SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 6852

Rješenje problema usmjeravanja vozila s vremenskim ograničenjima različitim inačicama evolucijskog algoritma

Sandra Zurak

Zagreb, lipanj 2020.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
ODBOR ZA ZAVRŠNI RAD MODULA

Zagreb, 11. ožujka 2020.

Grana: **2.09.04 umjetna inteligencija**

ZAVRŠNI ZADATAK br. 6852

Pristupnik: **Sandra Zurak (0036511685)**
Studij: Računarstvo
Modul: Računarska znanost

Zadatak: **Rješenje problema usmjeravanja vozila s vremenskim ograničenjima različitim inačicama evolucijskog algoritma**

Opis zadatka:

Opisati problem usmjeravanja vozila s vremenskim ograničenjima dostave u statičkom okruženju. Predstaviti postupak rješavanja kao optimizacijski problem uz uporabu evolucijskog algoritma. Ostvariti različite prikaze rješenja i mogućnost mijenjanja operatora križanja, mutacije i selekcije. Eksperimentalno usporediti rezultate dobivene različitim inačicama prikaza i operatora te evolucijskog algoritma. Radu priložiti izvorne tekstove programa, dobivene rezultate uz potrebna objašnjenja i korištenu literaturu.

Zadatak uručen pristupniku: 13. ožujka 2020.
Rok za predaju rada: 12. lipnja 2020.

Mentor:

Prof. dr. sc. Domagoj Jakobović

Predsjednik odbora za završni rad modula:

Doc. dr. sc. Marko Čupić

Djelovođa:

Izv. prof. dr. sc. Tomislav Hrkać

Veliko hvala obitelji, prijateljima, kolegama i mentoru na pomoći i podršci tijekom dosadašnjeg obrazovanja.

SADRŽAJ

1. Uvod

Poznati su brojni kombinatorni problemi koji se mogu proučavati, a jedan od najpoznatijih je problem usmjeravanja vozila. Njegova važnost ističe se u primijeni koja može biti vrlo raznolika. Rješenja ovog problema primjenjiva su u raznim sferama života. U osnovi je problem predstavljen na kupcima koje vozila moraju poslužiti robom koju prevoze iz skladišta. Poopćenjem ovog problema dolazi se do problema posjećivanja odredišta sa zadanog početnog mjesta s različitim ograničenjima kapaciteta. Različite verzije ovog problema prisutne su u trgovini, prometu, a sve više i u nekim modernim pogonima,

U ovom radu analizirat će se različiti prikazi rješenja ovog problema uz vremenska ograničenja (eng. VRP with Time Windows). Pri tome su korišteni razni problemi s više stotina kupaca, a rješenja su prikazana permutacijskim nizom i matričnim prikazom. Implementiran je genetski algoritam s više operatora te je analiziran njihov utjecaj na učinkovitost. Izdvojena je i najbolja kombinacija operatora za svaki prikaz. Korišten je i Clarke-Wright algoritam uštede te je uspoređen s rezultatima genetskog algoritma. Implementirano je i grafičko sučelje koje na vrlo jednostavan način vizualizira zadani problem.

2. Problem usmjeravanja vozila

2.1. Općenito o problemu

Problem usmjeravanja vozila (eng. Vehicle Routing Problem) kombinatorni je problem koji se pokušava riješiti na mnogo različitih načina, a svima je zajedničko da pokušavaju optimizirati rješenje kakao bi cijena bila što manja. Cijena nastaje kao posljedica kretanja vozila od skladišta do kupaca te povratka u skladište. Na početku se zadaje broj kupaca i vozila, a varijacije problema nastaju dodavanjem različitih parametara.

Problem usmjeravanja vozila definiran je 1959. godine, a predložili su ga dvojac Dantzig i Ramser. Tijekom godina, nastale su različite varijante ovog problema, a neki od najčešćih su problem s vremenskim ograničenjima (eng. VRP with Time Windows), VRP s mogućnošću prihvata i dostave tereta s vremenskim ograničenjima (VRP with Pick-up and Deliveries and Time Windows), problem riješen s većim brojem skladišta i mnogi drugi. U ovom radu rješava se problem usmjeravanja vozila s vremenskim ograničenjima.

2.2. Problem s vremenskim ograničenjima

Problem usmjeravanja vozila s vremenskim ograničenjima nadograđuje osnovni problem s vremenskim ograničenjem [t1, t2] tijekom kojeg vozila moraju poslužiti kupce. Da bi se problem uspješno riješio potrebno je na početku definirati:

$C = \{1,...,n\}$ - skup n kupaca

K - broj dostupnih vozila

Q - najveći mogući kapacitet vozila

B - kapacitet robe potrebne kupcu

$[t1, t2]$ - vremensko ograničenje kupca

t - trajanje posluživanja kupca (može biti različito za svakog kupca)

(x, y) - koordinate

D(x, y) - koordinate skladišta D

$\forall c \in C, \quad \exists\, c(x, y))$

Razlika problema s vremenskim ograničenjima i svih ostalih je u tome što je svaki kupac dostupan samo određeni period te se početak i kraj tog perioda definiraju pri zadavanju problema. Tada je definirano i vrijeme koje će vozilo morati provesti kod kupca da bi se posluživanje obavilo do kraja. Vozilo može posjetiti više kupaca, no svaki kupac smije biti poslužen samo jednom.

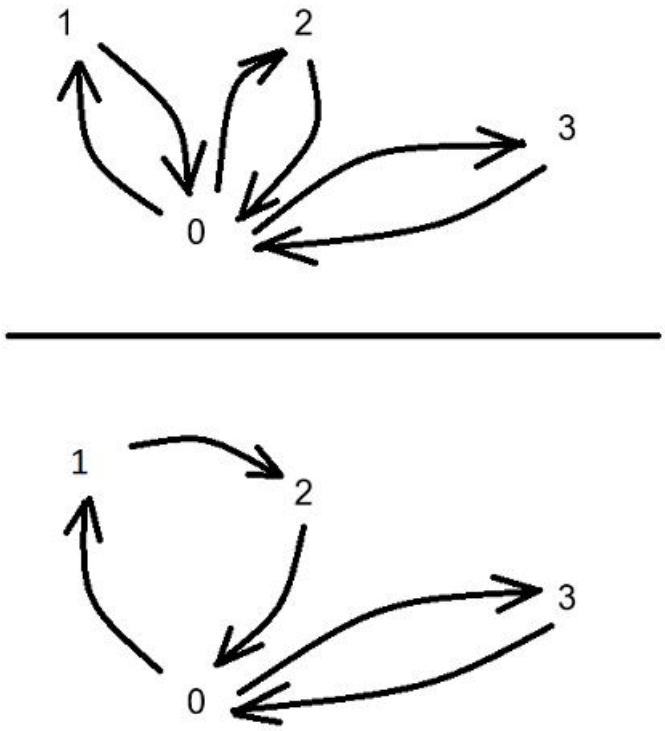
2.3. Clarke-Wright algoritam uštede

Clarke-Wright algoritam uštede pripada algoritmima s konstruktivnom heuristikom kojima se dobiva rješenje korak po korak.

Clarke i Wright u 2. polovici 20. stoljeća [6] dobili su algoritam koji se temelji na jednadžbi uštede. Ona se temelji na udaljenosti između kupaca *c(i,j)*, udaljenosti jednog kupca od skladišta *c(0, i)* te udaljenosti drugog kupca od skladišta *c(0,j)*. Koristeći te parametre dobiva se sljedeća jednadžba čiji je rezultat ušteda *S(i, j)*:

S(i, j) = c(0,i) + c(0,j) - c(i,j)

Na početku postoji onoliko ruta koliko ima kupaca te svaka ruta ide od skladišta do kupca i natrag. Potom se pokuša pronaći moguća ušteda spajanjem kupaca u parove. Tako se iz koraka u korak (slika 2.1.) pronalazi najveća moguća ušteda pri spajanju kupaca koji zatim ulaze u istu rutu. Pri tome se jedan kupac ne može naći na više ruta.



Slika 2.1: Korak Clarke-Wright algoritma

3. Evolucijski algoritmi

Evolucijski algoritmi pretražuju prostor problema, rješavaju optimizacijske probleme skupom stohastičkih metoda. Dobiveno rješenje nije nužno najbolje moguće.

Genetski algoritmi, evolucijske strategije te genetsko i evolucijsko programiranje samo su neki dijelovi velikog pojma evolucijskog algoritma. Ono što im je zajedničko je u svakom koraku pronalaženje boljih rješenja, no ne pretražuje se uvijek cijeli prostor stanja. Različitim je operatorima omogućen postepeni napredak, a nastali su po uzoru na prirodne procese koji omogućuju funkcioniranje i opstanak bića na Zemlji.

