prosto euklidsko rastojanje između i i j. Ta rastojanja se koriste da se formuliše MMDP kao problem binarnog celobrojnog kvadratnog programiranja, gde za $i = 1, \dots, n$, promenljva x_i ima vrednost 1 ako je element i odabran, a 0 u suprotnom:

$$(MMDP) \quad \max \quad z_{MM}(x) = \min_{i<j} d_{ij} x_i x_j$$

$$\sum_{i=1}^{n} x_i = m$$

$$x_i = \{0, 1\}, \quad i = 1, \dots, n \quad .$$

Da je MMDP NP-težak problem nezavisno su pokazali Erkut u radu [Erk90] i Ghosh u radu [Gho96]. Kako se navodi u korišćenoj reference [Res10] i za egzaktne i za heurističke metode teže je rešiti MMDP nego MDP.

3.2. Rešavanje problema MMDP primenom genetskih algoritama i VNS-a

Za rešavanje problema MMDP korišćene su iste tehnike kao i za rešavanje problema LOBA, uz odgovarajuće modifikacije.

3.3. Eksperimentalni rezultati

Izračunavanja za ovaj problem su izvršena na računarima istih performansi kao što je već navedeno kod problema LOBA.

Svi eksperimenti izvršeni su na tri skupa instanci:

1) Glover: Ovaj skup podataka se sastoji od 75 matrica za koje se vrednosti elemenata računaju kao Euklidska rastojanja od slučajno generisane tačke sa koordinatama u opsegu od 0 do 100. Broj koordinata za svaku tačku je takođe slučajno generisani između 2 i 21. Glover je 1998. godine razvio generator test problema i konstruisao instance gde je $n = 10$, 15 i 30. Vrednost za m kreće se od $0.2n$ do $0.8n$.
2) Geo: Ovaj skup podataka se sastoji od 60 matrica konstruisanih istim generatorom kao i za Glover skup instanci. Korišćene vrednosti za n su 100, 250, i 500. Za svaku vrednost n posmatrani su $m = 0.1n$, $0.3n$ (generisano je deset instanci za svaku kombinaciju n i m). Ove instance su slične geometrijskim instancama predstavljenim u radu [Erk90].
3) Ran: Ovaj skup podataka se sastoji od 60 matrica sa slučajnim brojevima. Ove instance su zasnovane na generatoru koji je uveo Silva [Sil04]. Što se tiče skupa Geo, generisano je dvadeset instanci gde je $n = 100$, 250, i 500 (i za svaku vrednost n posmatrano je $m = 0.1n$, $0.3n$). Slučajni celi brojevi su generisani između 50 i 100 u svim instancama osim u slučaju instance za koju su $n = 500$ i $m = 150$ i u kojoj su generisani između 1 i 200 (da bi bile teže u smislu komparacije između heuristika).

Tabela 3.1. Rezultati dobijeni primenom implementiranog genetskog algoritma za instance "Glover"

