Rješavanje problema usmjeravanja vozila s heterogenim flotama korištenjem genetskog algoritma

Jukić, Tin

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 697

RJEŠAVANJE PROBLEMA USMJERAVANJA VOZILA S HETEROGENIM FLOTAMA KORIŠTENJEM GENETSKOG ALGORITMA

Tin Jukić

Zagreb, veljača 2025.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 697

RJEŠAVANJE PROBLEMA USMJERAVANJA VOZILA S HETEROGENIM FLOTAMA KORIŠTENJEM GENETSKOG ALGORITMA

Tin Jukić

Zagreb, veljača 2025.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 30. rujna 2024.

DIPLOMSKI ZADATAK br. 697

Pristupnik:	**Tin Jukić (0036527458)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	izv. prof. dr. sc. Marko Đurasević

Zadatak: **Rješavanje problema usmjeravanja vozila s heterogenim flotama korištenjem genetskog algoritma**

Opis zadatka:

Opisati problem usmjeravanja flote vozila, koja se sastoji od konvencionalnih, plug-in hibridnih i električnih vozila. Navesti metode koje se mogu iskoristiti za rješavanje navedenog problema, posebice heurističke i metaheurističke metode. Istražiti prikaze kojima se mogu predstaviti rješenja problema usmjeravanja vozila. Riješiti problem korištenjem genetskog algoritma ili sličnog prirodom inspiriranog optimizacijskog algoritma. Osmisliti i testirati razne parametre korištene metode rješavanja. Prilagoditi prikaz rješenja i genetske operatore evolucijskih algoritama u svrhu kvalitetnijeg rješavanja problema. Analizirati rezultate te predložiti potencijalna poboljšanja odabranog algoritma te moguće daljnje korake istraživanja. Radu priložiti izvorne tekstove programa, dobivene rezultate uz potrebna objašnjenja i korištenu literaturu.

Rok za predaju rada: 14. veljače 2025.

Zahvaljujem se svojim roditeljima Nedžadi i Toniju, svom bratu Janu, mojoj keki Valdeti te ostaloj obitelji, djevojci Hani, prijateljima te mentoru, što su mi bili podrška tijekom studija te što su vjerovali u mene i pomagali mi da završim studij na FER-u.

Sadržaj

1. Uvod

Jedan od najčešće rješavanih problema u području optimizacije je problem usmjeravanja vozila (Vehicle Routing Problem, VRP). Današnji način života izrazito je užurban, ljudi nemaju vremena, pogotovo u većim gradovima te se sve češće okreću online kupovini. Samim time pojavljuje se potreba za boljim i organiziranijim načinom isporuke naručenih stvari i usluga do korisnika. Postavlja se pitanje: *kako na optimalan način obići korisnike*. Upravo tim pitanjem bavi se rješavanje problema usmjeravanja vozila.

Cilj rješavanja ovog problema je pronaći optimalni put kojim bi trebalo obići korisnike. No, ovo rješenje nije jedinstveno i ovisi o kriterijima koje želimo zadovoljiti. Ti kriteriji su brojni te ovise o korištenoj instanci problema.

U nastavku ovog rada bit će detaljnije opisan problem usmjeravanja vozila, njegove instance te će biti navedene metode koje se mogu koristiti za njegovo rješavanje. Također će biti navedena i detaljnije opisana korištena instanca u ovom radu te način na koji je taj problem riješen na konkretnim primjerima. Za znatiželjnije čitatelje, poziva se proučavanje rada [1], koji se bavi istom tematikom kao i ovaj rad.

2. Opis problema

2.1. Problem usmjeravanja vozila (VRP)

Problem usmjeravanja vozila (Vehicle Routing Problem, VRP) [2] jedan je od najčešće rješavanih optimizacijskih problema u području strojnog učenja. Svaka instanca VRP problema sastoji se od čvorova i vozila. Postoje tri glavne vrste čvorova: čvor izvorište, čvorovi korinisici te čvorovi postaje (benzinske crpke, punionice). Sva vozila kreću iz čvora izvorišta te obilaze čvorove korisnike, dok se ne obiđu svi korisnici. Ograničenja u problemu mogu biti razna, a samim time postoje i razne inačice ovog problema. Neke od inačica su: VRP s ograničenim kapacitetom vozila (CVRP), VRP s vremenskim prozorom (VRPTW), VRP s više izvorišta, VRP s heterogenom flotom, VRP s električnim vozilima (EVRP) te brojne druge [3] [2] [4]. Osnovna instanca VRP problema ima pretpostavku da su sva vozila jednaka te da su im kapaciteti identični, što ne mora biti zadovoljeno kod drugih inačica VRP problema. Cilj svakog VRP problema je pronaći put kojim se mogu obići svi korisnici uz zadovoljavanje zadanih ograničenja.

Za rješenje VRP problema dobije se put kojim vozila trebaju obići sve korisnike te ujedno zadovoljiti sva ograničenja. Ovaj problem nema jedinstveno rješenje, odnosno postoje brojni putevi kojim se mogu uspješno obići svi korisnici, uz zadovoljavanje svih navedenih ograničenja. Time možemo zaključiti da samo pronalaženje puta nije previše kompleksno. Izazov predstavlja drugi problem. Kako dobiti **optimalni** put? Tu se javlja ideja za korištenjem algoritama optimizacije.

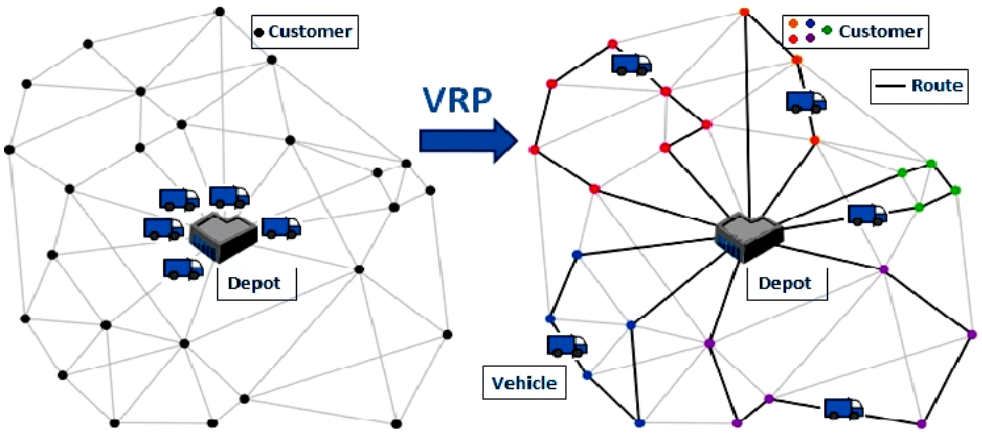
Postoje brojni algoritmi koji se mogu koristitit za rješavanje ovog problema, no najčešće se koriste dva poznata algoritma: genetski algoritam (Genetic Algorithm, GA) i umjetna neuronska mreža (Artificial Neural Network, ANN). Oni omogućuju pronalaženje optimalnog puta kojim bi se trebali obilaziti korisnici, kako bi se minimizirali tro-

škovi puta. Ti troškovi su razni i najčešće ovise o inačici problema koja se razmatra. Na primjer, može se težiti maksimalnom iskorištavanju kapaciteta pojedinog vozila ili minimiziranjnu prijeđene udaljenosti između pojedinih čvorova za svako vozilo. Uz to što postoje razni kriteriji, oni se također mogu i kombinirati. Tako možemo u isto vrijeme težiti maksimalnoj iskorištenosti vozila i minimizaciji prijeđenog puta [3].