U ovom se području koriste različiti pojmovi iz biologije jer je način djelovanja isti te pojmovi imaju isti smisao. Tijekom duge povijesti Zemlje, prema teorijama, određene su jedinke selekcijama, na osnovi različitih parametra pojedinih vrsta, nastavljale evoluirati, dok su neke izumrle. U evolucijskim algoritmima također razlikujemo različite selekcije, no svima je zajednički pokušaj odabira čim bolje jedinke koja će prijeći u iduću generaciju. Na taj se način smanjuje prostor pretraživanja te evolucijski algoritmi ostvaruju svoj cilj optimizacije problema u što kraćem vremenu.

U ovome radu naglasak je na genetskom algoritmu te će on biti pobliže opisan kao primjer evolucijskog algoritma.

3.1. Genetski algoritam

Da bi genetski algoritam pronašao čim bolje rješenje potrebno je na početku odrediti broj iteracija tijekom kojih će populacija jedinki napredovati te broj jedinki u svakoj populaciji. Kako bi se odabrale jedinke koje će prijeći u iduću generaciju važno je poznavati funkciju dobrote. Ona govori koliko je neka jedinka uspješna ili neuspješna u onome što se od nje očekuje.

Već spomenuta sličnost sa prirodnim procesima vidljiva je u strukturama koje se koriste. Kao i u prirodi, jedinka se sastoji od kromosoma koji se sastoje od gena. Geni su u implementaciji oni koji nose dobrotu jednike. Koristeći različite prikaze i geni se razlikuju, stoga je pri korištenju različitih prikaza rješenja potrebno koristiti

pripadajuće operatore mutacije i selekcije.

U svakoj iteraciji selekcijom se odabiru jedinke roditelja na koje se zatim djeluje operatorom križanja koje daje jedinku djeteta. Vrste selekcije, križanja i mutacije bit će opisane u idućem poglavlju. Djeca dobivena križanjem postaju nova populacija nad kojom se zatim odvija mutacija. Nova se populacija zatim proširuje rezultatom elitizma. Elitizam čuva najbolje rješenje za novu generaciju kako se ono ne bi izgubilo. U nastavku je prikazan opisani postupak:

1. biranje roditelja selekcijom
2. križanjem roditelja dobivaju se djecu
3. od nove djece stvora se nova populacija
4. mutiranje nove populacije
5. odabir najboljeg rješenja za novu generaciju
6. nova populacija proširuje se najboljim rješenjem

4. Programsko ostvarenje

4.1. Korišteni prikazi rješenja

4.1.1. Permutacijski niz

Permutacijski niz jednostavan je prikaz koji koristi samo listu cijelih brojeva. U toj su listi brojevi koji označavaju kupce te oni koji označavaju vozila. Iza svakog broja koji označava vozilo slijede brojevi koji označavaju kupce koje će to vozilo poslužiti. Permutacijski niz s brojem vozila (slika 4.1.) prikaz je koji se sastoji od dvije liste. To je malo složeniji prikaz od samog permutacijskog niza, ali daje jasniju sliku broja kupaca koji moraju biti posluženi na jednoj ruti. Prva lista veličine je ukupnog broja kupaca, a njeni elementi prikazuju kupce. Druga lista veličine je ukupnog broja vozila koji su u dostavi korišteni za posluživanje. Svaki broj u toj listi označava koliko je kupaca poslužilo vozilo na tom indeksu u listi. Brojevi kupaca u prvoj listi poredani su ovisno o vozilima u drugoj. To znači da, ako je prvi broj u drugoj listi 3, u prvoj listi prva tri kupca bit će poslužena istim vozilom koje je označeno indeksom jedan u drugoj. Suma brojeva upisanih u drugu listu, pri inicijalizaciji, je ograničena na ukupan broj kupaca koji moraju biti posluženi.

Slika 4.1: Prikaz permutacijskim nizom s brojem vozila

4.1.2. Matrični prikaz

Matrični prikaz je prikaz koji koristi realne brojeve od 0 do 1 koji prikazuju rješenje. Matrica se sastoji od redova i stupaca. Redovi su vozila, dok su stupci kupci. Pri inicijaliziranju matrice nasumično se odabire onoliko relanih brojeva koliko ima stupaca te se svaki upisuje u nasumično odabrano mjesto u stupcu. Nepopunjena mjesta u matrici pune se nulama. Realan broj u stupcu označava da će kupac u tom stupcu biti dio rute vozila koje predstavlja redak u kojem je taj broj. U jednom retku može postojati više realnih brojeva jer svako vozilo može posjetiti više kupaca, a isto tako ni ne mora posjetiti ijednog kupca pa u cijelom redu mogu biti nule. Kada u jednom redu postoji više realnih brojeva, najmanji broj predstavlja kupca koji će prvi biti poslužen, dok će kupac s najvećim realnim brojem biti poslužen zadnji na toj ruti. Ova vrsta prikaza vidljiva je na slici 4.2.

0	0,1	0	0
0	0	0	0
0,2	0	0,5	0
0	0	0	0,4

Slika 4.2: Matrični prikaz

4.2. Računanje dobrote

U korištenoj implementaciji jedinka je bolja ako ima manju dobrotu (*fitness*). Algoritam 1 prikazuje programski kod za računanje dobrote. Vidljivo je da se svako odstupanje od zadanih parametara sankcionira te je zbog toga sama vrijednost dobrote vrlo visoka. Ako je prijeđen dopušteni kapacitet vozila, to prekoračenje (*overcapacity*) množit će se s 1000, u suprotnom će biti nula. Ako postoje vremenska ograničenja posluživanja kupaca (*time window*) kažnjava se svaki dolazak nakon početka trajanja vremenskog ograničenja (*overtime*) i dolazak nakon isteka vremenskog ograničenja (*late*).

Algorithm 1: Calculate fitness

```
Data: distance, overcapacity, time window, late, vehicle fleet number
Result: fitness
fitness = distance + 1000 * overcapacity;
if time window then
    fitness += overtime;
    if late > 0 then
        fitness *= late * vehicle fleet number;
    else
    end
end
```

4.3. Operatori

Implementirane su različite vrste selekcija, križanja i mutacija za korištene prikaze.

4.3.1. Selekcija

K-turnirska selekcija odabire k jedinki od kojih će odabrati najbolju jedinku.

Jednostavna proporcionalna selekcija (eng. Roulette Wheel Selection) može se vizualizirati kao kotač ruleta. Na početku se izračuna suma dobrota (eng. fitness) svih jedinki u populaciji. Zatim se određuje nasumični broj koji je između nule i sume dobrota. Za svaku jedinku u populaciji zatim računamo dobrotu koja se zbraja sa svim prethodnim. To se ponavlja sve dok ta suma nije veća od nasumičnog broja.

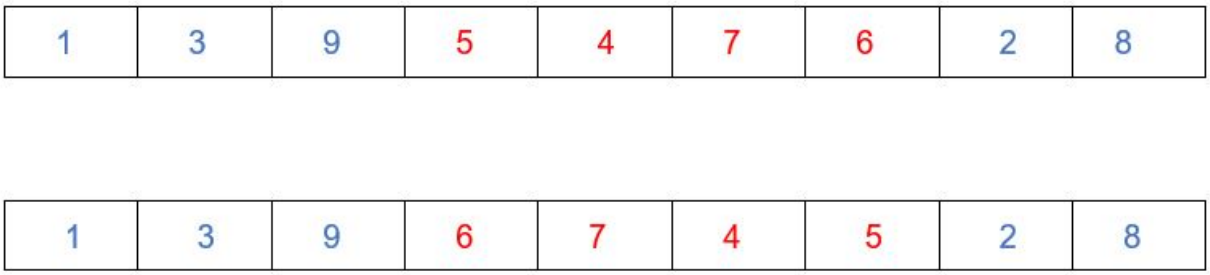
4.3.2. Operatori za prikaz permutacijskim nizom

Križanje s jednom točkom prekida (eng. Point Crossover) provodi se tako što se odabere nasumični broj koji maksimalno može biti broj kupaca. Prva lista u permutacijskom prikazu sastoji se samo od kupaca te će se prva lista jedinke nastale križanjem sastojati od dijela prve liste prvog roditelja i dijela prve liste drugog roditelja. Mjesto na kojem će se ti dijelovi spajati je prije nasumično odabrani broj. Druga lista nove jedinke bit će druga lista prvog roditelja.

