Interaktivna web aplikacija za rješavanje problema usmjeravanja vozila

Vrbanc, Lucian

Sveučilište u Zagrebu

Fakultet prometnih znanosti

ZAVRŠNI RAD

INTERAKTIVNA WEB APLIKACIJA ZA RJEŠAVANJE PROBLEMA USMJERAVANJA VOZILA

INTERACTIVE WEB APPLICATION FOR SOLVING VEHICLE ROUTING PROBLEM

Mentor: dr. sc. Tomislav Erdelić

Student: Lucian Vrbanc

JMBAG: 0135257990

Zagreb, rujan 2022.

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI
ODBOR ZA ZAVRŠNI RAD

Zagreb, 5. svibnja 2022.

Zavod: **Zavod za inteligentne transportne sustave**
Predmet: **Optimizacija prometnih procesa**

ZAVRŠNI ZADATAK br. 6902

Pristupnik: **Lucian Vrbanc (0135257990)**
Studij: Inteligentni transportni sustavi i logistika
Smjer: Inteligentni transportni sustavi

Zadatak: **Interaktivna web aplikacija za rješavanje problema usmjeravanja vozila**

Opis zadatka:

U radu je potrebno objasniti i primijeniti heurističke algoritme za rješavanje problema usmjeravanja vozila. Potom je potrebno implementirati neke od odabranih algoritama za rješavanje problema te izraditi web sučelje za prikaz rješenja problema. Dodatno potrebno je analizirati mogućnosti nadogradnje web sučelja s interaktivnom komponentom, odnosno da korisnik može sam definirati problem i ograničenja problema. Na kraju je potrebno analizirati rješenja problema dobivena različitim algoritmima te ih usporediti.

Mentor:

Tomislav Erdelić

dr. sc. Tomislav Erdelić

Predsjednik povjerenstva za završni ispit:

Interaktivna web aplikacija za rješavanje problema usmjeravanja vozila

Sažetak:

Područje inteligentnih transportnih sustava je opširno i bavi se rješavanjem velikog broja problema koji postoje danas u prometu i transportu. Jedan od tih problema je i problem usmjeravanja vozila. Rješenje samog problema je predstavljeno kao skup ruta kojima se prometni entitet kreće do početne do završne lokacije. Navedeni problem može se matematički modulirati koristeći teoriju grafova. Problem dodatno otežavaju uvjeti u realnom svijetu gdje postoje određeni vremenski prozori, različita potražnja korisnika, dostava i prikupljanje, stohastičnost prometa kao cjelina i sl. Ovaj rad prikazuje web aplikaciju koja rješava problem usmjeravanja vozila sa ograničenjem kapaciteta koristeći VNS i lokalnu pretragu izrađenu pomoću razvojnog okruženja PyCharm, programskog jezika Python i web okvira Flask. Sučelje se sastoji od interaktivne web karte te od bočne trake koja sadrži padajuću traku iz koje se odabire željeni testni problem, instrukcija za korištenje web aplikacije i gumba koji pokreće algoritam.

Ključne riječi: Problem usmjeravanja vozila, web aplikacija, heuristički algoritmi, PyCharm, Python, Flask

Interactive web application for solving vehicle routing problem

Abstract:

The field of intelligent transportation systems is vast and it deals with solving a large number of problems in traffic and transportation. One of such problems is the vehicle routing problem. The solution of the problem itself is presented as a set of routes within which the traffic entity moves from origin to destination. Using the graph theory the problem can be mathematically modeled. The problem is further aggravated by real world conditions such as time windows, different user demand, pickup and delivery, stohastic nature of traffic etc. This paper shows a web application that solves capacitated vehicle routing problem using VNS and local search algorithms that is made in PyCharm development environment, Python programming language and web framework Flask. The interface consists of interactive web map and a sidebar that has a drop-down bar within which are some test problems , instructions on how to use the web application and a button that starts the algorithm are located.