Ime instance	M	Vrednost	T	GA best	T	T u	NG	Agap	sigma
Glover (n 10) 1	2	243.972519	0.02900	243.972519	0.00	4.33	10000	0	0
Glover (n 10) 2	2	210.650421	0.03100	210.650421	0.00	5.71	14500	0	0
Glover (n 10) 3	2	146.615362	0.03000	146.615362	0.00	20.74	55001	0	0
Glover (n 10) 4	2	186.181030	0.02900	186.181030	0.00	6.10	14500	0	0
Glover (n 10) 5	2	209.615695	0.03200	209.615695	0.00	9.21	23500	0	0
Glover (n 10) 1	3	184.358268	0.03300	184.358268	0.00	22.32	55001	0	0
Glover (n 10) 2	3	183.786091	0.03400	183.786091	0.00	20.90	55002	0	0
Glover (n 10) 3	3	114.428698	0.03100	114.428698	0.00	13.27	37003	0	0
Glover (n 10) 4	3	167.251792	0.03200	167.251792	0.00	12.72	32500	0	0
Glover (n 10) 5	3	189.864461	0.03300	189.864461	0.00	14.58	37001	0	0
Glover (n 10) 1	4	175.631873	0.03400	175.631873	0.00	13.36	37002	0	0
Glover (n 10) 2	4	158.069369	0.03100	158.069369	0.00	14.85	41506	0	0
Glover (n 10) 3	4	107.222914	0.03400	107.222914	0.01	14.78	41513	0	0
Glover (n 10) 4	4	137.020925	0.03700	137.020925	0.00	10.05	28000	0	0
Glover (n 10) 5	4	179.709603	0.04500	179.709603	0.00	26.58	73003	0	0
Glover (n 10) 1	6	162.996330	0.04200	162.996330	0.00	21.61	46002	0	0
Glover (n 10) 2	6	136.578528	0.04600	136.578528	0.00	19.71	46003	0	0
Glover (n 10) 3	6	71.920340	0.05000	71.920340	0.00	6.17	14500	0	0
Glover (n 10) 4	6	120.472147	0.04300	120.472147	0.00	24.33	59501	0	0
Glover (n 10) 5	6	159.057904	0.05200	159.057904	0.01	32.84	77519	0	0
Glover (n 10) 1	8	131.151473	0.02700	131.151473	0.00	5.73	10000	0	0
Glover (n 10) 2	8	118.751394	0.02700	118.751394	0.00	5.19	10000	0	0
Glover (n 10) 3	8	63.308742	0.03200	63.308742	0.00	5.48	10000,	0	0
Glover (n 10) 4	8	92.219577	0.03800	92.219577	0.00	5.54	10000	0	0
Glover (n 10) 5	8	140.081703	0.02900	140.081703	0.00	9.99	19000	0	0
Glover (n 15) 1	3	100.407694	0.05900	100.407694	0.00	33.08	77510	0	0
Glover (n 15) 2	3	218.186632	0.05300	218.186632	0.01	37.55	95514	0	0

Glover (n 15) 3	3	198.451001	0.05500	198.451001	0.00	25.83	64010	0	0
Glover (n 15) 4	3	167.324931	0.05100	167.324931	0.00	33.25	77504	0	0
Glover (n 15) 5	3	177.222450	0.05900	177.222450	0.00	14.45	37001	0	0
Glover (n 15) 1	4	83.800980	0.06000	83.800980	0.01	38.21	95531	0	0
Glover (n 15) 2	4	206.568539	0.06500	206.568539	0.00	33.69	86508	0	0
Glover (n 15) 3	4	181.238579	0.06200	181.238579	0.00	33.90	82008	0	0
Glover (n 15) 4	4	139.045673	0.06200	139.045673	0.38	42.98	96372	0	0
Glover (n 15) 5	4	175.203773	0.06100	175.203773	0.00	25.14	64011	0	0
Glover (n 15) 1	6	64.572966	0.12000	64.572966	0.01	34.33	91014	0	0
Glover (n 15) 2	6	177.909809	0.11000	177.909809	0.01	32.76	86512	0	0
Glover (n 15) 3	6	162.419629	0.12000	162.419629	0.02	37.71	100039	0	0
Glover (n 15) 4	6	126.890982	0.10700	126.890982	0.01	37.43	95527	0	0
Glover (n 15) 5	6	142.199090	0.12800	142.199090	0.04	37.10	100107	0	0
Glover (n 15) 1	9	40.395094	0.10700	40.395094	0.04	39.19	91008	0	0
Glover (n 15) 2	9	154.423984	0.14000	154.423984	0.04	40.30	100096	0	0
Glover (n 15) 3	9	152.134061	0.11800	152.134061	0.02	39.01	100054	0	0
Glover (n 15) 4	9	105.526041	0.11300	105.526041	0.00	26.83	68502	0	0
Glover (n 15) 5	9	122.405470	0.09000	122.405470	0.00	30.42	77504	0	0
Glover (n 15) 1	12	29.376416	0.07000	29.376416	0.00	15.51	28000	0	0
Glover (n 15) 2	12	138.775401	0.06000	138.775401	0.00	33.56	73002	0	0
Glover (n 15) 3	12	126.498024	0.05200	126.498024	0.00	7.14	14500	0	0
Glover (n 15) 4	12	97.713203	0.05300	97.713203	0.00	23.67	41501	0	0
Glover (n 15) 5	12	107.655698	0.05400	107.655698	0.00	19.41	41501	0	0
Glover (n 30) 1	6	185.559034	0.24300	185.559034	3.40	45.20	108098	0	0
Glover (n 30) 2	6	180.303513	0.25700	180.303513	0.26	43.00	100622	0	0
Glover (n 30) 3	6	164.014971	0.17700	164.014971	0.09	44.81	100223	0	0
Glover (n 30) 4	6	105.166692	0.22800	105.166692	0.35	39.42	100751	0	0
Glover (n 30) 5	6	104.243271	0.25800	104.243271	0. 97	41.54	102231	0	0
Glover (n 30) 1	9	165.742968	1.12800	165.742968	1.82	41.56	104354	0	0