Svaki VRP problem mora imati definirane potrebne parametre. Potrebno je definirati koliko čvorova problem sadrži te koji su to čvorovi. Također, mora biti navedeno koliko vozila problem sadrži, koja su to vozila i njihova svojstva. Slika 2.1. predstavlja ilustraciju klasičnog problema VRP [5].

Valja napomenuti da postoji vjerojatnost da algoritam ne može pronaći put kojim bi se mogli obići svi zadani korisnici. No, to nije toliko čest problem u literaturi, jer se za potrebe istraživanja kreiraju posebni podaci, za koje postoji put kojim se mogu obići svi korisnici.

U nastavku će biti detaljnije opisana korištena instanca u ovom radu te će biti navedena sva njena svojstva i ograničenja.



Slika 2.1. Problem usmjeravanja vozila (VRP) [5]

2.2. Problem usmjeravanja električnih vozila (EVRP)

Problem usmjeravanja električnih vozila (EVRP) sve je izraženiji problem današnjice. Vlade teže ugljičnoj neutralnosti u transportu te su voljni dati poticaje, kako bi povećali prodaju električnih vozila. Zbog toga raste broj električnih vozila u prometu. Problem usmjeravanja električnih vozila (EVRP) posebna je inačica problema usmjeravanja vo-

zila, koja koristi električna vozila. Razlika kod ove varijante problema VRP u odnosu na druge je ta što električna vozila imaju **ograničeni doseg**. To ograničenje postavljeno je zbog načina rada i punjenja električnih vozila. EVRP problem ne mora imati samo flotu električnih automobila. Flota može sadržavati i druga vozila. To je posebna inačica EVRP problema i naziva se EVRP s heterogenom flotom vozila [1]. Ovo je prednost u odnosu na obični EVRP problem, jer mogu postojati čvorovi do kojih električna vozila ne mogu doći. Također, mogu postojati scenariji u kojima je isplativije imati klasična vozila u odnosu na električna i obrnuto. Detaljnije o ovome bit će riječi kasnije. Slijedi opis pojedinih vrsta vozila.

2.2.1. Vozila s motorom s unutarnjim izgaranjem (ICE vozila)

Vozila s motorom s unutarnjim izgaranjem (ICE vozila) tip su vozila koja, kao izvor energije, koriste fosilna goriva. Prvo takvo vozilo konstruirao je i napravio 3. srpnja 1886. godine Karl Benz u Mannheimu [6]. Ova vozila su prošla kroz brojne iteracije i usavršavanja tijekom godina. Zbog toga nije začuđujuće da većina ljudi koristi upravo ovaj tip vozila.

ICE vozila imaju jednu veliku prednost u odnosu na električna vozila. Kao što i samo ime govori, ova vozila pokretana su motorom s unutarnjim izgaranjem goriva. Prednost leži u činjenici da je punjenje spremnika s gorivom kod ovih vozila izuzetno brzo. Također, vozila s motorom s unutarnjim izgaranjem pretežno imaju više mjesta nego električna vozila, jer nemaju bateriju koja zauzima dosta prostora te, uobičajeno, imaju veći doseg nego električna vozila.

Slika 2.2. 2018 Mazda3 [7]

Slika 2.3. 2024 Lexus LC500 [8]

Ipak, ova vozila nisu bez svojih mana. Glavna mana leži upravo u glavnoj prednosti.

Prilikom rada, ova vozila ispuštaju plinove, koji su štetni za okoliš. Uz to, tijekom usporavanja i zaustavnjanja, sva energija se pretvara u toplinu i gubi se.

Valja napomenuti da proizvođači ovih vozila rade na poboljšanju termanlne učinkovitosti motora, kako bi se smanjila potrošnja goriva i smanjio utjecaj ovih vozila na okoliš, prilikom njihovog rada. Jedan takav primjer je Toyota-in A25A, koji u regularnoj izvedbi ima termalnu učinkovitost do 40%, a u hibridnoj izvedbi do 41% [9].

Slika 2.4. Toyota A25A [9]

2.2.2. Električna vozila (EV)

Popularnost električnih vozila u današnje vrijeme raste, pogotovo u razvijenijim zemljama. Iako su postojala i ranije, radi problema s punjenjem baterija nisu zaživjela. Brojne su prednosti ovih vozila. Kao što je prethodno navedeno, električna vozila nemaju motor s unutarnjim izgaranjem te ne ispuštaju štetne plinove prilikom rada. Ova vozila još imaju bolji odziv te se mogu puniti i kod kuće, što korisnici smatraju glavnim prednostima. Dodatno, prilikom kočenja i zaustavljanja, koristi se elektromotor, koji pritom služi kao generator, prikuplja energiju kočenja te ju pretvara u električnu energiju, koja se sprema u bateriju vozila. Samim time produljen je vijek kočnica, jer se vrlo rijetko koriste. Zanimljiva je činjenica da se kočnice na ovim vozilima najčešće mijenjanju, ne zbog istrošenosti, nego radi hrđanja. Jedan od najpopularnijih proizvođača električnih vozila je američka tvrtka Tesla.

Ipak, postoje mane ovih vozila. Iako proizvođači rade na poboljšanjima, punjenje

Slika 2.5. Tesla Cybertruck [10]

baterija je još uvijek vrlo sporo (minimalno je potrebno pola sata). Također, baterije su kraćeg vijeka nego motori s unutarnjim izgaranjem te su vrlo skupe. Postoji i opasnost od zapaljenja baterije, no taj problem nije toliko raširen. Ova vozila su, zbog baterija, vrlo teška. Još jedna bitan problem predstavljaju punionice, kojih nema dovoljno te su redovi i vremena čekanja vrlo veliki. Doseg električnih vozila manji je nego vozila s motorom s unutarnjim izgaranjem te su ova vozila osjetljiva na temperaturu okoline. Također, ljudi kritiziraju električna vozila, jer smatraju da ova vozila "nemaju dušu", no to je kritika korisnika koji vole automobile.

2.2.3. Hibridna vozila (HEV)

Hibridna vozila danas dobivaju sve veću popularnost, što potvrđuje podatak da njihova prodaja raste i veća je nego prodaja električnih i vozila s motorom s unutarnjim izgaranjem. Prvo masovno proizvedeno hibridno vozilo je Toyota Prius. Ova vozila su vrlo popularna, jer rješavaju probleme koje imaju električna vozila, a ujedno nude velik broj prednosti električnih vozila. Naime, ova vozila pokretana su na dvije vrste pogona: motor s unutarnjim izgaranjem i elektromotor. Zbog toga je njihovo punjenje izrazito brzo, jer se pune "klasičnim" gorivom. Uz to, kod modernih hibridnih vozila, stražnji prostor je minimalno zahvaćen (baterije se najčešće nalaze ispod stražnjih sjedala) te nude mogućnost regenerativnog punjenja baterije prilikom kočenja. U slučaju da vozilo ostane bez

električne energije, vozilo će se moći dalje voziti, no isključivo na motor s unutarnjim izgaranjem.

Slika 2.6. 2024 Toyota Prius [11]

Glavna mana ovih vozila je činjenica da koriste dva izvora energije, a zbog baterija su teža nego ekvivalent samo s motorom s unutarnjim izgaranjem. Ljudi su nezadovoljni time, jer smatraju da ova vozila "nisu dobra klasična vozila, kao niti električna vozila", no to je kritika entuzijasta.