Key words: Vehicle routing problem, web application, heuristic algorithms, PyCharm, Python, Flask

Sažetak

1. Uvod

Krajnji cilj u transportu tereta kao i ljudi je u što manje vremena prevesti što veću količinu uz što manji trošak. Iz toga razloga se još uvijek razvija nova matematika te njezina implementacija u različitim sektorima industrije, a jedan od njih je i promet. U prometu takav problem pripada području Inteligentnih transportnih sustava (eng. „Intelligent Transport Systems“, ITS) i logistike. Komponente problema usmjeravanja vozila sastoje se od ruta kojima prometni entitet se kreće do određenog cilja, vrijeme putovanja te ukupna pređena udaljenost. Što se tiče samog prometnog entiteta on isto ima svoje karakteristike koje se uzimaju u obzir kao što su volumni i maseni kapacitet, nehomogenost voznog parka itd. Jedan od glavnih problema današnjice jeste zagađenje uzrokovano prometnim entitetima. Razvojem novih tehnologija kao što su namjenske komunikacije kratkog dometa (eng. „Dedicated short-range communication“ DSRC) bežične komunikacije koje omugućuju komunikaciju vozilo – vozilo (eng. „Vehicle-to-vehicle“, V2V), vozilo-infrastruktura(eng. „vehicle-to-infrastructure“ V2I) te infrastruktura-vozilo (eng. „Infrastructure-to-vehicle“ I2V) komunikaciju će omogućiti ITS sustavim da na temelju trenutačnih podataka u usporedbi sa prijašnjim podacima omogući puno efikasnije rutiranje vozila. Istraživanja pokazuju kako u Sjedinjenim Američkim Državana 50% svog nastalog zagušenja na prometnicama je ne ponovljivo odnosno nije bilo moguće ga predvidjet temeljno na prijašnjim podacima čineći rješenja koja se baziraju na prijašnjim podacima u 50% slučajeva ne upotrebljivima. Gleda se kako će ove tehnologije osim implementacije električnih vozila, najviše utjecat na zagađenje prometnih entiteta nastalo prilikom zagušenjem prometnice, [1].

U ovom radu se opisuju vrste VRP (eng. „Vehicle routing problem“, VRP) problema, načini na koji se rješavaju ti problemi sa fokusom na heurističke metode rješavanja problema koji se ujedno i koristi u programskom dijelu ovoga rada. Heuristički algoritmi koji se koriste u radu su Sweep algoritam i Clark and Wright algoritam.

U drugom dijelu rada objašnjava se problematika VRP problema kao i neke od njegovih podvrsta. Jedno poznatije pojednostavljenje problema VRP-a je problem trgovačkog putnika (eng. „Travelling Salesman Problem“, TSP) u kojem je cilj obići sve zadane gradove samo jednom i vratiti se nazad u početni grad bez korištenja istog puta između dvaju gradova dva puta.

U trećem dijelu rada objašnjava se heuristički pristup rješavanja problema VRP-a te su objašnjeni algoritmi koji se koriste u web aplikaciji za rješavanje problema VRP-a. Heuristički algoritmi za razliku od egzaktnih ne daju nužno optimalno rješenje već fokus je na brzini izvršavanja sa prihvatljivim rješenjem.

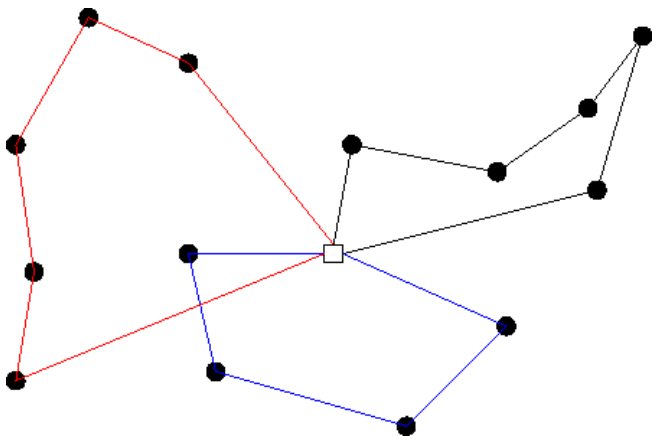
U četvrtom dijelu rada opisuje se izrada web aplikacije to jest dijelovi i tehnologije od kojih se sastoji kao i njihova pojašnjenja. Za potrebe izrade web aplikacije koristio se programski jezik Python sa web okvirom Flask.