Glover (n 30) 2	9	161.115388	1.16500	161.115388	1.23	47.16	102515	0	0
Glover (n 30) 3	9	141.890862	0.49500	141.890862	0.09	39.77	100201	0	0
Glover (n 30) 4	9	85.644500	0.98800	85.644500	0.08	39.75	100191	0	0
Glover (n 30) 5	9	90.281803	0.56600	90. 281803	5.03	50.24	112775	0	0
Glover (n 30) 1	12	158.517386	1.30200	158.517386	1.35	40.80	103397	0.105244	0.458750
Glover (n 30) 2	12	148.608580	3.58300	148.608580	0. 89	41.08	102193	0	0
Glover (n 30) 3	12	129.171662	1.29400	129.171662	0. 12	40.48	100304	0	0
Glover (n 30) 4	12	69.527749	0.69500	69.527749	3.41	42.74	108758	0	0
Glover (n 30) 5	12	73.584952	0.80800	73.584952	3.74	43.41	109339	0.078485	0.342110
Glover (n 30) 1	18	143.297627	1.62000	143.297627	2.32	43.16	105642	0	0
Glover (n 30) 2	18	136.885495	1.44400	136.885495	2.47	44.89	105985	0	0
Glover (n 30) 3	18	108.142552	1.43400	108.142552	0. 85	42.39	102043	0	0
Glover (n 30) 4	18	56.884194	1.59000	56.884194	0.07	41.38	100170	0	0
Glover (n 30) 5	18	61.311505	1.51200	61.311505	0.03	41.12	10007	0	0
Glover (n 30) 1	24	119.963448	0.33200	119.963448	0.03	52.65	100046	0	0
Glover (n 30) 2	24	124.108779	0.17900	124.108779	0.05	46.61	100094	0	0
Glover (n 30) 3	24	96.181072	0.28400	96.181072	0.05	50.32	100089	0	0
Glover (n 30) 4	24	46.736852	0.37300	46.736852	0.02	51.23	100034	0	0
Glover (n 30) 5	24	40.686627	0.41900	40.686627	0.01	53.04	100011	0	0

Kao i za prethodni problem, kako prostor i namena ovog rada ograničavaju mogućnost prikazivanja svih tabela sa rezultatima izračunavanjima na svim instancama, kompletniji rezultati dostupni su u elektronskoj formi na već navedenom linku.

4. ZAKLJUČAK

U ovom radu opisane su implementacije genetskog algoritma, metode promenljivih okolina, kao i hibridizacije te dve metaheuristike koje su posebno dizajnirane za rešavanje razmatranih problema diskretne optimizacije. Implementirani algoritmi primenjeni su na dva NP-teška problema iz ove oblasti koji imaju veliku primenu u praksi: problem izbalansiranosti lokacija (LOBA) i problem maksimizacije minimalnog rastojanja (MMDP). Data je matematička formulacija, opis problema, opis implementacije primenjenih metoda, kao i prikaz dobijenih rezultata.