2.2.4. Plug-in hibridna vozila (PHEV)

Slika 2.7. 2024 Lexus NX450h+ [12]

Plug-in hibridna vozila nastala su kao odziv na popularnost električnih vozila. Ova su vozila skoro pa identična kao hibridna vozila, u smislu da imaju iste prednosti i mane. Dodatna prednost je veći kapacitet baterije, koja se sad nalazi ispod vozila. S povećanjem kapaciteta baterije, uvedena je i mogućnost punjenja baterija na punionicama te

je povećan električni doseg ovih vozila. Kako je kapacitet ovih baterija manji nego kod električnih vozila, trajanje punjenja je osjetno smanjeno, a za punjenje se može koristiti i motor s unutarnjim izgaranjem. Plug-in hibridna vozila nude "najbolje od oba svijeta", i od električnih vozila, i od vozila s motorom na unutarnje izgaranje te je njihova popularnost u najvećem rastu.

2.3. Korištena instanca EVRP-a

U ovom radu korištena je instanca EVRP-a [1] koja sadrži tri vrste vozila: vozila s motorom s unutarnjim izgaranjem goriva, električna vozila te plug-in hibridna vozila. Za vozila s motorom s unutarnjim izgaranjem vrijedi pretpostavka da ona ne moraju ići do postaja (punionica), jer im je doseg neograničen, dok električna vozila moraju. Svaki posjet postaji se naplaćuje vremenski. Plug-in hibridna vozila mogu, ali ne moraju posjetiti postaju. Kada se kapacitet vozila u potpunosti napuni, vozilo ide do sjedišta (čvor izvorište), kako bi se ostavio teret korisnika te se, ovisno o potrebi, ova vozila nastavljaju dalje koristiti. Važno je napomenuti da se, nakon obilaska svih korisnika, sva vozila vraćaju u čvor izvorište. Također, vozila nemaju isti kapacitet. Uz to, neke mape sadrže iste čvorove, no vozila su različitih karakteristika [1].

3. Metode za rješavanje problema usmjeravanja vozila

Kao što je prethodno rečeno, rješenje problema usmjeravanja vozila je put kojim vozila trebaju obići korisnike. To rješenje nije jedinstveno i postoji eksponencijalno mnogo rješenja. Postavlja se pitanje pronalaska optimalnog puta za pojedinu mapu korisnika. U svrhu pronalaska optimalnog puta koriste se algoritmi optimizacije. Jedan takav algoritam bit će opisan u nastavku. Dva najčešće korištena algoritma također su opisana i u mom završnom radu [3].

3.1. Evolucijsko računanje

Evolucijsko računanje posebna je grana računarstva. Definicija evolucijskog računarstva nije jedinstvena. Jedna moguća definicija glasi:

"evolucijsko računanje je područje računarske znanosti koje ramatra algoritme koji simuliraju evolucijski razvoj vrsta, život i ponašanje jedinke u društvu ili pak simuliraju različite aspekte umjetnog života." [13]

Karakteristično za ove algoritme je da su populacijski te da je najveći dio ovih algoritama nastao modeliranjem procesa koji su nastali u prirodi. Međutim, glavno, a za ovaj rad i najbitnije svojstvo ovih algoritama je mogućnost i prikladnost za rješavanje optimizacijskih problema, što je upravo EVRP problem kojim se ovdje bavimo. Važno je napomenuti da nisu svi evolucijski algoritmi prikladni za svaku vrstu optimizacijskog problema te stoga treba pažljivo odabrati koji algoritam koristiti u koju svrhu [13].

Evolucijsko računanje dijeli se na tri grane [13]:

- Evolucijski algoritmi (u ovoj skupini nalazi se i genetski algoritam)

- Algoritmi rojeva
- Ostali algoritmi

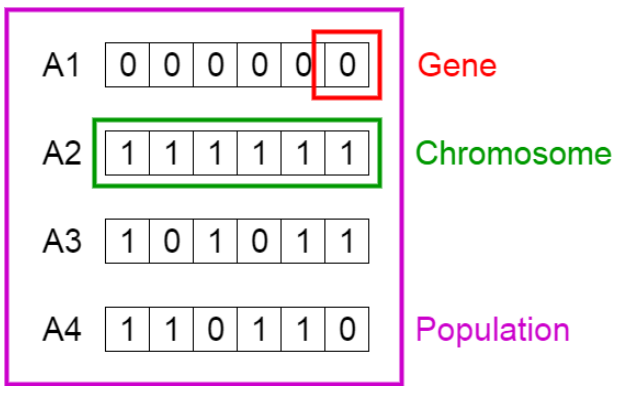
Za više informacija o evolucijskom računanju predlaže se proučavanje knjige "Neizrazito, evolucijsko i neuroračunarstvo" [13]. U nastavku će biti pobliže objašnjen vrlo često korišten algoritam za optimizaciju problema: genetski algoritam.

3.1.1. Genetski algoritam (GA)

Genetski algoritam heuristička je metoda optimiranja. Kao što je prethodno rečeno, ovaj algoritam spada u algoritme evolucijskog računanja. On je zapravo poopćenje evolucijskih strategija. Zbog njegove jednostavnosti i sposobnosti dobrog rješavanja optimizacijskih problema, vrlo često je korišten.

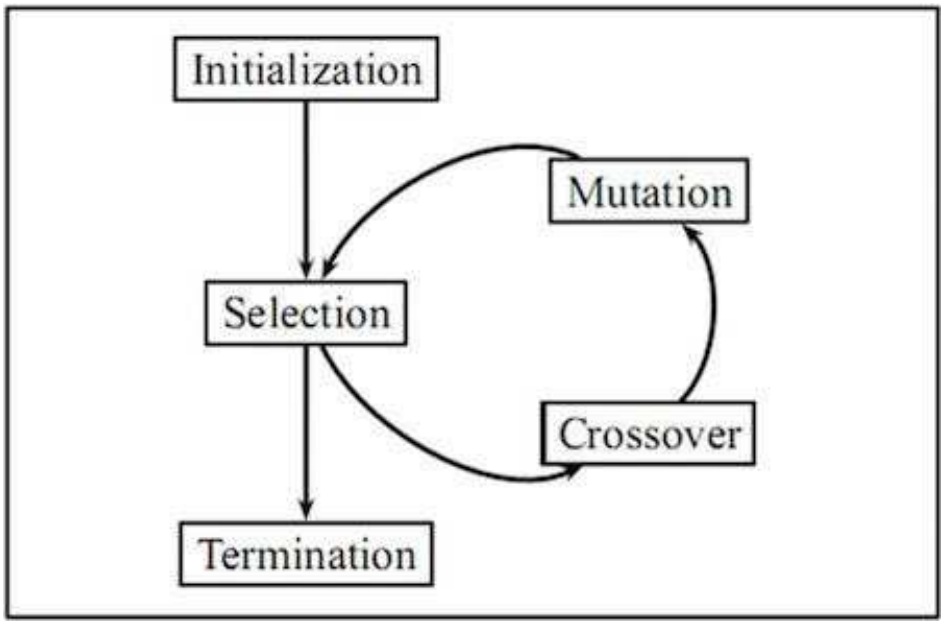
Genetski algoritam [14] inspiriran je istraživanjem biologa Charlesa Darwina o prirodnoj evoluciji i preživljavanju samo najboljih jedinki ("survival of the fittest"). Preživjele jedinke daju nove jedinke (potomke), koje ulaze u sljedeću generaciju algoritma i taj ciklus se ponavlja.