U petom dijelu rada opisuje se način rada web aplikacije koji je popraćen sa dijagramom djelovanja.

U šestom dijelu rada donesen je zaključak na tematiku rada.

2. Problem usmjeravanja vozila

Problem usmjeravanja vozila je problem iz optimizacijske kombinatorike i cjelobrojnog programiranja koji postavlja pitanje „Koji je optimalni skup ruta za flotu vozila koja ih prolazi s ciljem dostave određenom skupu kupaca?“. Prvo spominjanje problema pojavljuje se u radu od George Dantzig-a i John Ramser-a 1959. godine u kojem je napisan prvi algoritamski pristup rješavanja problema[2]. Njegova primjena je bila namijenjena za dostavu goriva. Najčešća problematika VRP-a vezana je uz kontekst dostave određenog dobara iz određenog skladišta kupcima koji su naručili to dobro. Zadaća VRP-a je minimizirati ukupan trošak rute. 1964. godine Clarke i Wright su poboljšali Dantzig-ov i Ramser-ov pristup koristeći učinkoviti pohlepni algoritam nazvan algoritam uštede (eng. „the saving algorithm“)[3]. Određivanje optimalnog rješenja za VRP problem je NP teško[4]. To znači da veličina problema koje može biti riješeno optimalno koristeći matematičko programiranje ili kombinatornu optimizaciju je limitirano. VRP problem ima mnogo primjena u industriji. Korištenje računalnih optimizacijski programa mogu pomoći kompanijama koje se bave transportom dobara kako bi uštedili 5 do 30%. Čitavi transporti sektor čini 10% cijelog EU BDP-a, [4].



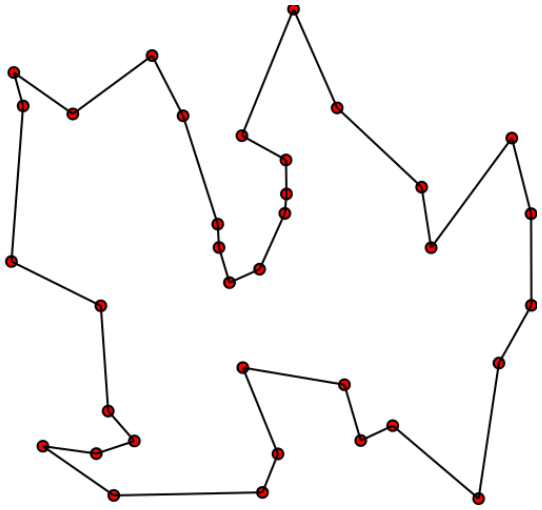
Slika 1. Grafički primjer VRP problema

Izvor:[5]

Iz slike 1. je vidljiv bijeli kvadratić u sredini grafičkog prikaza koji predstavlja skladište iz kojeg proizlaze linije. Crnim točkama su predstavljeni korisnici, a obojane linije predstavljaju rutu kojom se vozilo kreće do tih korisnika. Razlikuju se 3 linije: crvena, plava i crna. Svaka od njih predstavlja jednu rutu kojom vozilo se kreće. Može se primijetiti kako svaka linija dva puta ulazi to jest izlazi iz kvadratića što predstavlja izlaz vozila iz skladišta te njegovo vraćanje nazad.