Razmatrani problemi su teški za rešavanje i to je jedan od razloga što egzaktne metode mogu rešiti samo instance problema manjih dimenzija. Zato je posebna pažnja posvećena heurističkim metodama za rešavanje ovih problema. Dizajnirani su i implementirani modifikovani genetski operatori, koji vode računa o načinu kodiranja i prirodi problema. Primenjeni su specijalni načini kodiranja i funkcija cilja, dizajnirani i genetskih operatora koji odgovaraju prirodi problema, prilagođeni načinima kodiranja tako da čuvaju korektnost jedinki. Implementirana je i metoda promenljivih okolina, pri čemu je važno bilo odabrati pravo kodiranje i osmisliti koncept okolina koji je korišćen. Dobro se pokazala i hibridizacija ove dve metode koja je implementirana za oba problema razmatrana u radu. Primena ovih metoda dovodi efikasno do rešenja na testiranim instancama.

Najznačajniji rezultati i doprinosi u ovom radu su:

- prvi put su korišćene navedene heuristike za rešavanje i za problem LOBA i za problem MMDP prema opisanim formulacijama,
- hibridizacija metode pretrage okolina i genetskog algoritma,
- dobijanje rešenja za problem LOBA i za instance velikih dimenzija koje do sada nisu rešavane.

Rezultati dobijeni u ovom radu idu u prilog heurističkim pristupima rešavanju problema diskretne optimizacije. Posebno kod rešavanja NP-teških problema većih dimenzija, gde je rešavanja egzaktnim metodama za većinu realnih praktičnih problema nemoguće.

Kako je ostalo prostora za nastavak istraživanja u ovoj oblasti, potencijalna ideja za dalji rad je modifikacija implementiranih metoda za rešavanje sličnih problema. Dalje unapređenje, koje je u planu, je paralelizacija implementacije metode promenljivih okolina. A jedno od proširenja izvršenih istraživanja u ovoj oblasti bi moglo biti rešavanje razmatrana dva problema LOBA i MMDP i nekim drugim heurističkim metodama, ali i rešavanje ovih problema za instance još većih dimenzija.

5. LITERATURA

[Ary04] **Arya V., Garg N., Khandekata R., Meyerson A., Munagala K., Pandit V.,** "Local Search Heuristics for the *k*-median and Facility location problems" Society for Industrial and Applied Mathematics, *SIAM J. COMPUT.,* Vol. 33, No. 3, pp. 544–562., (2004).

[Bar97] **Barros A.I., Frenk J.B.G., Gromicho J.,** "Fractional Location Problems" *Location Science,* Vol. 5, No. 1, pp. 47-58, (1997).

[Bud96] **Budin, L., Golub, M., Budin, A.**, “Traditional Techniques of Genetic Algorithms Applied to Floating-Point Chromosome Representations”, KoREMA ‘96. - 41st Annual Conference, Opatija, pp. 93-96, (1996).

[Cha01] **Charikar M., Khuller S., Mountz D.M., Narasimhan G.,** "Algorithms for Facility Location Problems with Outliers", *Proc. 12th ACM-SIAM Sympos. Discrete Algorithms*, pp. 642-651, Washington, DC, (2001).

[CLM01] **Cruz Lopez-de-los-Mozos M., Mesa J.A**., "The maximum absolute deviation measure in location problems on networks", *European Journal of Operational Research*, Vol. 135, pp.184-194, (2001).

[Cook71] **Cook S.A,** "The Complexity of theorem proving procedures", Proceedings of the Third Annual ACM, *Symposium on Theory of Computing*, pp.151-158 ACM Press, New York, NY USA, (1971)

[Cook97] **Cook W.J., Cunningham W.H., Pulleyblank W.R., SchrijverA**., "*Combinatorial Optimization*"; John Wiley&Sons; 1 edition, November 12, (1997)

[Cor10] **Correia I., Nickel S., Saldanha-da-Gama F**., “Single-assignment hub location problems with multiple capacity levels”, *Transportation Research Part B* 44, pp.1047-1066, (2010)

[Cplex] IBM ILOG CPLEX Optimizer
http://www.ilog.com/products/cplex/

[Cve04] **Cvetković D., Hansen P., Kovačević-Vujičić V**., "On Some Interconnections Between Combinatorial Optimization and Extremal Graph Theory", *Yugoslav Journal of Operations Research*, Vol. 14, No. 2, pp.147-154, Belgrade, (2004).