Svaki genetski algoritam sastoji se od populacije jedinki. Na kraju svake generacije, jedinke trenutne populacije daju nove jedinke za populaciju iduće generacije algoritma. Stoga se može zaključiti kako je temeljna jedinici za rad s ovim algoritmom jedinka. Svaka jedinka sadrži svoj genetski zapis, odnosno gene. Geni su najčešće predstavljeni matricom decimalnih brojeva, koja se još naziva i kromosom. Raspon u kojem se mogu kretati decimalni brojevi ovisi o problemu i konkretnoj implementaciji algoritma. Broj gena u kromosomu ovisi o problemu koji se rješava, a geni se još nazivaju težinama.



Slika 3.1. Ilustracija populacije genetskog algoritma sa svojstvima jedinki [14]

Na početku rada svakog genetskog algoritma, prvo se generira početna populacija. U ovom trenutku, kromosomi, odnosno geni jedinki generiraju se slučajno. Prilikom rada algoritma formirat će se geni, koji predstavljaju rješenje problema. Također, potrebno je definirati i funkciju dobrote. Kako samo ime kaže, ova funkcija govori koliko je pojedina jedinka dobra te je u direktnom odnosu s funkcijom troška. Postoje razne metode kako se može dobiti funkcija dobrote iz troška, a korištena metoda bit će definirana kasnije. Svaki genetski algoritam sastoji se od tri faze: selekcije, križanja i mutacije. Valja napomenuti da postoje razne izvedbe genetskog algoritma, koje se razlikuju u odabiru operatora selekcije, križanja i mutacije, no svima im je zajedničko da rade s populacijom jedinki.



Slika 3.2. Dijagram rada genetskog algoritma [15]

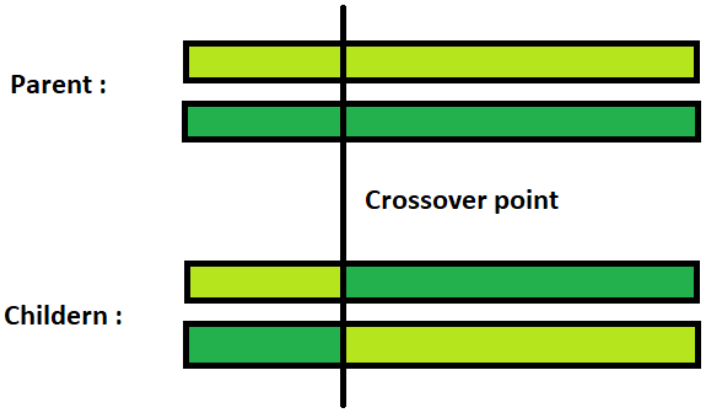
Faza selekcije

Faza selekcije zadužena je za odabir jedinki koje će se križati. Za odabir jedinki najčešće se odabiru one koje imaju najveću dobrotu i nazivaju se roditeljima. Odabrani roditelji koriste se za dobivanje nove jedinke, koja ide u iduću generaciju algoritma.

Faza križanja

Faza križanja [16] smatra se najbitnijim dijelom genetskog algoritma. U ovoj fazi koriste se roditelji odabrani u fazi selekcije. Kao produkt križanja, nastaje nova jedinka, koja sadrži kombinaciju gena obaju roditelja. Novonastala jedinka ulazi u iduću generaciju

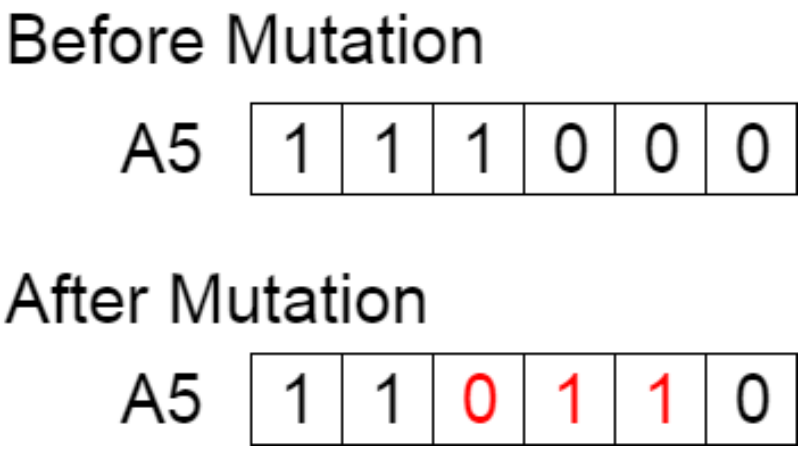
genetskog algoritma. Geni roditelja mogu biti decimalni brojevi, no mogu biti predstavljeni kao bitovi, čije su vrijednosti 0 i 1. U tom slučaju postoji točka prekida (crossover point). Točka prekida zapravo je nasumično odabrana cjelobrojna vrijednost. Tada se geni roditelja kombiniraju na način da se geni jednog roditelja kopiraju u novu jedinku do točke prekida te se ostatak gena nadopuni genima drugog roditelja.



Slika 3.3. Faza križanja [16]

Faza mutacije

Posljednja faza u generaciji genetkog algoritma jest faza mutacije. Ideja je promijeniti gene pojedinih jedinki, kako ne bi došlo do prerane konvergencije te kako bi se održala različitost unutar populacije. Valja napomenuti da neće kod svake jedinke doći do mutacije gena. Koliko često dolazi do mutacije u populaciji ovisi o vjerojatnosti mutacije.



Slika 3.4. Faza mutacije [14]

Detalji oko korištenih faza bit će navedeni u poglavlju 4.

Postoje tri uvjeta kojima se definira kraj rada algoritma:

- Dosegao se maksimalni broj iteracija
- Došlo je do konvergencije
- Desegnuta je dovoljno dobra dobrota.

Maksimalni broj iteracija i dovoljno dobru dobrotu zadaje korisnik. Kada dođe do konvergencije ili je pronađena jedinka dovoljne dobrote, možemo reći da je algoritam pronašao rješenje. Ako se dosegne maksimalni broj iteracija, ne možemo sa sigurnošću reći da je algoritam pronašao dobru jedinku. Tada bi bilo poželjno isprobati kako se algoritam ponaša, kada bismo povećali maksimalni broj iteracija.

4. Korištena implemetacija

U nastavku bit će objašnjen i detaljnije prikazan algoritam korišten za rješavanje konkretne instance EVRP problema 2.3. Također, bit će prikazani i opisani svi korišteni pseudokodovi istog algoritma.

4.1. Korišteni algoritam

Za potrebe rješavanja ove instance EVRP problema korišten je generacijski genetski algoritam. Kao što je ranije rečeno, različite implementacije genetskih algoritama razlikuju se po implementacijama operatora selekcije, križanja i mutacije. Općenito, svaki genetski algoritam prolazi kroz iteracije. Nakon svake iteracije stvaraju se nove jedinke koje onda tvore novu populaciju za iduću iteraciju. Problem koji može nastati prilikom treniranja ovog algoritma jest problem prenaučenosti, koji se očituje u nemogućnosti algoritma da dobro generalizira. Kako bismo to izbjegli, poželjno je podijeli skup podataka na skup za treniranje i skup za testiranje. Najčešća podjela podataka je 70:30, što znači da se 70% podataka koristi za treniranje, a 30% podataka za testiranje algoritma, što se može vidjeti na slici 4.1. Ta je podjela slučajna i događa se prilikom pokretanja algoritma. Zbog toga, potrebno je uvesti dva algoritma u genetski algoritam: algoritam za treniranje i algoritam za testiranje.