2.1 Problem trgovačkog putnika

Problem trgovačkog putnika (eng. „Travelling Salesperson Problem“ TSP) koje pita pitanje: „U danom popisu gradova i udaljenosti između svakog para gradova, koja je najkraća moguća ruta koja posjećuje svaki grad točno jednom i vraća se u grad iz kojeg se krenulo?“. TSP je kategoriziran kao NP - težak problem u kombinatornoj optimizaciji. U računskoj teoriji složenosti TSP problem pripada NP potpunim problemima. To znači da se procesno vrijeme bilo kojeg algoritma za TSP superpolinomno povećava sa brojem gradova. Problem je prvi put formuliran 1930. godine i jedan je on najpoznatijih i najistraženijih optimizacijski problema. Često se koristi kao mjerilo za optimizacijske metode. Iako je problem računski teško rješiv poznati su brojni heuristički i egzaktni algoritmi koji ga mogu potpuno riješit, ili u slučaju velikog broja gradova rješenje je približno dano u 1% percentilu, [6].



Slika 2. Grafički prikaz rješenja TSP problema

Izvor: [7]

Na slici 2. gradovi su prikazani crvenim kružićima i spojeni crnom linijom. Vidljivo je da se je vozilo kroz svaku točko prošlo točno jednom te da je put zatvoren.

2.2 Vrste VRP problema

Zbog velike opširnosti VRP problema, njegove primjene u različitim industrijama, različitim ograničenjima i radi pojednostavljenja samog problema razlikuju se brojne varijante i specijalizacije VRP problema.

2.2.1 Problem usmjeravanja vozila s ograničenjem kapaciteta

Problem usmjeravanja vozila s ograničenjem kapaciteta (eng. „Capacitated Vehicle Routing Problem“, CVRP) odnosi se na VRP problem s ograničenjem kapaciteta vozila. Smatra se osnovnim tipom problema usmjeravanja vozila. Problem se sastoji skladišta iz kojeg svako vozilo kreće i završava, korisnika do kojeg se treba izvršit dostava te samog ograničenja kapaciteta vozila. Cilj je pronaći rutu koja ima minimalni trošak, a da je svaki korisnik poslužen. [11]

2.2.2 Problem usmjeravanja vozila s vremenskim prozorima

Problem usmjeravanja vozila s vremenskim prozorima (eng. „Vehicle Routing Problem with Time Windows“ VRPTW) je proširenje CVRP problema s vremenskim ograničenjima u kojima roba mora biti dostavljena korisniku. Svako vozilo ima limitirani kapacitet. Počinje se iz skladišta i završava se u skladištu. Svaki korisnik mora biti poslužen točno jednom. Cilj je minimiziranje transportnog troška, [10].

2.2.3 Problem usmjeravanja vozila sa dobiti

Problem usmjeravanja vozila sa dobiti (eng. „Vehicle Routing Problem with Profits“ VRPP) se razlikuje od ostalih VRP problema po tome što je svakom korisniku pridodana numerička dobitna vrijednost koja se daje vozilu prilikom posjete i to što nije nužno posjetiti sva odredišta. Dakle umjesto minimiziranja ukupnog pređenog puta kroz rute koje posjećuju svakog korisnika, VRPP problem nastoji odrediti koje će korisnike i njihove rute posjetiti po najvećem mogućem iznosu koji može dobit od korisnika pritom zadovoljavajući uvjete kao što su potražnja [8].

2.2.4 Problem usmjeravanja vozila s prikupljanjem i dostavom

Problem usmjeravanja vozila s prikupljanjem i dostavom (eng. „VRP with Pickup and Delivery“, VRPPD) je proširenje VRP problema gdje se osim standardnog zahtjeva za transport robe od skladišta do korisnika, sada korisnik može imati robu koju vozilo mora pokupiti i dostaviti drugom korisniku. Cilj je minimiziranje ukupne prijeđene udaljenosti[9].