[Cve96] **Cvetković D., Čangalović M., Kovačević-Vujičić V., Dugošija Đ, Simić S., Vuleta J.,** "Kombinatorna optimizacija: Matematička teorija i algoritmi", Društvo operacionih istraživača Jugoslavije, (1996).

[Čuš03] **Čuš F., Balič J**., "Optimization of cutting process by GA approach", *Robotics and Computer Integrated manufactoring,* Vol. 19, (1/2), pp.113-121, (2003).

[Dom03] **Domínguez-Marín P.**, "The discrete ordered median problem: models and solution methods", Kluwer Academic Publishers, (2003).

[Dor99] **Dorigo M., Caro G.D., Gambardella L. M**., “Ant algorithms for discrete optimization”, *Journal Artificial Life archive*, Volume 5 Issue 2, MIT Press Cambridge, MA, USA, (1999), doi: 10.1162/106454699568728

[Dor06] **Dorigo M., Birattari M., StützleT.**, "Ant Colony Optimization, Artificial Ants as a Computational Intelligence Technique", IRIDIA – Technical Report Series, Technical Report No.TR/IRIDIA/2006-023, (2006).

http://iridia.ulb.ac.be/IridiaTrSeries/IridiaTr2006-023r001.pdf

[Dow96] **Dowsland K. A**., "Genetic algorithms a tool for OR?", *Journal of Operational Research Society*, Vol. 47, pp.550-561 (1996).

[Erk90] **Erkut E**., "The discrete p-dispersion problem", *European Journal of Operational Research*, 46: pp. 48–60, (1990).

[Esp09] **Espejo I., Marín A., Puerto J., Rodríguez-Chía A.M**., "A comparison of formulations and solution methods for the minimum-envy location problem", *Computers& Operations Research,* Vol.36, pp.1966–1981, (2009).

[Fie95a] **Field P**., "A Multary Theory for Genetic Algorithms: Unifying Binary and Nonbinary Representations," PhD thesis, Queen Mary and Westfield College, London, (1995).

[Fie95b] **Field P**., "Nonbinary Transforms for Genetic Algorithm Problem", Complex Systems 9, pp.11-28 (1995). http://www.complex-systems.com/pdf/09-1-2.pdf

[Fil98] **Filipović V.,** "Predlog poboljšanja operatora turnirske selekcije kod genetskih algoritama", Magistarski rad, Univerzitet u Beogradu, Matematički fakultet, (1998).

[Fil00] **Filipović V., Kratica J., Tošić D., Ljubić I**., "Fine Grained Tournament Selection for the Simple Plant Location Problem", *Proceedings of the 5th Online World Conference on Soft Computing Methods in Industrial Applications - WSC5*, pp.152-158, (2000).

[Fil01] **Filipović V., Tošić D., Kratica J**., "Experimental Results in Applying of Fine Grained Tournament Selection", *Proceedings of the 10th Congress of Yugoslav Mathematicians,* pp. 331-336, Belgrade, 21.-24.01. (2001).

[Fil03] **Filipović V**., "Fine-Grained Tournament Selection Operator in Genetic Algorithms", *Computing and Informatics*, 22(2), pp.143-161, (2003).

[Gal96] **Galvão R.D., ReVelle, C.S.**, "A Lagrangean heuristic for the maximal covering location problem", *European Journal of Operational Research,* 88, pp. 114-23, (1996).

[Gho96] **Ghosh. J.B.,** "Computational aspects of the maximum diversity problem", *Operations Research Letters*, 19: pp. 175–181, (1996).