Slika 4.1. Podjela skupa podataka na skup za treniranje i skup za testiranje

Na početku algoritma, nužno je postaviti sve potrebne podatke. To uključuje: postav-

Algoritam 4.1: Pseudokod vanjske petlje genetskog algoritma

```
podijeli skup podatak na skup za treniranje i testiranje
postavi pocetne podatke algoritma
generairaj pocetnu populaciju
pokreni algoritam

dok je broj trenutne iteracije manji od maksimalnog broja
ili se ne postigne konvergencija ili nije pronadjena dovoljno
dobra jedinka:
    treniraj na skupu podataka za treniranje
    testiraj na skupu podataka za testiranje

pokreni najbolju jedinku za sve podatke i prikazi grafove
kraj
```

ljanje maksimalnog broja iteracija i broja jedinki unutar populacije. Dodatno, potrebno je postaviti operatore algoritma, koristi li se elitizam te funkcije troška i dobrote. Zatim je potrebno definirati početnu populaciju algoritma. Opisani pseudokod prikazan algoritmom 4.1 predstavlja vanjsku petlju genetskog algoritma. U nastavku će biti opisani dio algoritma za treniranje i testiranje, kao i svi korišteni operatori genetskog algoritma.

4.1.1. Treniranje algoritma

Algoritam za treniranje, kako samo ime govori, zadužen je za treniranje genetskog algoritma. U ovom dijelu dolazi do ažuriranja težina, odnosno gena jedinki 3.1.1., kako bi se dobila jedinka koja dovodi do dobrog rješenja danog problema. Kao što je ranije rečeno, skup podataka dijeli se na dva poskupa: skup za treniranje i testiranje podataka, čime se sprječava prenaučenost algoritma.

Prilikom treniranja prolazi se kroz podatke iz skupa podataka za treniranje i evaluiraju se jedinke, odnosno, računa im se dobrota. Nakon toga, počinju s radom ključni dijelovi genetskog algoritma: selekcija, križanje i mutacija. Na ovaj način nastaju nove jedinke koje idu u populaciju za sljedeću iteraciju. Svojstvo elitizma govori hoće li u iduću populaciju ući najbolja jedinka iz trenutne populacije. Ovo je vrlo korisno, kako bi se očuvalo najbolje rješenje trenutne iteracije. Međutim, elitizam može uzrokovati jedan bitan problem. Može se dogoditi da algoritam zapne u lokalnom optimumu i ne pronađe globalno rješenje problema. Kako bismo eliminirali ovaj problem, elitizam se

Algoritam 4.2: Pseudokod algoritma za treniranje

```
prodi kroz sve podatke iz skupa za treniranje i evaluiraj
dobrotu jedinki

sortiraj populaciju po dobroti od najbolje do najgore jedinke
nova populacija = []
ako je elitizam omogucen:
    dodaj najbolju jedinku u novu populaciju

dok velicina nove populacije < velicina populacije:
    odaberi jedinke
    kreiraj novu jedinku koristeci prethodno odabrane
    dodaj novogeneriranu jedinku u novu populaciju

mutiraj jedinke

postavi populaciju algoritma na novu populaciju
```

dinamički mijenja tijekom rada algoritma. Na početku je elitizam onesposobljen, kako bi se povećao prostor pretrage. Nakon određenog broja iteracija, elitizam se omogućava. Ako algoritam dugo ostane na jednom najboljem rješenju, elitizam se onesposobi, kako bi se potaknulo ponovno pretraživanje prostora rješenja te se potom elitizam opet omogućuje. Pseudokod zadužen za treniranje algoritma prikazan je u algoritmu 4.2 .

Algoritam selekcije

Prilikom procesa selekcije jedinki za križanje, koristi se troturnirska selekcija. Ideja je da se, na neki način odaberu tri jedinke, od kojih se potom uzimaju dvije najbolje jedinke, koje se koriste za križanje. Postoje razni načini na koje se mogu birati jedinke za troturnirsku selekciju. Jedna moguća metoda je "roulette wheel selection" [17]. Odabir jedinki odvija se na način da se odabire neki nasumični broj u rasponu između 0 i 1. Ovisno u kojem rasponu se taj broj nalazi, jedinka na odgovarajućem indeksu se odabire. Ovaj način odabira daje svakoj jedinki jednaku vjerojatnost odabira. Precizniji način bio bi kada bi vjerojatnost odabira jedinke ovisila o njenoj dobroti. Vjerojatnost odabira računa se formulom (4.1). Tada jedinke s većom dobrotom imaju i veći raspon odabira, a samim time i veću vjerojatnost odabira 4.2. Algoritam selekcije prikazan je algoritmom 4.3, a algoritam otežane roulette wheel selekcije prikazan je algoritmom 4.4 [17]

Algoritam 4.3: Pseudokod algoritma za selekciju

```
# jedinke moraju biti sortirane od najbolje do najgore
odabrane jedinke = []

dok nisu odabrane 3 jedinke:
    odaberi jedinku koristeci otezani roulette wheel selection

    ako jedinka nije u odabranim jedinkama:
        dodaj odabranu jedinku

vrati odabrane jedinke
```

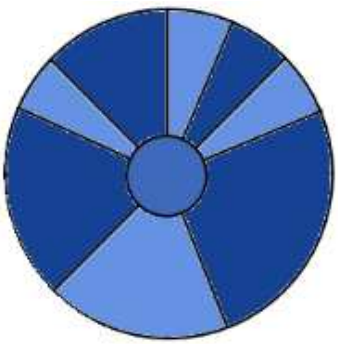
Algoritam 4.4: Pseudokod algoritma otežane roulette wheel selekcije

```
# jedinke moraju biti sortirane od najbolje do najgore
izracunaj sumu dobrota jedinki

za svaku jedinku u populaciji:
    izracunaj vjerojatnost odabira

odaberi nasumicni decimalni broj u intervalu [0, 1]
vrati jedinku cija vjerojatnost odabira odgovara odabranom
nasumicnom decimalnom broju
```

$$p_x = \frac{f_x}{\sum_{i=1}^{n} f_i} \tag{4.1}$$



Slika 4.2. Weighted roulette wheel selection [17]

Algoritam križanja

Algoritam križanja koristi se za stvaranje novih jedinki za iduću generaciju genetskog algoritma. Postoje brojni algoritmi križanja roditelja, no u ovu svrhu korištena je njegova jednostavnija verzija. Naime, odabiru se dvije od tri najbolje jedinke, koje se nazivaju

Algoritam 4.5: Pseudokod algoritma križanja

```
sortiraj roditelje po dobroti od najboljeg do najgoreg
odaberi dva od tri roditelja

geni nove jedinke = []
za index gena u genima:
    geni1 = geni prvog roditelja na indeksu index
    geni2 = geni drugog roditelja na indeksu index
    novi gen = (geni1 + geni2) / 2
    dodaj nove gene u gene nove jedinke

vrati novu jedinku s generiranim genima
```

Algoritam 4.6: Pseudokod algoritma mutacije

```
k = 1 / 100
sigma = inicijalni sigma * e^(-k * trenutni indeks iteracije)

za svaki gen u jedinki:
    dodaj na gen gaussovu mutaciju
```

roditeljima te koje će proizvesti novu jedinku. Težine, odnosno geni obaju roditelja se zbrajaju i normaliziraju te na taj način nastaju geni za novu jedinku, što je vidljivo u algoritmu 4.5

Algoritam mutacije

Mutacija je posljednji korak u svakoj iteraciji genetskog algoritma. Njena glavna uloga jest sprječavanje zaglavljivanja u lokalnom optimumu te povećanje prostora pretrage rješenja. No, mutaciju treba koristiti pažljivo. Najbolji način bio bi implementirati dinamičku mutaciju, čiji je raspon vrijednosti veći na početku algoritma te se s vremenom smanjuje. Na taj način osiguravamo veće pretraživanje prostora rješenja na početku algoritma, a poslije fino dorađujemo dobivena rješenja. U ovom radu korištena je dinamička Gaussova mutacija, prikazana algoritmom 4.6 . Valja napomenuti da neće svaka jedinka proći proces mutacije. To ovisi o vjerojatnosti mutacije, o kojoj je bilo više riječi u 3.1.1.