2.2.5 Problem usmjeravanja vozila s zadnji unutra prvi van

Problem usmjeravanja vozila s zadnji unutra prvi van (eng. „Vehicle Routing Problem with Last-In-First Out“ VRPLIFO) sličan je VRPPD problemu, ali sa dodatnim ograničenjem vezano za utovar vozila. Na bilo kojoj dostavnoj lokaciji roba koja se dostavlja mora biti roba koja je zadnje bila utovarena. Ovaj problem smanjuje vrijeme utovara i istovara na dostavnim lokacijama jer nema privremenog istovara robe osim one koje mora biti dostavljena na toj lokaciji, [4]

2.2.6 Otvoreni problem usmjeravanja vozila

Otvoreni problem usmjeravanja vozila (eng. „Open Vehicle Routing Problem“, OVRP) je varijanta VRP problema koja se razlikuje od standardnog VRP problema tako što u njemu vozila ne moraju vratiti u skladište ili se moraju vratiti tako da posjete određene korisnike u obrnutom redoslijedu. Dakle vozne rute nisu zatvoreni putevi već otvoreni, [12].

2.2.7 Problem usmjeravanja vozila s dostavom i povratnim prikupljanjem

Problem usmjeravanja vozila s dostavom i povratnim prikupljanjem (eng. „VRP with Backhauls“, VRPB) je problem sličan VRPPD problemu. Ključna stvar kod VRPB problema je ta da na svakoj ruti prvo moraju biti obavljene dostave prije bilo kakvog prikupljanja. Prikupljena roba se vraća nazad u skladište. Količine koje se dostavljaju i prikupljaju su fiksne i znaju se unaprijed. Cilj je minimizirati ukupni prijeđeni put uz dana ograničenja, [13].

2.3 Matematički model CVRP-a

Sukladno onome što je već prije objašnjeno vezano uz CVRP, problem matematički opisujemo na slijedeći način. Potpuni neusmjereni graf G se zapisuje kao $G = (V, E)$. $E = \{(i,j) : i,j \in V, i < j\}$ predstavlja skup bridova koji međusobno povezuju čvorove, a $V = \{0, \dots, n\}$ predstavlja skup svih čvorova. Korisnik je predstavljen čvorom $i \in V \setminus \{0\}$ s negativnom potražnjom q_i, a skladište s čvorom indeksa 0. Svaki luk $e \in E$ ima pridruženu težinu c_e ili c_{ij}. Problem se može predstaviti tako da se za flotu veličine m vozila koji dijele jednake kapacitete Q treba odrediti m broj ruta koje će ukupni trošak putovanja uz uvjete da svaka ruta počinje i završava u skladištu svesti na minimum, da svaki korisnik pripada točno jednoj ruti te da ukupna potražnja svih korisnika u ruti ne prelazi ograničenje kapaciteta vozila. Problem se prikazuje kao problem trgovačkog putnika kada je $m = 1$. To znači da postoji samo jedno vozilo koje posjećuje sve korisnike u zadanom problemu. Minimalan broj vozila potreban za rješavanje CVRP problema određuje se izrazom (1), [14].

$$m = \left[\frac{\sum_{i=1}^{n} q_i}{Q} \right] \tag{1}$$

3. Heuristički postupci rješavanja problema

Heuristike su metode koje se koriste za rješavanja problema koje se temelji na nekom iskustvu. Pomoću heuristike se pronalaze optimalna rješenjate ovisno o problemu mogu procjeniti odluku. Heuristički algoritmi se najčešće primjenjuju u rješavanju optimizacijskih problema za koje možemo dokazati da je njihova kompleksnost NP teška. Ono što čini heurističkih algoritama boljim od nekih jednostavnijih algoritama jest to da smanjuje prostor pretrage koristeći neke iskustvene spoznaje, te time znatno ubrzava proces pronalaženja rješenja. To znači da heuristički algoritmi ne moraju uvijek davati optimalno rješenje te ponekad njihovo izvođenje može trajati duže od algoritma koji koristi egzaktnu metodu rješavanja problema. Dva su osnovna uvjeta koje algoritam mora zadovoljiti ukoliko se želi pronaći globalni optimum. To su uvjeti stabilnosti (stabilizaciju u globalnom optimumu) i uvjeti oslobađanja iz lokalnog optimuma odnosno bijeg iz lokalnog optimuma, [15].