[Glo98] **Glover F., Kuo C.C., Dhir. K.S**., "Heuristic algorithms for the maximum diversity Problem", *Journal of Information and Optimization Sciences*, 19: pp. 109–132, (1998).

[Gol10] **Golub M**., "Genetski algoritmi", (2010). http://www.zemris.fer.hr/~golub/ga/ga.html

[Han99] **Hansen P., Mladenović N.**, "An Introduction to Variable Neighborhood Search", In: Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, Voss S., Martello S., Osman I.H., Roucairol C. (eds.), Kluwer Academic Publishers, pp. 433-458 (1999).

[Han01a] **Hansen P., Mladenović N.,** "Variable neighborhood search: Principles and applications", *European Journal of Operational Research,* Vol. 130, pp. 449-467 (2001).

[Han01b] **Hansen P., Mladenović N., Perez-Brito D**., "Variable neighborhood search: Principles and applications", *Journal of Heuristics*, 7, pp. 335-350, Kluwer Academic Publishers (2001).

[Han07] **Hansen P., Brimberg J., Urošević D., Mladenović N**., "Primal-Dual Variable Neighborhood Search for the Simple Plant-Location Problem", *INFORMS Journal on Computing*, 19, pp. 552-564 (2007).

[Hll75] **Holland J.H.**, "Adaptation in Natural and Artificial Systems", The University of Michigan Press, Ann Arbor (1975).

[Hua11] **Huang M., Smilowitz K., Balcik B**., "Models for relief routing: Equity, efficiency and efficacy", *Transportation Research Part E*, in press (2011).

[Kal05] **Kalcsics J., Nickel S., Schröder M.,** "Towards a unified territory design approach – Applications, algorithms and GIS integration", Top 13, pp.1–56. (2005).

[Kah11] **Kahl F.,** "Discrete optimziation", PhD course, Lund University, Sweden, (2011)
http://www.maths.lth.se/matematiklth/personal/fredrik/discreteoptimization/

[KH79] **Kariv O., Hakimi S.I**., "An algorithmic approach to network location problems.II: The p-median", *SIAM Journal on Applied Mathematics*, Vol.37, pp. 539-560 (1979)

[Koz92] **Koza J.R**. "Genetic Programming: On the Programming of Computers by Means of Natural Selection", MIT Press, Cambridge, MA, (1992).

[Kra06] **Kratica J., Stanimirović Z**., "Solving the uncapacitated multiple allocation p-hub center problem by genetic algorithm", *Asia- Pacific Journal of Operational Research*, Vol. 23, 4, pp.425-437, (2006).

[Kra07] **Kratica J., Stanimirović Z., Tošić D., Filipović V**., "Two genetic algorithms for solving the uncapacitated single allocation p-hub median problem", *European Journal of Operational Research,* Vol. 182, Issue 1, pp. 15-28, (2007).

[Kuo93] **Kuo C.C., Glover F., Dhir. K.S.,** "Analyzing and modeling the maximum diversity problem by zero-one programming", *Decision Sciences*, 24: pp.1171–1185, (1993).

[Law01] **Lawler E.L**., "Combinatorial optimization: networks and matroids", Courier Dover Publications, (2001).
http://books.google.com/books?id=m4MvtFenVjEC&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[Lee04] **Lee J.**, "A First Course in Combinatorial Optimization", Cambridge University Press, (2004).
http://books.google.com/books?id=3pL1B7WVYnAC&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[Lia04] **Liang Y.C., Smith A.E.**, "An ant colony optimization algorithm for the redundancy allocation problem (RAP)", IEEE Transactions on Reliability, Vol. 53, No. 3, pp.417–423, (2004).