Inverzna mutacija (eng. Inverse Permutation) permutacijskog niza pronalazi dva nasumična broja čije su vrijednosti manje od veličine prve liste permutacijskog niza (lista s kupcima). Vrijednosti koje se nalaze u prvoj listi između dva indeksa dobivena

nasumično zamijene redoslijed. Isto se ponovi i u drugoj listi permutacije (listi sa vozilima). Postupak je prikazan na slici 4.3.

U mutaciji zamjene mjesta (eng. Toggle Mutation) sa slike 4.4, dvije nasumično odabrane vrijednosti u prvoj list permutacije zamijene mjesta. Isto se događa i u drugoj listi prikaza.



Slika 4.3: Primjer inverzne mutacije za prvu listu vozila

Slika 4.4: Primjer mutacije zamjene mjesta

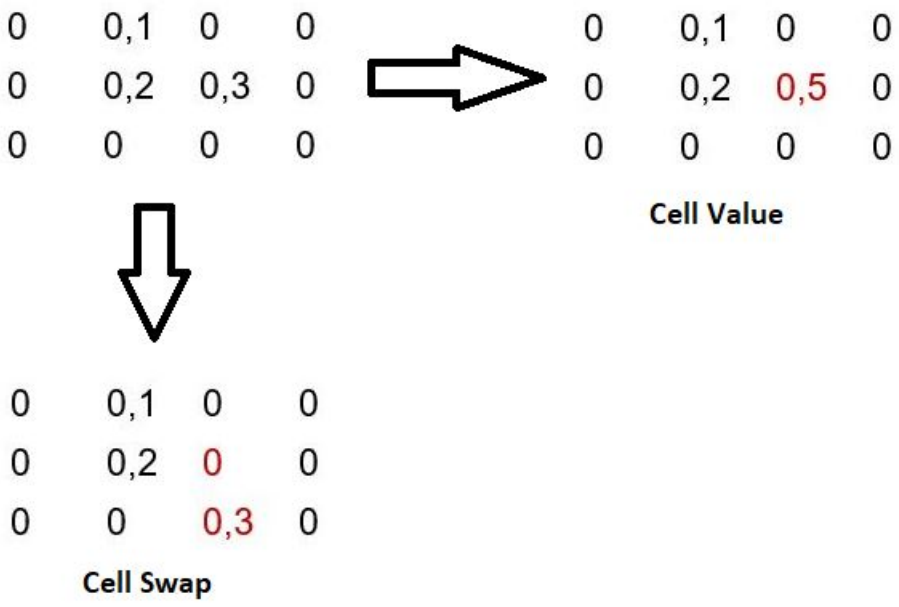
4.3.3. Operatori za matrični prikaz

Rezultat uniformnog križanja (eng. Uniform Crossover) nastaje tako što će svaki novi stupac doći od stupca koji se pod istim brojem nalazi kod jednog od roditelja. Roditelj čiji je stupac uzet odabran je nasumično.

Mutacija zamjenom ćelija (eng. Cell Swap) nasumično bira mjesto u matrici na kojem je broj različit od nule. Taj broj će biti upisan na neko nasumično mjesto u istom stupcu, dok će na mjesto na kojem je bio taj broj biti upisana nula.

Mutacija vrijednosti ćelije (eng. Cell Value) generira novi nasumični broj te se njime zamijeni neka vrijednost u matrici koja je različita od nule.

Obje vrste mutacije prikazene su na slici 4.5.



Slika 4.5: Primjer mutacija za matrični prikaz

4.4. Prikaz korisničkim sučeljem

U ovom je radu implementirano korisničko sučelje. Na alatnoj traci (slika 4.6.) može se pritisnuti *Import* te tako odabrati datoteka koja sadrži instancu problema. Korisnika će se zatim tražiti da odabere datoteku s već postojećim rješenjem. Izgled datoteke s instancom problema isti je kao izgled one s Hombergerovom instancom prikazan na slici 5.2. Datoteka s rješenjem ima također zadani izgled vidljiv na slici 4.7. Rješenje će se zatim prikazati u sučelju kao na slici 4.8.

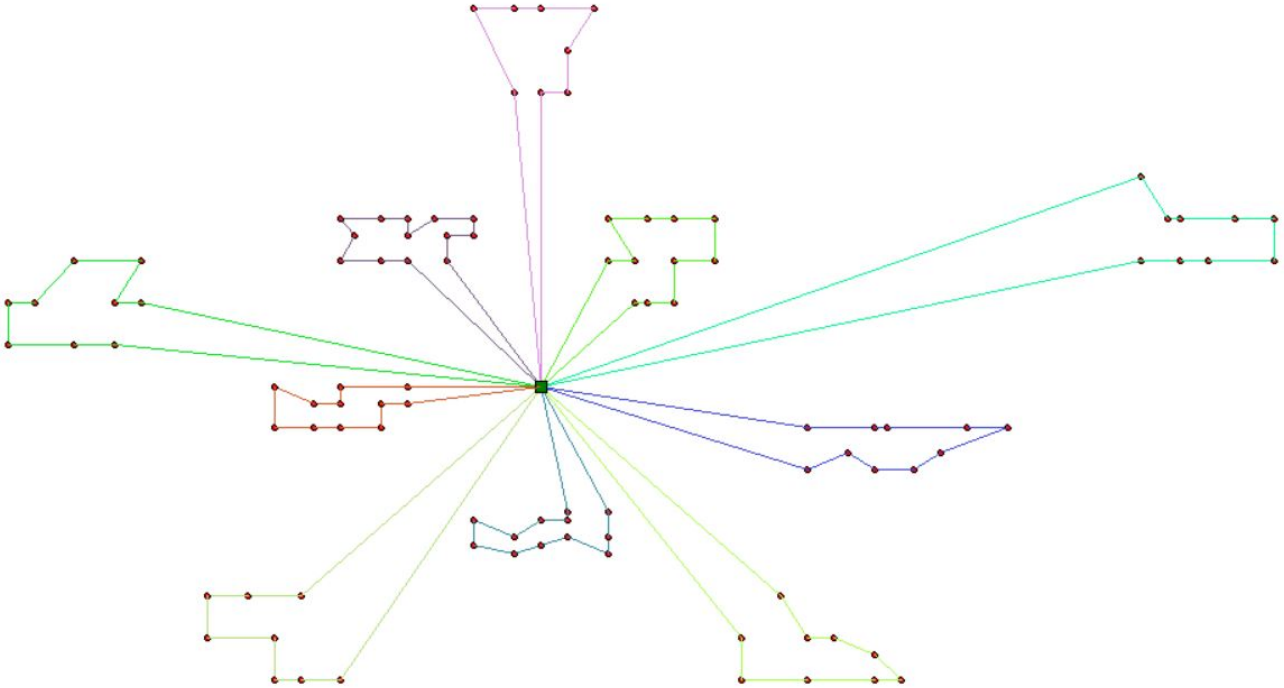
Slika 4.6: Alatna traka

```
Route 0: 81, 78, 76, 71, 70, 73, 77, 79, 80
Route 1: 57, 55, 54, 53, 56, 58, 60, 59
Route 2: 98, 96, 95, 94, 92, 93, 97, 100, 99
Route 3: 32, 33, 31, 35, 37, 38, 39, 36, 34
Route 4: 13, 17, 18, 19, 15, 16, 14, 12
Route 5: 90, 87, 86, 83, 82, 84, 85, 88, 89, 91
Route 6: 43, 42, 41, 40, 44, 46, 45, 48, 51, 50, 52, 49, 47
Route 7: 67, 65, 63, 62, 74, 72, 61, 64, 68, 66, 69
Route 8: 5, 3, 7, 8, 10, 11, 9, 6, 4, 2, 1, 75
Route 9: 20, 24, 25, 27, 29, 30, 28, 26, 23, 22, 21
```