3.1 Pohlepni algoritam

Pohlepni algoritam (eng. „Greedy algorithm“) je heuristički algorirtam koji lokalno odabire najbolji odabir u danom trenutku. On ne razmatra da li će trenutačno najbolje rješenje dovesti do globalnog optimuma. Algoritam nikad ne promjeni prijašnju odluku iako je odabir krivi. Djeluje u pristupu odzgo prema dolje. Stoga pohlepni algoritam neće uvijek naći optimalno rješenje, no međutim znatno je brži od nekih drugih algoritama čineći ga dobrim odabirom za korištenje kao algoritam za prvotno rješenje koje se naknadno može optimizirati, [16].

Premda pohlepni algoritam neće uvijek voditi ka globalnom optimumu, njegovo rješenje i dalje može biti korisno u razmatranju NP-teških problema. Naime, pohlepni pristup vodi do rješenja koje sigurno predstavlja donju granicu globalnog maksimuma, ukoliko se traži maksimum, odnosno gornju granicu globalnog minimuma, ukoliko se traži minimum. Kako se pohlepni algoritam brzo izvodi i ograđuje nam optimalno rješenje, on nam omogućava da uz pomoć drugih algoritama smanjimo prostor pretrage rješenja. Dodatna prednost pohlepne strategije je da ukoliko ne pronađe

optimalno rješenje dobiveno rješenje često vodi na novu strategiju razvoja algoritma koja onda rezultira efikasnijim rješenjem ili tehnikom koja će bbrzo pronaći bolju aproksimaciju rješenja. Opis rada heuristike koju koristi pohlepni algoritam je jednostavna: pronađi najbolje trenutno rješenje i idi za njim. Postoje mnogi algoritmi koji su zasnovani na pohlepnom pristupu, pa kada govorimo o pohlepnom algoritmu, ustvari govorimo o većoj skupini algoritama, [15].

3.2 Lokalno pretraživanje

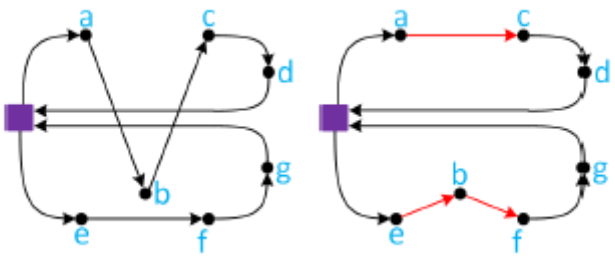
Kao i ostali heuristički algoritmi, tako i lokalno pretraživanje (eng. „Local search“, LS) pokušava pronaćio optimalno rješenje zadanog problema smanjujući prostor pretraživanja. Rad LS algoritma možemo opisati u nekoliko koraka. Jedan od zahtijeva je da u svakom trenutku raspolažemo jednim dopustivim rješenjem. Prvo se odabere rješenje i odredi se njegova vrijednost funkcije cilja. Zatim promatramo okolinu (susjedstvo) od odabranog rješenja i u ovisnosti o njegovim vrijednostima se izabere najbolji ili koji je bolji od trenutačnog. Susjedi su pri tome definirani ovisno o problemu koji se promatra. Izabrano rješenje postaje novo trenutačno najbolje rješenje. Zatim se nizom iteracija nastoji pronaći optimalno rješenje. Trajanje algoritma je diskretno određeno brojem iteracija, bez obzira je li pronađeno bolje rješenje ili ne pa ga stoga svrstavamo u nezavršne algoritme. Glavni nedostatak lokalne pretrage je što može ostati u lokalnom optimumu te iz tog razloga nužno je osigurati mehanizam bijega iz lokalnog optimuma. U problemu CVRP-a lokalno rješenje se dobiva računajući razlike u ukupnoj pređenoj udaljenosti koje naprave vozila nakon zamjene pozicija korisnika ili ubacivanjem to jest izbacivanjem korisnika iz određenih ruta vozila. U nastavku će biti pojašnjeni neki od operatora za optimizaciju rješenja lokalne pretrage, [17].