4.1.2. Testiranje algoritma

Nakon svake uspješne faze treniranja, slijedi faza testiranja. U tu svrhu koriste se algoritam za testiranje u kombinaciji s testnim skupom podataka. Važnost faze testiranja već je bila rečena. Testiranjem provjeravamo jesmo li pronašli model dovoljne točnosti te provjeravamo jesmo li prenaučili naš model. U fazi testiranja se ne mijenjaju geni jedinki niti se mijenja populacija. Dakle, faza testiranja služi samo za provjeru trenutnog rješenja na dosad **neviđenim podacima**.

4.1.3. Funkcija troška i funkcija dobrote

Funkcija troška i funkcija dobrote često su spominjane (3.1.1.), no nigdje nije objašnjeno kako su one definirane. Stoga će se, u ovom poglavlju, pobliže objasniti ove dvije funkcije.

Funkcija troška

Funkcija troška za zadaću ima računanje troška pojedine jedinke na određenom podatku. Logično je pretpostaviti da će bolje jedinke imati manji trošak, a lošije veći. Budući da postoje brojne instance VRP problema, može se zaključiti da postoje i drukčije funkcije troška. No, ovdje ćemo definirati funkciju koja opisuje ovu instancu EVRP problema. Budući da je doseg pojedinih vozila ograničen, jedinka bi trebala pokušati maksimalno iskoristiti dostupan kapacitet vozila, kako bi se smanjio put koji pojedino vozilo prijeđe. Također, kako imamo vozila koja sadrže bateriju, bitan je posjet postajama za punjenje. Punjenje traje određeno vrijeme, stoga jedinka treba pokušati minimizirati vrijeme punjenja. Dodatno, svako vozilo prijeđe određeni put, kojeg jedinka treba minimizirati, npr. jedinka bi trebala minimizirati broj posjeta postajama za punjenje te broj odlazaka u čvor izvorište. Svaki uvjet potrebno je povezati s pojedinim genom, odnosno težinom jedinke, kako bismo omogućili pravila rad algoritma, što je sažeto prikazano formulom (4.2).

$$trosak_x = gen_1 * udaljenost + gen_2 * vremenski\_trosak + \frac{1}{gen_3 * kapacitet\_korisnika} \tag{4.2}$$

Funckija dobrote

Funkcija dobrote u direktnom je odnosu s funkcijom troška. Postoje brojni načini kako se može izračunati dobrota iz troška, no ovdje je korišten jednostavniji pristup. Jednostavno se izračuna recipročna vrijednost troška te se na taj način dobije dobrota trenutne jedinke (4.3). To nam također omogućuje jednostavno dobivanje troška iz dobrote, ako se pojavi potreba za ponovnim izračunavanjem istoga.

$$dobrota_x = \frac{1}{trosak_x} \tag{4.3}$$

5. Rezultati i rasprava

Zadatak ovog rada bio je riješiti instancu EVRP problema prethodno opisanu u 2.3. koristeći genetski algoritam. Konkretna implementacija genetskog algoritam opisana je u poglavlju 4. Skup podataka sastoji se od 56 različitih mapa, svaka s 10 vozila različitih karakteristika. Taj skup se podijelio na skup za treniranje i testiranje, kako je opisano u 4.1. Algoritam je pokrenut s različitim brojem iteracija za treniranje, konkretno s četiri broja iteracija: 100, 500, 1000 i 5000. Iz rezultata je zaključeno da nije bilo potrebe pokretati algoritam na preko 5000 iteracija, je se rješenje pronađe i na manjem broju iteracija.

5.1. Arhitektura i vremena izvođenja

Za potrebe treniranja algoritma korišten je Appleov MacBook uređaj s Apple M1 procesorom ARM arhitekture, brizine 3.2 GHz. Ovo je bitan podatak u analizi performansi algoritamskog rješenja, jer je brzina izvođenja direktno povezana s uređajem na kojem se algoritam pokretao. Vremena izvođenja prikazana su u tablici 5.1.

Broj iteracija	Vrijeme izvođenja (HH:MM:SS)
100	00:06:14
500	00:27:03
1000	00:52:41
5000	01:35:50

Tablica 5.1. Vremena izvođenja za različite brojeve iteracija algoritma

Zanimljivo je vidjeti da prilikom pokretanja algoritma na 5000 iteracija, vrijeme izvođenja je svega 43 minute dulje nego na 1000 iteracija. To nas dovodi do zaključka da je zadovoljen jedan od uvjeta zaustavljanja genetskog algoritma: došlo je do konvergencije. Daljnjom analizom ustanovljeno je da je algoritam završio na 1703. iteraciji.

5.2. Rezultati

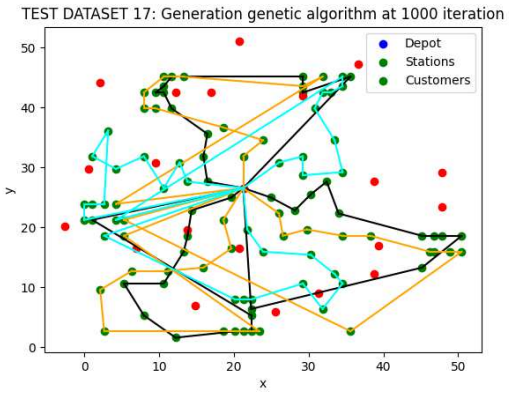
EVRP je tip problema koji nema jedinstveno rješenje. Njegovo rješenje je put kojim vozila trebaju obilaziti korisnike. Uz to, postoje i ograničenja, koja algoritma mora poštivati. Detaljnije o tome može se pročitati u 2.3. Kako je rješenje put, najlakše je interpretirati rezultate iz vizualnih podataka. Svaka vrsta vozila prikazana je drugom bojom. Kako postoje tri vrste vozila koja se ovdje koriste, tako se koriste i tri boje:

- crna boja - predstavlja ICE vozila (vozila s motorom s ununtarnjim izgaranjem)
- narančasta boja - predstavlja PHEV (plug-in hibridna vozila)
- cyan boja - predstavlja EV (električna vozila).