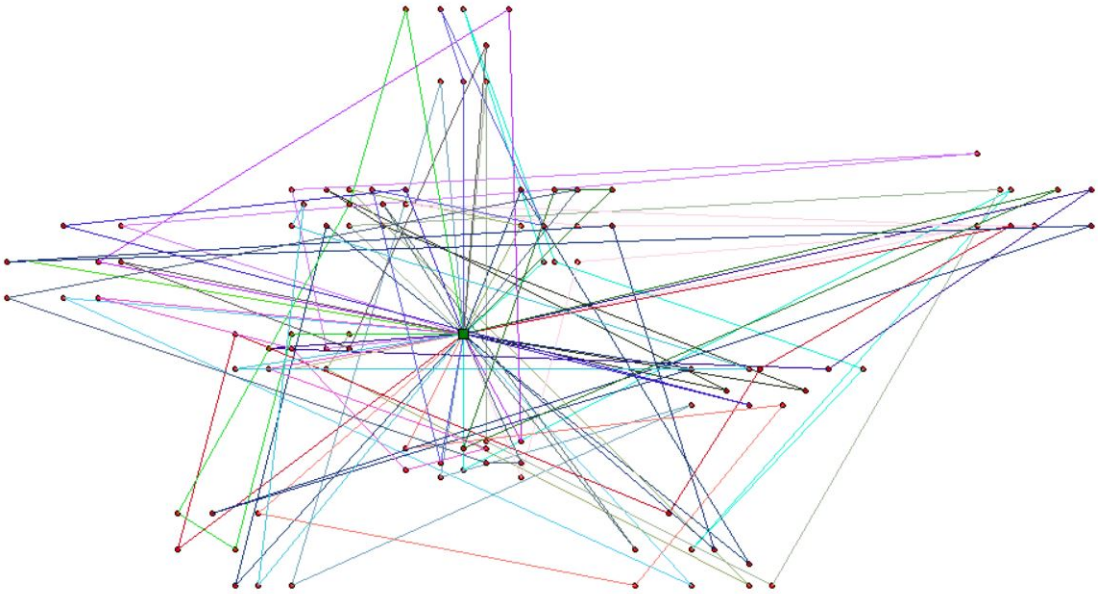
Slika 4.7: Primjer rješenja zapisanog u datoteci

Pritiskom na *AI* korisnik bira datoteku instance problema te algoritam kojim želi riješiti problem. Moguće je odabrati Clarke-Wright ili genetski algoritam (slike 4.9 i 4.10). Rješenje problema bit će prikazano po završetku algoritma.

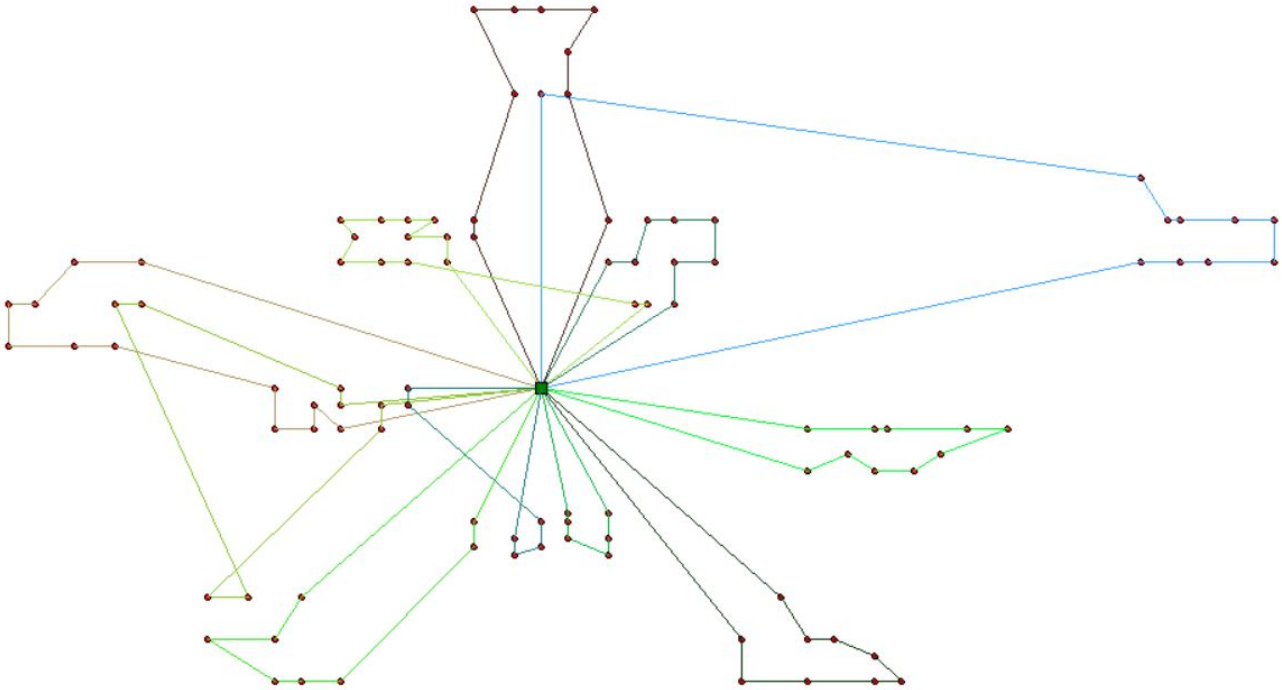
Odabirom *New* na alatnoj traci briše se sve trenutno nacrtano te korisnik može ponovno prikazati što želi.



Slika 4.8: Prikaz rješenja u sučelju (zadano rješenje) - *Import*

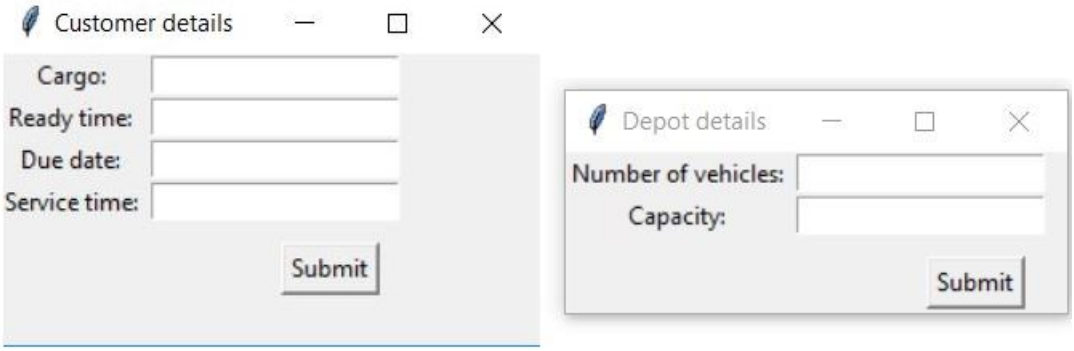


Slika 4.9: Prikaz rješenja genetskim algoritmom (prikaz permutacijskim nizom s brojem vozila)

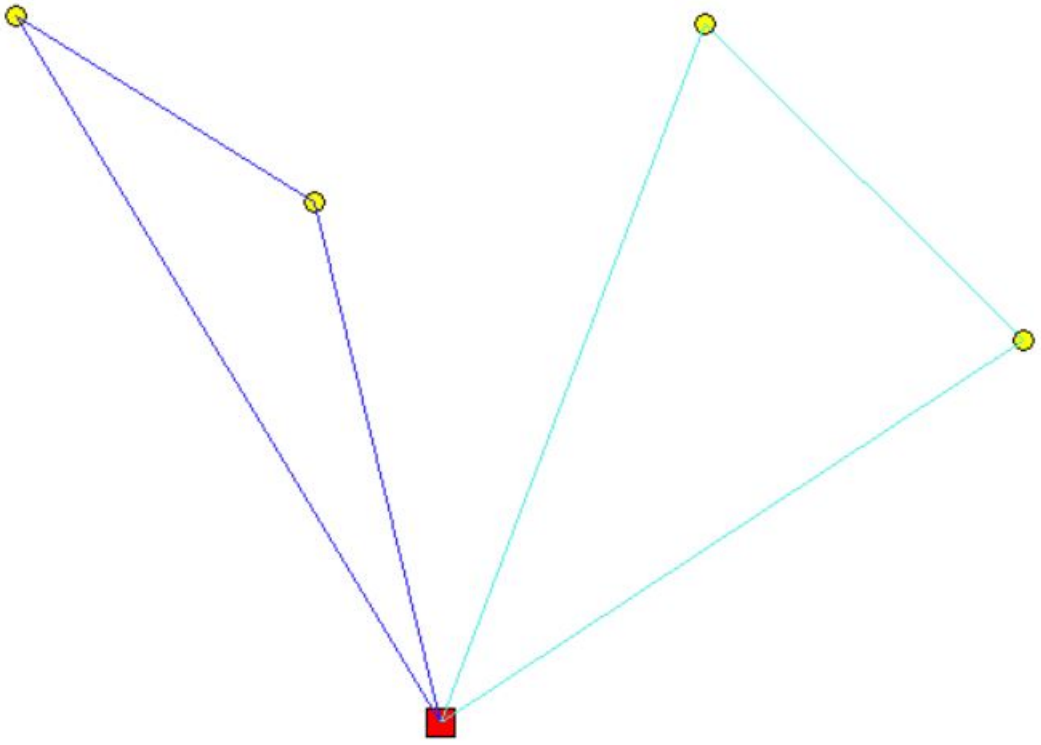


Slika 4.10: Prikaz rješenja Clarke-Wright algoritmom

Također postoji mogućnost da korisnik sam crta svoje skladište i kupce jednostavnim pritiskom na Canvas. Po prvom pritisku na bijelu pozadinu otvara se prozor za upis parametara skladišta, a svaki put kada se označi bilo koje drugo mjesto podrazumijeva se da je označen kupac te se otvara prozor za upis parametara za kupca (slika 4.11.). Odabirom *Export problem* na alatnoj traci korisnik može odabrati kojim algoritmom želi riješiti problem koji je sam zadao. Na slici 4.12. prikazno je rješnje jednostavnog problema.