3.2.1 Inter operatori

Inter operatori koriste se za premještanje korisnika koje ne dijele istu rutu. Kod inter operatora nužno je provjeravati ograničenje kapaciteta vozila te do zamjene korisnika dolazi samo u slučaju ako ne dolazi do prekoračenja ograničenja kapaciteta, [17]

Inter relocate

Inter relocate operator premješta korisnika iz svoje rute na novu poziciju u drugoj ruti. Grafički primjer rada operatora može se vidjet iz slike 3, [17].

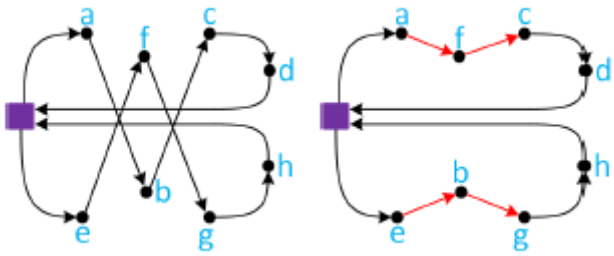


Slika 3. Grafički prikaz Inter relocate operatora

Izvor:[17]

Inter exchange

Inter exchange operator zamjenjuje dva korisnika u dvjema različitim rutama. Grafički prikaz rada operatora može se vidjeti iz slike 4, [17].

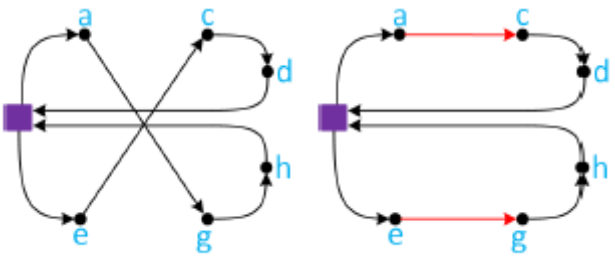


Slika 4. Grafički prikaz Inter exchange operatora

Izvor: [17]

Inter 2-Opt* operator

Inter 2-Opt* operator zamjenjuje djelomične krajeve rute između dviju ruta, uzimajući u obzir smjer ruta. Ovaj operator ima veliku primjenu kod problema sa vremenski prozorima pošto operator izbjegava izmjenu smjera rute. Grafički prikaz rada operatora je vidljiv iz slike 5, [17].



Slika 5. Inter 2-Opt* operator

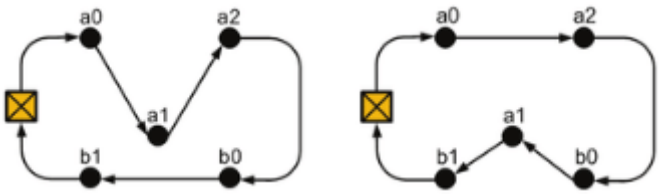
Izvor: [17]

3.2.2 Intra operatori

Za razliku od inter operatora ograničenje kapaciteta vozila kod intra operatora nije potrebno provjeravati jer se radi o rotaciji korisnika unutar iste rute, [16].

Intra relocate

Intra relocate operator zamjenjuje poziciju korisnika iz jedne pozicije u ruti na mjesto neke druge pozicije unutar iste rute. Grafički prikaz operatora je vidljiv iz slike 6., [17].

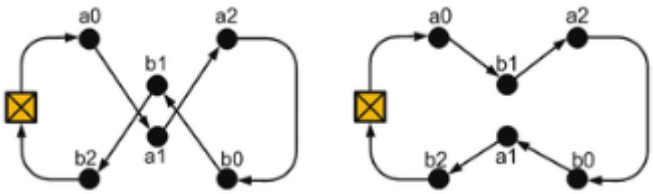


Slika 6. Intra relocate operator

Izvor:[18]

Intra exchange

Intra exhange operator zamjenjuje poziciju dvaju korisnika međusobno. Grafički prikaz rada operatora je vidljiv iz slike 7, [18].