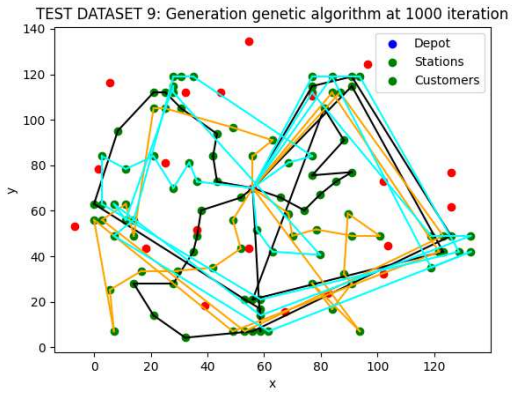
Vozila su grupirana na ovaj način, kako bi se mogli lakše izvući zaključci iz podataka. Glavno pitanje koje se postavlja prilikom prelaska na električna vozila jest: je li isplativo prijeći na ova vozila ili je možda bolje u nekim slučajevima imati i druge tipove vozila. Odgovor na ovo pitanje nije jednostavan. Međutim, rješavanje ovakvih problema algoritmima strojnog učenja može nam dati uvid u isplativost korištenja električnih i hibridnih vozila. Detaljnije o ovim vozilima može se vidjeti u poglavlju 2.1.

Kako u životu, tako i podaci koji su korišteni u ovom radu nisu jednako pogodni za električna vozila. No, kako zaključiti je li mapa pogodna za električno ili hibridno vozilo? Jedan način kako to zaključiti jest pogledati kako izgleda putanja kojom pojedina vrsta vozila putuju na određenoj mapi. Iz složenosti putanje se može zaključiti je li mapa pogodna za električna i hibridna vozila ili nije. Ako je put kompliciran i isprekidan te ako ima dosta posjeta izvorištu, može se zaključiti da mapa nije baš pogodna za određeno vozilo. Također, ako je putanja relativno jednostavna, može se zaključiti da vozila mogu dobro i jednostavno obići korisnike te je ta mapa pogodna za te tipove vozila. Jedna mapa koja je pogodna za električna vozila prikazana je slikom 5.1., dok je manje pogodna mapa prikazana slikom 5.2.

Također, na slikama 5.1. i 5.2. može se vidjeti da klasična ICE vozila, odnosno, vozila s motorom s unutarnjim izgaranjem nemaju problem kompliciranih ruta. To je zbog pretpostavke da ova vozila nemaju ograničenje koje korisnike mogu obići, zbog kapaciteta baterije te ne moraju posjećivati punionice. Tu se upravo ističe glavna mana električnih

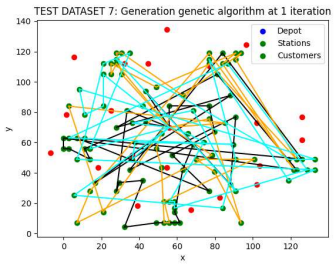


Slika 5.1. Pogodna mapa za EV

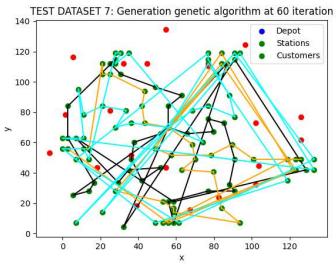


Slika 5.2. Manje pogodna mapa za EV

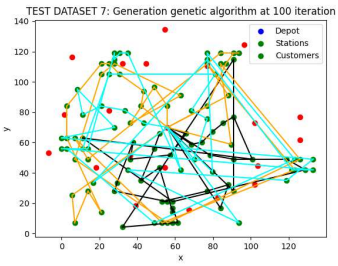
vozila: posjeti punionicama te nemogućnost obilaska pojedinih korisnika zbog njihovih ograničenja.



Slika 5.3. Test 1/100



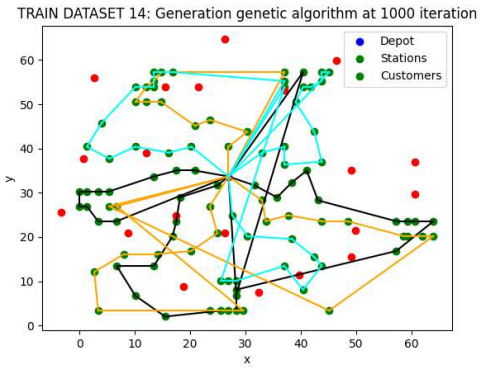
Slika 5.4. Test 60/100



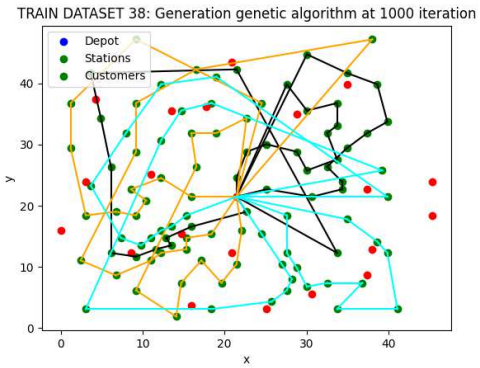
Slika 5.5. Test 100/100

Kako bismo se uvjerili da algoritam pokušava pronaći rješenje, prikazane su tri slike koje su pokrenute na 100 iteracija. Slika 5.3. prikazuje rješenje na početku treniranje, slika 5.4. na sredini treniranje i slika 5.5. prikazuje rješenje na kraju treniranja. Kao što se može vidjeti, put kojim vozila obilaze korisnike se mijenja. Bitno je napomenuti da ova mapa ne mora biti pogodna za sve tipove vozila te bi vjerojatno trebalo više iteracija, kako bi se pronašlo rješenje. No, ideja je bila prikazati proces treniranje algoritma u 3 trenutka, kako bi se dokazalo da se algoritam trenira. Također, potrebno je uvjeriti se da ovaj algoritam radi i na skupu za treniranje. Slike 5.6. i 5.7. prikazuju rezultate za dvije mape odabrane iz skupa za treniranje.

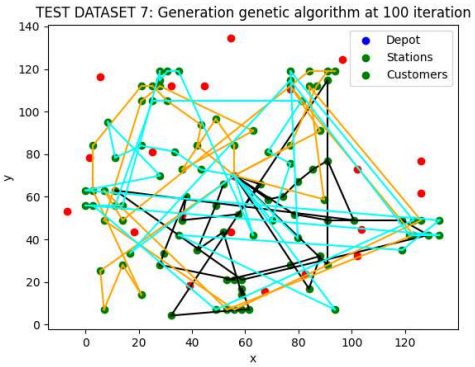
Dodatno, kako bismo se uvjerili da algoritam može pronaći rješenje, ako se zada dovoljan broj iteracija, prethodno korištena mapa sa slike 5.8. prikazana je i za algoritam koji je pokrenut na 1000 iteracija te je njeno rješenje vidljivo na slici 5.9. Uspoređujući te dvije slike, vidimo da je algoritam u mogućnosti pronaći bolje rješenje, ako mu se zada dovoljan broj iteracija.



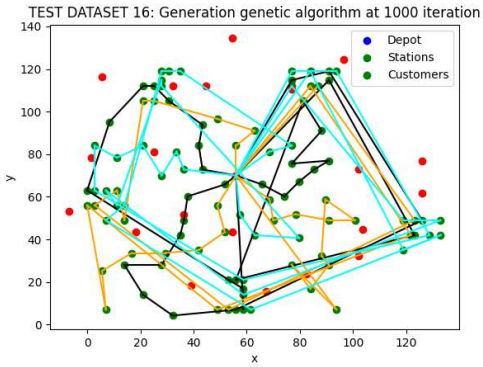
Slika 5.6. Train dataset 14, 1000 iteracija



Slika 5.7. Train dataset 38, 1000 iteracija



Slika 5.8. Test dataset, 100 iteracija



Slika 5.9. Test dataset, 1000 iteracija

Dobrote najboljih dobivenih jedinki za 100, 500, 1000 i 5000 iteracija prikazane su u tablici 5.2., gdje se može uočiti da s povećanjem broja iteracija raste i dobrota jedinki te da je dobrota jedinke veća na skupu za treniranje, nego na skupu za testiranje, što je i očekivano.