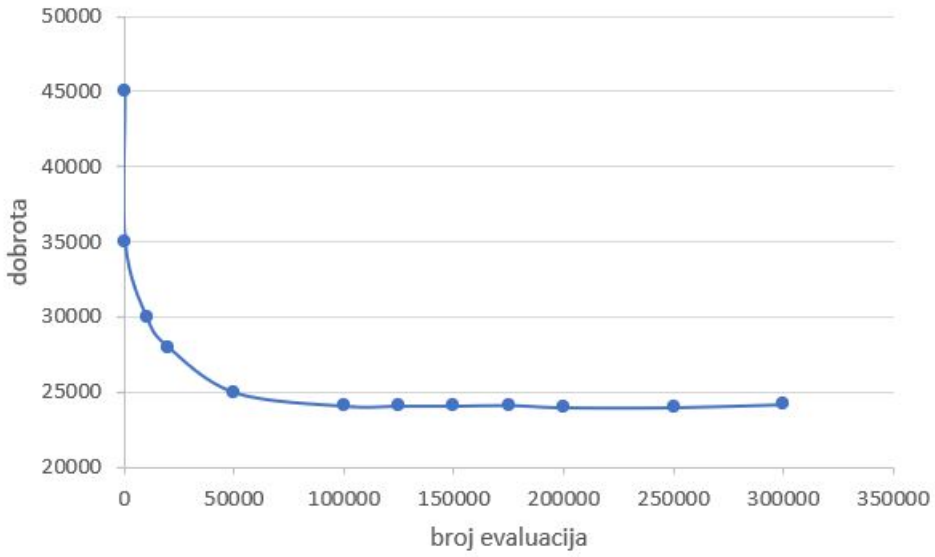
Slika 4.11: Unos parametara skladišta i kupca za korisnikov problem



Slika 4.12: Jednostavan korisnikov problem riješen genetskim algoritmom

5. Analiza rezultata

Rezultati implementiranog genetskog algoritma analizirat će nekoliko pitanja. U svakoj analizi broj evaluacija je 120 000 te je na slici 5.2. prikazan nagli pad funkcije dobrote pri rastu broja evaluacija nakon čega funkcija dobrote poprima konstantan oblik. Za svako ispitivanje pojedinog slučaja algoritam je pokrenut 20 puta. Korištene su Hombergove instance s 400 kupaca (slika 5.2.). U tablici 5.1 prikazane su početne vrijednosti promjenjivih parametara u algoritmu. Ako je neka vrijednost promijenjena, to će u tekstu biti naglašeno.



Slika 5.1: Kriterij zaustavljanja

Tablica 5.1: Početne vrijednosti svih korištenih parametara

parametar	vrijednost
broj evaluacija	120 000
veličina populacije	50
broj kupaca	400
veličina turnira	3
faktor mutacije	0.05

```
PROBLEM name

VEHICLE
NUMBER     CAPACITY
x            y

CUSTOMER
CUST NO.   XCOORD.   YCOORD.   DEMAND    READY TIME   DUE DATE   SERVICE TIME
 ...        ...       ...       ...         ...          ...         ...
```

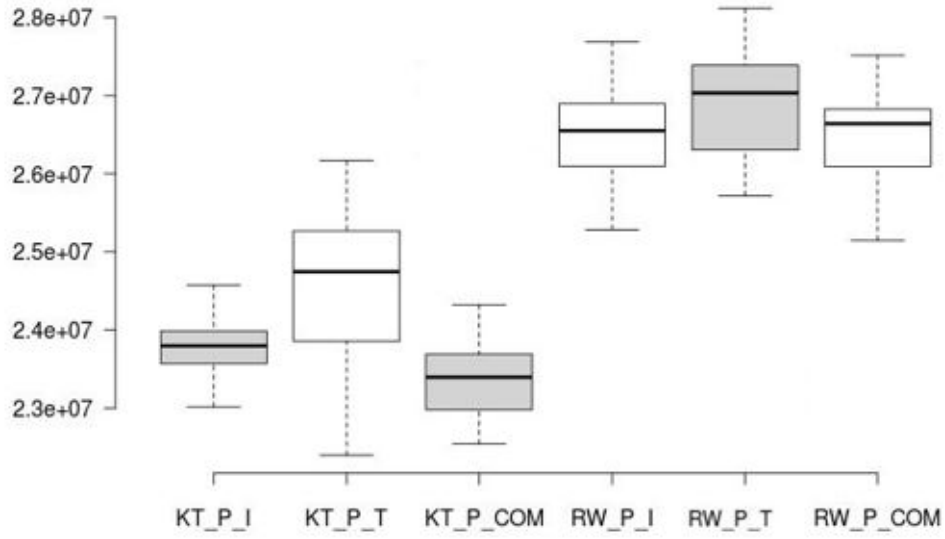
Slika 5.2: Hombergerova instanca

5.1. Kombiniranje operatora prikaza permutacijom

Kombiniranjem različitih operatora dobivaju se sljedeće kombinacije:

1. K-turnirska selekcija s križanjem u jednoj točci prekida i inverzna mutacija (KTPI)
2. K-turnirska selekcija s križanjem u jednoj točci prekida i mutacija zamijene mjesta (KTPT)
3. Jednostavna proporcionalna selekcija u jednoj točci prekida i mutacija zamijene mjesta (RWPT)
4. Jednostavna proporcionalna selekcija u jednoj točci prekida i inverzna mutacija (RWPI)
5. K-turnirska selekcija s križanjem u jednoj točci prekida i kombinacija inverzne mutacije i mutacije zamijene mjesta (KTPCOM)
6. Jednostavna proporcionalna selekcija s križanjem u jednoj točci prekida i kombinacija inverzne mutacije i mutacije zamijene mjesta (RWPCOM)

Analizom rezultata iz grafa (slika 5.3.) se može zaključiti kako se najbolji rezultati postižu K-turnirskom selekcijom čija je veličina turnira 3 s križanjem u jednoj točci prekida i kombinacijom inverzne mutacije i mutacije zamijene mjesta, dok se najlošiji rezultati bilježe jednostavnom proporcionalnom selekcijom u jednoj točci prekida i mutacijom zamijene mjesta.
Isto se tako može zaključiti da mutacija zamijene mjesta daje malo lošije rezultate od inverzne mutacije te da K-turnirska daje bolje rezultate u odnosu na jednostavnu proporcionalnu selekciju.