[Lib05] **Libert L, Dražić M.**, Variable Neighbourhood Search for the Global Optimization of Constrained NLPs, Proceedings of GO, pp. 1 – 5., (2005). http://www.lix.polytechnique.fr/~liberti/vnsgo05.pdf

[Lju98] **Ljubić I., Kratica J., Filipović V**., "Primena genetskih algoritama u nalaženju minimalnog Steinerovog stabla", *Zbornik radova sa drugog međunarodnog simpozijuma iz industrijskog inženjerstva*, Mašinski fakultet, Beograd, str. 277-280, (1998).

[MM08a] **Marić M**., "Rešavanje nekih NP – teških hijerarhijsko-lokacijskih problema primenom genetskih algoritama", Doktorska disertacija, Matematički fakultet, Beograd, (2008).

[MM08b] **Marić M., Kratica J., Tuba M.**, "One Genetic Algorithm for Hierarchical Covering Location Problem", *Proceedings of the 9th WSEAS International Conference on Advanced Topics On Evolutionary Computing* (EC'08), pp. 122-126, (2008).

[MM10] **Marić M.**, "An Efficient Genetic Algorithm For Solving The Multi-Level Uncapacitated Facility Location Problem", *Computing and Informatics*, Vol. 29, pp. 183-201, (2010).

[Mar05] **Marín A**., "Formulating and solving splittable capacitated multiple allocation hub location problems", *Computers & Operations Research*, 32, pp 3093-3109, (2005).

[Mar10] **Marín A., Nickel S., Velten S**., "An extended covering model for flexible discrete and equity location problems", *European Journal of Operational Research*, Vol. 210, Issue 1, pp.125-163, (2010).

[Mar11] **Marín A**., "The discrete facility location problem with balanced allocation of customers", *European Journal of Operational Research*, Vol. 210, Issue 1, pp.27-38, (2011).

[Mes03] **Mesa J.A., Puerto J., Tamir A**., "Improved algorithms for several network location problems with equality measures", *Discrete Applied Mathematics*, 130, p.437–448, (2003).

[Mil07] **Milfelner M., Čuš F**., "Sistem za spremljanje in optimiranje procesa frezanja z uporabo Genetskih algoritmov", in Balič J., Veža I., Čuš F., "Napredne proizvodne tehnologije", Univerza v Mariboru, Sveučilište Split, pp.121-138, (2007).

[Mla95] **Mladenović N.**, "A variable neighborhood algorithm - a new metaheuristics for combinatorial optimization", *Abstracts of papers presented at Optimization days*, Montreal (1995).

[Mla97] **Mladenović N., Hansen P.** "Variable neighborhood search", *Computers Operations Research*, Vol. 24, pp. 1097-1100, (1997).

[Mla11] **Mladenović N., Urošević D.,** "Metoda promenljivih okolina za problem trgovačkog putnika sa minimalizacijom kašnjenja", SYM-OP-IS 2011, zbornik radova, pp. 349-352, (2011).

[Nic01] **Nickel S**., "Discrete Ordered Weber Problem", in Fleichmann B., Lach R. and Derigs U. eds.: Operations Research Proceedings 2000., pp. 71-76, (2001)

[Ognj04] **Ognjanović Z., Krdžavac N.,** "Uvod u teorijsko računarstvo", Fakultet organizacionih nauka, Beograd, (2004).

[Ogr00] **Ogrycsak W.,** "Inequality measures and equitable approaches to location problems", *European Journal of Operational Research*, Vol. 122, p.374-391, (2000).

[Opr92] **Opricović S.,** "Optimizacija sistema", Nauka, Beograd, (1992).

[Pan04] **Pandit V.,** "Local Search Heuristics for Facility Location Problems", Indian Institute of Technology Delhi, doktorska disertacija, (2004). http://www.cse.iitd.ernet.in/~pandit/thesis.pdf

[Rad10] **Radhakrishnan P., Gopalan M. R., Jeyanthi N**., "Design of Genetic Algorithm Based Supply Chain Inventory Optimization with Lead Time", IJCSNS *International Journal of Computer Science and Network Security,* Vol.10, No.4, (2010). http://paper.ijcsns.org/07_book/201004/20100435.pdf

[Res10] **Resende M., Martí R., Gallego M., Duarte A.**, "GRASP and Path Relinking for the Max-Min Diversity Problem", *Computers and Operations Research*, 37, pp. 498-508, (2010).