Broj iteracija	Dobrota na skupu za treniranje	Dobrota na skupu za testiranje
100	109.45	47.55
500	122.23	53.26
1000	147.49	64.46
5000	180.66	80.28

Tablica 5.2. Dobrote najboljih jedinki po iteracijama algoritma

Sudeći po prethodno opisanim podacima, možemo zaključiti da ovom algoritmu na ovim podacima treba oko 1000 iteracija, kako bi se uspješno istrenirao. Također, korištenjem skupa podataka za treniranje i testiranje, osiguralo se da ne dođe do prenaučenosti algoritma te da isti može raditi i na dosad neviđenim podacima, odnosno da dobro generalizira.

6. Zaključak

Problem usmjeravanja vozila vrlo je zanimljiv problem zbog svoje primjene u svakodnevnom životu, kompleksnosti i nejednoznačnosti rješenja. U današnje vrijeme ljudi žive užurbanim načinom života i često se oslanjaju na dostave. Uz to, područje umjetne inteligencije i strojnog učenja postaje sve popularnije. Također, vlade brojnih država pokušavaju povećati udio električnih vozila te stoga postaje sve bitnija posebna instanca VRP problema: EVRP.

Rješavanjem ovog problema dobije se optimalni put kojim treba obići korisnike. Kako taj put nije jedinstven, ovaj problem rješava se algoritmima optimizacije. Jedan takav algoritam je genetski algoritam, opisan u ovom radu. Ideja je trenirati algoritam na mapama korisnika te ga potom testirati, kako bismo dobili mogućnost pronalaska optimalnog puta za dosad neviđene mape.

No, električna vozila nisu uvijek dobar izbor. Postoje mape na kojima se može vidjeti da bi bilo bolje koristiti vozila s motorom s unutarnjim izgaranjem ili hibridna vozila. Rješavanje ove instance EVRP problema može nam reći kada bi bilo korisno upotrebljavati električna vozila, a kada ne.

Kako se genetski algoritmi razlikuju po svojima karakteristikama, tako se zaključuje da nije svaka konfiguracija tog algoritma pogodna za svaki problem. Kao nastavak ovog rada mogu se isprobati druge konfiguracije genetskog algoritma te se potom mogu usporediti dobiveni rezultati i zaključiti koja bi konfiguracija bila bolja za ovaj problem.

Literatura

[1] G. Hiermann, R. F. Hartl, J. Puchinger, i T. Vidal, “Routing a mix of conventional, plug-in hybrid, and electric vehicles”, *European Journal of Operational Research*, sv. 272, br. 1, str. 235–248, 2019. https://doi.org/https://doi.org/10.1016/j.ejor.2018.06.025

[2] P. Toth, D. Vigo, P. Toth, i D. Vigo, *Vehicle Routing: Problems, Methods, and Applications, Second Edition*. USA: Society for Industrial and Applied Mathematics, 2014.

[3] T. Jukić, “Rješavanje problema usmjeravanja vozila korištenjem neuronskih mreža”, *N/A*, 2022.

[4] C. M. Frey, A. Jungwirth, M. Frey, i R. Kolisch, “The vehicle routing problem with time windows and flexible delivery locations”, *European Journal of Operational Research*, sv. 308, br. 3, str. 1142–1159, 2023. https://doi.org/https://doi.org/10.1016/j.ejor.2022.11.029

[5] A. Gupta i S. Saini, “An enhanced ant colony optimization algorithm for vehicle routing problem with time windows”, 12 2017., str. 267–274. https://doi.org/10.1109/ICoAC.2017.8441175

[6] Wikipedia, https://hr.wikipedia.org/wiki/Automobil, [zadnje osvježeno: 24. travna 2024.].

[7] Jack Healy, https://www.carkeys.co.uk/car-reviews/2018-mazda-3-review, [10. siječnja 2018.].

[8] Lexus Canada, https://media.lexus.ca/en/releases/2023/2024-lexus-lc-500-inspiration-series.html, [godina modela: 2024.].

[9] Eugenio,77, https://toyota-club.net/files/faq/18-03-20_faq_df_r4_eng.htm, [u upotrebi od srpnja 2018. do danas].

[10] Eva Benedik, https://www.telegram.hr/zivot/tesla-je-priprijetila-da-ce-tuziti-vozace-koji-preprodaju-cybertruck-u-prvoj-godini-vlasnistva-sada-su-to-uklonili/, [15. studenoga 2023.].

[11] Toyota, https://www.toyota.com/prius/, [godina modela: 2024.].

[12] Lexus Ireland, https://www.lexus.ie/plug-in-hybrid, [godina modela: 2024.].

[13] M. Čupić, B. D. Bašić, i M. Golub, *Neizrazito, evolucijsko i neuroračunarstvo.* Marko Čupić, Bojana Dalbelo Bašić, Marko Golub, 2013.

[14] Vijini Mallawaarachchi, https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3, [8. srpnja 2017.].

[15] Nnamdi Okpala, https://medium.com/@okpalan86/demystifying-evolutionary-algorithms-c3a8fd154aa4, [14. srpnja 2023.].

[16] GeeksforGeeks, https://www.geeksforgeeks.org/crossover-in-genetic-algorithm/, [zadnje osvježeno: 1. ožujka 2023.].

[17] Gabriele De Luca, https://www.baeldung.com/cs/genetic-algorithms-roulette-selection, [zadnje osvježeno: 18. ožujka 2024.].

Sažetak

Rješavanje problema usmjeravanja vozila s heterogenim flotama korištenjem genetskog algoritma

Tin Jukić

Problem usmjeravanja vozila (VRP) sve je popularniji problem u domeni strojnog učenja. Postoje brojne instance VRP-a, a jedna od njih je EVRP, koja sadrži plug-in hibridna i električna vozila. Opisan je EVRP problem te sva vozila koja se u njemu koriste. Dodatno, opisan je genetski algoritam (GA) i njegova konfiguracija koja se u ovom radu koristila. Prikazani su dobiveni rezultati te su izvučeni odgovarajući zaključci. Uz to, objašnjeno je zašto električna vozila nisu pogodna za svaku mapu te kada bi bilo bolje koristiti alternative.

Ključne riječi: problem usmjeravanja vozila; problem usmjeravanja električnih vozila; vozila s motorom s unutarnjim izgaranjem; plug-in hibridna vozila; električna vozila; strojno učenje; genetski algoritam

Abstract

Solving vehicle routing problem with mixed fleet using genetic algorithm

Tin Jukić

Vehicle routing problem (VRP) is a more and more popular problem in machine learning domain. There are many instances of VRP, and one of them is EVRP, which contains plug-in hybrid and electric vehicles. EVRP problem is described, alongside all the vehicles that are used in it. Furthermore, genetic algorithm (GA) is also described with its configuration in this article. All results are shown and appropriate conclusions are drawn. Alongside that, here is described why electric vehicles are not suitable for all maps, and when the alternatives should be considered.

Keywords: vehicle routing problem; electric vehicle routing problem; internal combustion engine vehicles; plug-in hybrid vehicles; electric vehicles; machine learning; genetic algorithm