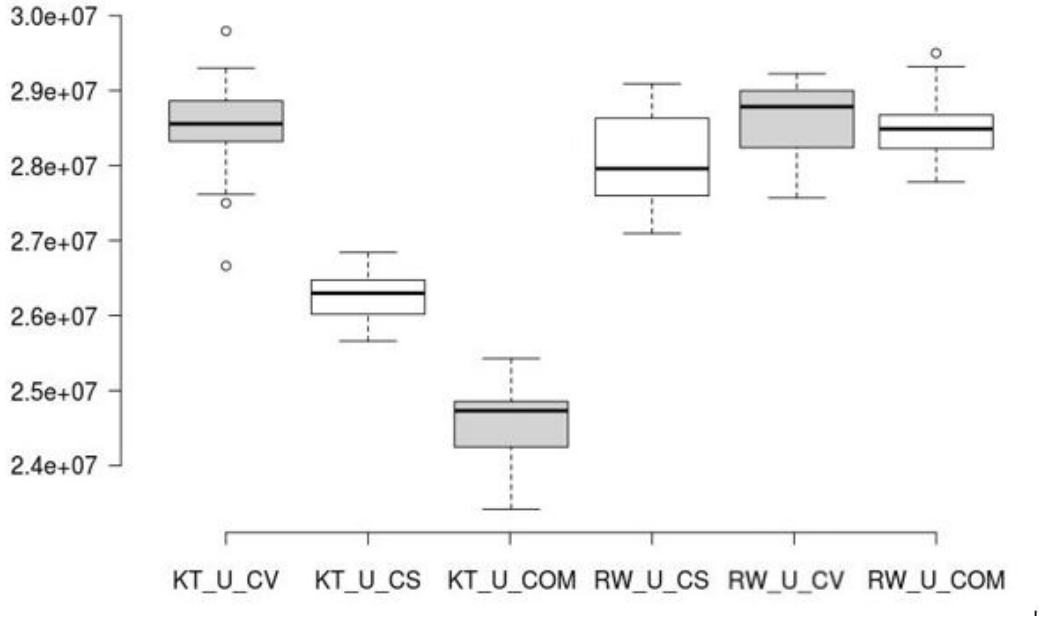
Slika 5.3: Kombiniranje operatora prikaza permutacijskim nizom s brojem vozila

5.2. Kombiniranje operatora u matričnom prikazu

Kombiniranjem različitih operatora dobivaju se sljedeće kombinacije:

1. K-turnirska selekcija s uniformnim križanjem i mutacijom vrijednosti ćelije (KTUCV)
2. K-turnirska selekcija s uniformnim križanjem i mutacijom zamjene ćelija (KTPCS)
3. Jednostavna proporcionalna selekcija s uniformnim križanjem i mutacijom vrijednosti ćelije (RWUCV)
4. Jednostavna proporcionalna selekcija s uniformnim križanjem i mutacijom zamjene ćelija (RWUCS)
5. K-turnirska selekcija s uniformnim križanjem i kombinacijom mutacije zamijene ćelija i mutacije vrijednosti ćelije (KTUCOM)
6. Jednostavna proporcionalna selekcija s uniformnim križanjem i kombinacijom mutacije zamijene ćelija i mutacije vrijednosti ćelije (RWUCOM)

Analizom različitih kombinacija operatora u matričnom prikazu utvrđeno je da najbolje rezultate daje kombinacija K-turnirske selekcije s uniformnim križanjem i kombinacijom mutacije zamjene ćelija i mutacije vrijednosti ćelije (slika 5.4). Malo lošije rezultate, ali ipak bolje od ostalih kombinacija, daje K-turnirska selekcija s uniformnim križanjem i mutacijom zamijene ćelija.



Slika 5.4: Kombiniranje operatora matričnog zapisa

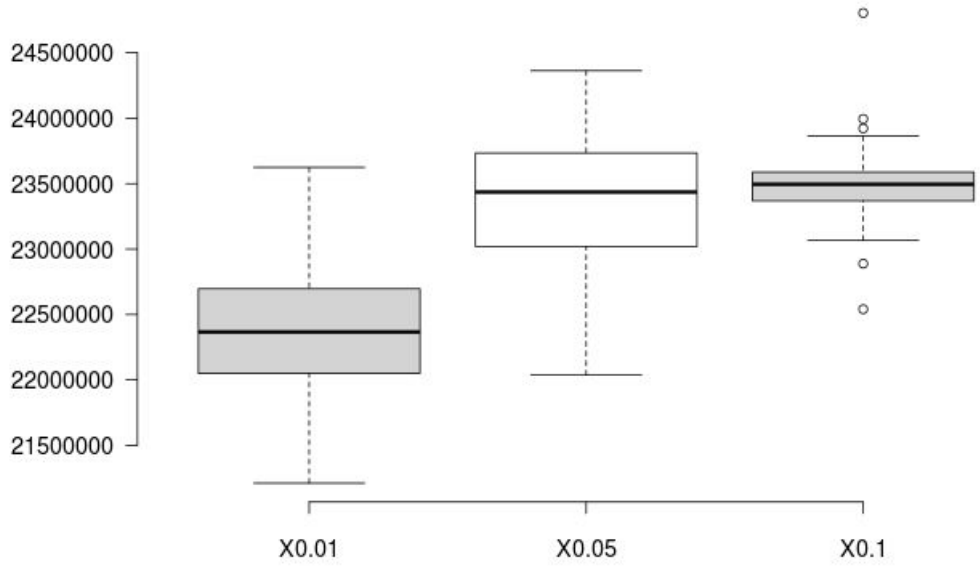
5.3. Utjecaj mutacije

Pri analiziranju utjecaja faktora mutacije na algoritam u svakom su pokretanju jednaki: instanca problema, broj jedinki u populaciji i korišteni operatori . Analizirani su slučajevi gdje je faktor mutacije jednak 0,01, 0,05 i 0.1. 100 vozila na raspolaganju je za posjećivanje 400 kupaca.

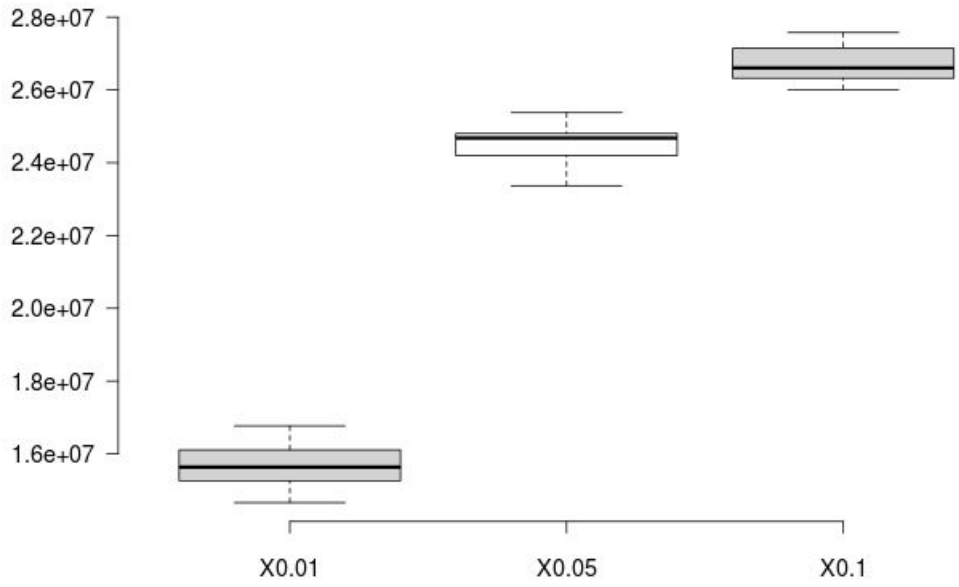
Operatori korišteni pri ovoj analizi su najbolje kombinacije operatora dobivene analizom istih za svaki prikaz, dok su ostali parametri jednaki početnim.

Na slici 5.5. prikazan je utjecaj na permutacijski prikaz te je vidljivo da su rezultati bolji ako se kao faktor mutacije koristi 0.01. Veći faktori mutacije daju približno jednake rezultate.

Bolji rezultati u matričnom prikazu dobivaju se ako je faktor mutacije 0.01, dok se korištenjem većih faktora dobivaju značajno lošiji rezultati što je prikazano na slici 5.6.



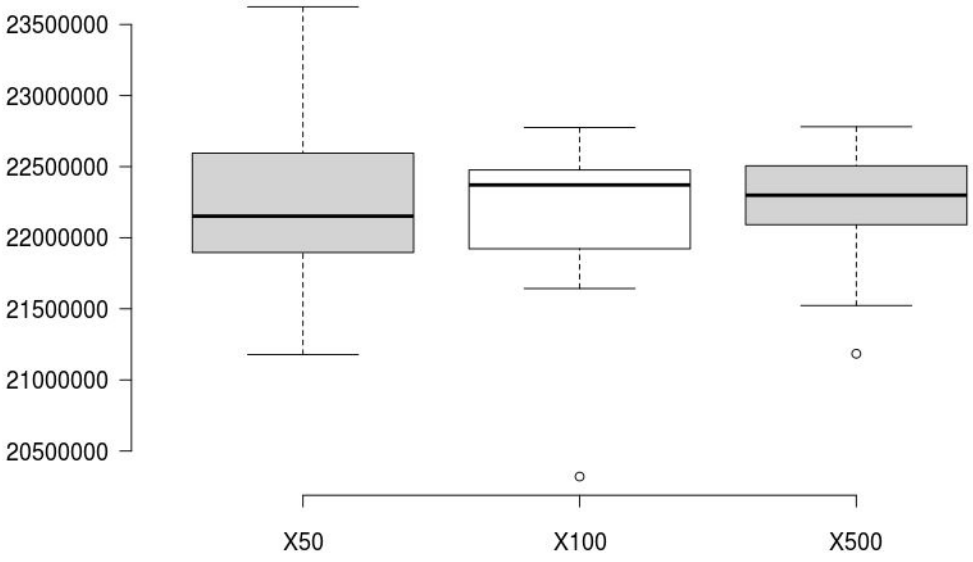
Slika 5.5: Utjecaj različitih faktora mutacije za permutacijski prikaz