[ReV05] **ReVelle C.S., Eiselt H.A.,** "Location analysis: A synthesis and survey", *European Journal of Operational Research,* Vol. 165, p.1–19., (2005).

[ReV08] **ReVelle C.S., Eiselt H.A., Daskin M.S**., "A bibliography for some fundamental problem categories in discrete location science", *European Journal of Operational Research,* Vol.184*,* pp.817–848., (2008).

[Sch10] **Schrijver A.,** "A Course in Combinatorial Optimization", University of Amsterdam, The Netherlands, August, (2010). http://homepages.cwi.nl/~lex/files/dict.pdf

[SchA] **Schrijver A.,** "On the History of Combinatorial Optimization (till 1960)", CWI and University of Amsterdam, http://homepages.cwi.nl/~lex/files/histco.pdf

[Sil04] **Silva G.C., Ochi L.S., Martins S.L.,** “Experimental comparison of greedy randomized adaptive search procedures for the maximum diversity problem”, Lecture Notes in Computer Science, 3059:498–512, (2004).

[Sta04] **Stanimirović Z.,** "Rešavanje nekih diskretnih lokacijskih problema primenom genetskih algoritama", Magistarska teza, Matematički fakultet, Beograd, (2004).

[Sta07a] **Stanimirović Z., Kratica J., Dugošija Đ**., "Genetic Algorithms for Solving the Discrete Ordered Median Problem", *European Journal of Operational Research*, Vol. 182, 3, pp.983-1001, (2007).

[Sta07b] **Stanimirović Z.,** "Genetski algoritmi za rešavanje nekih NP-teških hab lokacijskih problema", Doktorska disertacija, Matematički fakultet, Beograd, (2007).

[Sta08] **Stanimirović Z**., "An Efficient Genetic Algorithm for the Uncapacitated Multiple Allocation p-Hub Median problem", *Control and Cybernetics,* Vol. 37, 3, pp.669-692, (2008).

[Sta10] **Stanimirović Z.**, "A Genetic Algorithm Approach For The Capacitated Single Allocation P-hub Median Problem", *Computing and Informatics*, Vol. 29, No.1, pp.117-132, (2010).

[Tal03] **Talbott W. A**., "Automatic Creation of Team-Control Plans Using an Assignment Branch in Genetic Programming", Genetic Algorithms and Genetic Programming at Stanford, (2003). http://www.genetic-programming.org/sp2003/Talbott.pdf

[T’ki02] **T’kindt V., Monmarché N., Tercinet F., Laügt D.**, "Systems An Ant Colony Optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem", European Journal of Operational Research, 142, pp.250–257, (2002)

[Tch95] **Tcha D.W., Paik C.H**., "Parametric Uncapacitated Facility Location", *International Journal of Production Research*, Vol. 33, No. 3, (1995).

[Toš04] **Tošić D**., **Mladenović N., Kratica J., Filpović V**., "Genetski algoritmi", Matematički institut SANU, Beograd, (2004).

[Top05] **Topcuoglu H., Corut F., Ermis M., Yilmaz G**., "Solving the Uncapacitated Hub Location Problem Using Genetic Algorithms", *Computers & Operations Research*, Vol. 32, pp.967-984, (2005).

[Šeš02] **Šešum V., Tošić D**., Genetic Algorithms and Smoothing Filters in Solving the Geographysical Inversion Problem, *Yugoslav Journal of Operations Research*, Vol.12, No.2, pp.215-226, Belgrade, (2002).

[Vug96] **Vugdelija M., Kratica J., Filipović V., Radojević S**., "Mogućnosti genetskih algoritama u mašinskom učenju", *XXII Jupiter konferencija, Zbornik radova*, Mašinski fakultet, Beograd, str.4.55-4.59, (1996).

[Živ00] **Živković M**., "Algoritmi", Matematički fakultet, Beograd, (2000).