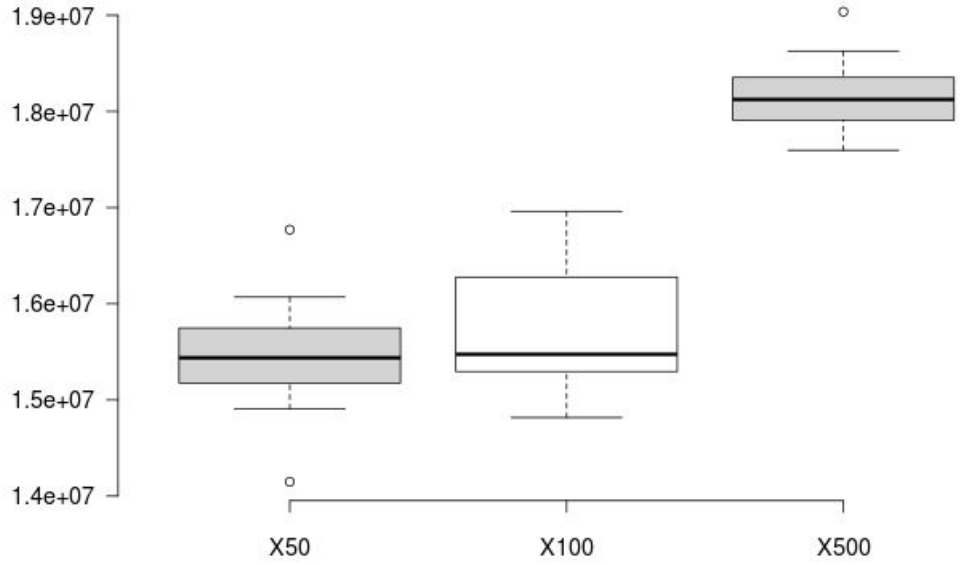
Slika 5.6: Utjecaj različitih faktora mutacije za matrični prikaz

5.4. Utjecaj veličine populacije

Korištenjem najboljih kombinacija operatora i faktora mutacije 0.01 za oba prikaza dobiveni su rezultati analize utjecaja veličine populacije. Svi ostali parametri su jednaki početnim. Iz slika 5.7. i 5.8 vidljivo je da su za populacije od 50, 100 i 500 jedinki rezultati vrlo slični u permutacijskom prikazu. U prikazu matricom korištenjem populacija veličine 50 i 100 jedinki rezultati su slični, dok se populacijom od 500 jedinki dobivaju ipak malo lošiji rezultati.



Slika 5.7: Utjecaj veličine populacije za permutacijski prikaz

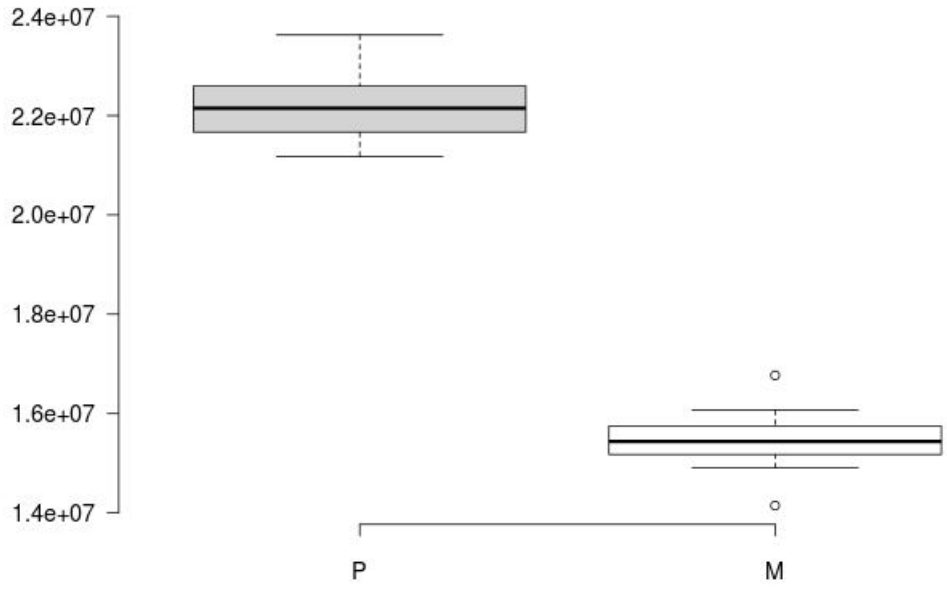


Slika 5.8: Utjecaj veličine populacije za matrični prikaz

5.5. Usporedba prikaza permutacijom i matricom

Na slici 5.9. su prikazi permutacijskog i matričnog prikaza dobiveni kombinacijom operatora koje su pokazale najbolje rezultate te faktorom mutacije 0.01 i veličinom populacije od 50 jedinki. Svi ostali parametri jednaki su početnim. Pri analiziranju, primjeri su pokrenuti za iste instance problema. Vidljivo je da za istu instancu pro-

blema bolje rezultate daje prikaz matricom



Slika 5.9: Usporedba permutacijskog i matričnog prikaza

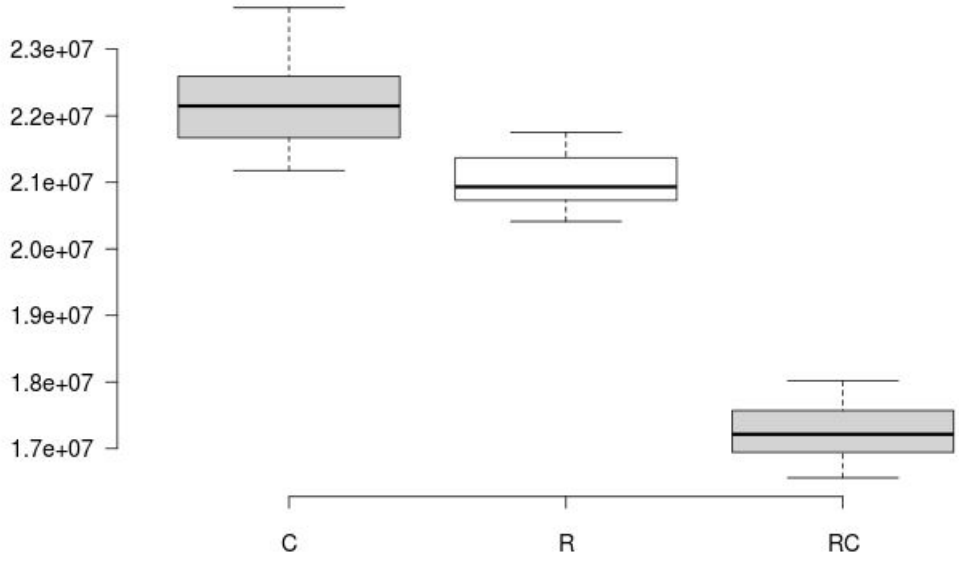
5.6. Različite instance problema

U radu su korištene Hombergerove instance s 400 kupaca. Postoje 3 kategorije instanci, a to su:

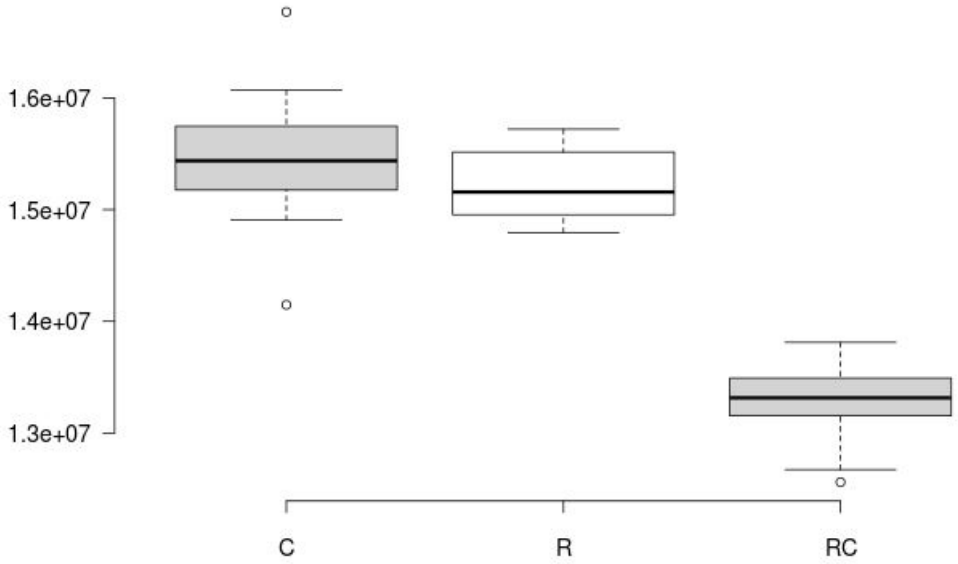
1. C-tip : grupirani kupci (eng. clustered customers)
2. R-tip : uniformno distribuirani kupci (eng. uniformly distributed customers)
3: RC-tip: kombinacija C i R tipa

Za obje vrste prikaza korištene su kombinacije operatora koje su u prethodnoj analizi dale najbolje rezultate za istu vrstu problema, faktor mutacije 0.01. Ostale vrijednosti parametara jednake su početnim.

Analizom je utvrđeno da su najlakši primjeri problema upravo instance RC tipa. Kombinacija grupiranja kupaca i uniformno distribuiranih kupaca daje najbolje rezultate za obje vrste prikaza što se vidljivo na slikama 5.10. i 5.11.



Slika 5.10: Različite instance problema prikazane permutacijom



Slika 5.11: Različite instance problema prikazane matricom

5.7. Usporedba Clarke-Wright i genetskog algoritma

Usporedba Clarke-Wright algoritma i prikaza rješenja genetskog algoritma odvija se pomoću jedinstvene dobrote Clarke-Wrighta i prosječne dobrote najboljih jedinki dobivenih s 20 pokretanja pojedinog problema (tablica 5.2.).

Iz tablice je vidljivo da je korištenjem matričnog prikaza dobrota manja od Clarke-Wright algoritma te je time rezultat algoritma bolji. Clarke-Wright i prikaz permutacijskim nizom s brojem vozila daju vrlo slične rezultate.

Tablica 5.2: Usporedba algoritama

	R	C	RC
Clarke-Wright	20895398.57	21652588.29	17678605.42
GA-permutacija	20932182.17	22147781.47	17214872.75
GA-matrica	15157595.82	15437166.21	13314553.09

5.8. Analiza dobrote

Pri analizi dobrote korišteno je 400 kupaca i 100 mogućih vozila. U tablici 5.3. prikazani su parametri koji čine dobrotu jedinke. Analizirane su dvije jedinke. Jedna je dobivena permutacijskim nizom s brojem vozila i njena je dobrota 16687111.86, a u dostavi je sudjelovalo 99 vozila, dok je druga jedinka dobivena matričnim prikazom i njena je dobrota 13189007.03 s 97 korištenih vozila. Obje su jedinke dobivene korištenjem najboljih kombinacija operatora iz prethodne analize, faktorom mutacije 0.01 i veličinom populacije 50. Svi ostali parametri jedinki su početnim. Kapacitet je jednak kod obje jedinke i iznosi 7127.

Vidljivo je da nema prekoračenja kapaciteta (*capacity*) te da je kod više od polovice kupaca primijećen dolazak nakon isteka vremenskog ograničenja (*late*). Kod oba su prikaza prisutni i dolasci nakon što je krenulo vrijeme za posluživanje kupca. Svi prikazani parametri bolji su kod matričnog prikaza što na posljetku vodi tome da on ima bolju dobrotu.

Tablica 5.3: Analiza dobrote

	overcapacity	distance	overtime	late	fitness
GA-permutacija	0	42997.66	16988.40	244	16687111.86
GA-matrica	0	39487.09	15750.24	208	13189007.03

6. Zaključak

U ovom radu opisan je i analiziran problem usmjeravanja vozila s vremenskim ograničenjima. Korištenjem Clarke-Wright algoritma uštede i genetskog algoritma pokušala su se dobiti čim bolja rješenja. Analizom je utvrđeno da matrični prikaz daje bolje rezultate u odnosu na prikaz permutacijom s brojem vozila. Kombinacija K-turnirske selekcije, uniformnog križanja i kombinacije mutacije zamijene ćelija i mutacije vrijednosti ćelije daje najbolji rezultat od svih analiziranih inačica genetskog algoritma.
Permutacijski prikaz s brojem vozila daje lošije rezultate od matričnog prikaza, a uspoređujući samo kombinacije tog permutacijskog prikaza može se zaključiti da K-turnirska selekcija s križanjem u jednoj točci prekida i kombinacijom inverzne mutacije i mutacije zamijene mjesta daje najbolje rezultate.
Od analiziranih instanci problema osjetno bolji rezultati kod oba prikaza dobivaju se kombinacijom grupiranih kupaca i uniformno distribuiranih kupaca (RC tip). Manji faktor mutacije i manji broj jedinki u populaciji dobro utječu na rezultat.
Uspoređujući Clarke-Wright i genetski algoritam zaključuje se da je rezultat dobiven matričnim prikazom bolji od rezultata dobivenog Clarke-Wright algoritmom uštede, dok prikaz permutacijskim nizom s brojem vozila ima slične rezultate kao korišteni algoritam uštede.
Implementirano je grafičko sučelje koje pruža mogućnost vizualizacije postojećih rješenja instanci problema i rješenje zadanih instanci problema nekim od implementiranih algoritama. Omogućeno je i rješavanje problema koji zada sam korisnik implementiranim algoritmima.
Različiti pokušaji rješavanja ovog problema česti su, ali potrebni jer su verzije problema prisutne u mnogim područjima. Zanimljivo je to što jedan kombinatorni problem ima toliko širok utjecaj i primjenu. Važno je pokušavati pronaći bolja rješenja jer to vodi ne samo rješenju nekog apstraktnog zadatka, već rezultati mogu biti vidljivi u stvarnom svijetu na mnogo mjesta. U budućnosti se može sadašnji algoritam upotpuniti s još raznovrsnih operatora i parametara koji će proširiti skup mogućih rješenja te povećati vjerojatnost dobivanja boljeg.

LITERATURA

[1] Vidulić, J. *Rješavanje problema usmjeravanja vozila metaheuristikama u statičkim i dinamičkim uvjetima.* Završni rad. Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, 2019.

[2] Nemčić, J. *Primjena metaheurističkih algoritama na problem usmjeravanja vozila s vremenskim prozorima i mogućnošću preuzimanja tereta.* Završni rad. Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, 2019.

[3] Vlašić, I., Đurasević, M., Jakobović, D. *A comparative study of solution representations for the unrelated machines environment.* Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva

[4] Networking and Emerging Optimization, Vehicle Routing Problem, 2013, URL `http://neo.lcc.uma.es/vrp/vehicle-routing-problem/`; pristupljeno travanj i svibanj 2020.

[5] Bradavica, V. *Algoritmi koji oponašaju procese u prirodi.* Seminarski rad, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva

[6] Halim, S., Yoanita, L. *Adjusted Clustering Clarke-Wright Saving Algorithm for Two Depots-N Vehicles.*, Department of Industrial Engineering, Petra Christian University, Surabaya, Indonesia

[7] Prokopec, A. *Prilagodljiv genetski algoritam.* Diplomski rad, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, 2009.

Rješenje problema usmjeravanja vozila s vremenskim ograničenjima različitim inačicama evolucijskog algoritma

Sažetak

Tema ovog rada je problem usmjeravanja vozila s vremenskim ograničenjima. Problem je opisan i implementiran Clarke-Wrightom algoritmom uštede i genetskim algoritmom. Genetski algoritam opisan je u kontekstu evolucijskog algoritma. Rješenja genetskog algoritma prikazana su permutacijskim nizom s brojem vozila i matričnim prikazom te su opisani operatori za svaki prikaz. Objašnjeno je implementirano korisničko sučelje. U radu su prikazani rezultati dobiveni analizom genetskog algoritma te utjecaj operatora, tipa instance problema, vrste prikaza i vrijednosti mutacije na sam algoritam.

Ključne riječi: problem usmjeravanja vozila, VRPTW, evolucijski algoritam, genetski algoritam, selekcija, križanje, mutacija