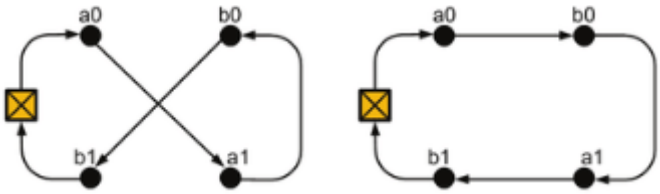


Slika 7. Intra exchange operator

Izvor: [18]

Intra 2-Opt

Intra 2-Opt operator uklanja križanje lukova unutar rute. Grafički prikaz rada operatora je vidljiv iz slike 8, [18].



Slika 8. Intra 2-Opt operator

Izvor: [18]

3.3 Simulirano kaljenje

Algoritam simuliranog kaljenja (eng. „Simulated anneling“, SA) je stohastički optimizacijski algoritam. Algoritam je nastao po analogiji sa metalurškim kaljenjem, kod kojeg je cilj oplemenjivanje metala tako da postane čvršći. Algoritam je modeliran po fizičkom procesu zagrijavanja materijala i zatim polaganog snižavanja temperature kako bi se smanjili nedostaci, čime se energija sustava smanjuje na minimum. U svakoj iteraciji algoritma simuliranog kaljenja, nova se točka nasumično generira. Udaljenost te nove točke od trenutne točke odnosno opseg pretraživanja područja optimuma temeljen je na distribuciji vjerojatnosti s ljestvicom proporcionalnom temperaturi. Algoritam prihvaća sve nove točke koje snižavaju cilj, ali i točke koje podižu cilj s određenom vjerojatnošću. Prihvaćanjem točaka koje podižu cilj, algoritam izbjegava mogućnost da ostane ograničen u lokalnom optimumu i omogućeno mu je da globalno istraži više mogućih rješenja. Raspored kaljenja se odabire kako bi se sistematski smanjila temperatura kroz njegovo trajanje. Kako se temperatura smanjuje, algoritam smanjuje opseg svoje pretrage kako bi konvergirao prema minimumu, [19].

3.4 Iterativno lokalno pretraživanje

Iterativno lokalno pretraživanje (eng. „Iterated local search“, ILS) se kroz niz iteracija koristi lokalnim pretraživanjem, ali početno rješenje lokalnog pretraživanja kreće na različitim mjestima. Kada lokalno pretraživanje ostane u lokalnom optimumu, primjenjuje se poremećaj kako bi se izbjegao lokalni optimum. Sama suština ILS algoritma je ta da se iterativno gradi niz rješenja generiranih ugrađenom heuristikom, što dovodi do daleko boljih rješenja nego da se koriste ponovljena nasumična ispitivanja te heuristike. Poremećaji se mogu dinamički skalirati kako bi se prevladali lokalni optimumi. Učinkovitost postupka uvelike ovisi o korištenom lokalnom pretraživanju kao i operatorima poremećaja. ILS algoritam je najčešće povezan s nekim drugim metaheurističkim algoritmom kao što je kolonija mrava, pretraživanje velikog susjedstva i dr., [12].

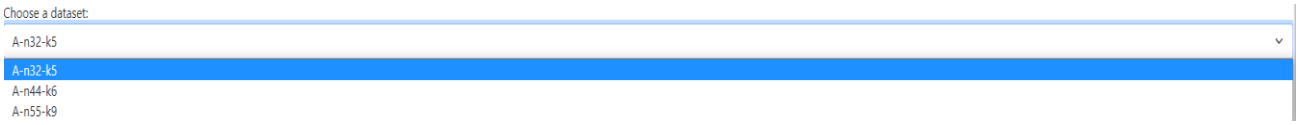
4. Izrada web aplikacije u programskom jeziku Python

U ovom poglavlju opisan je izgled web aplikacije kao i njezine funkcionalnosti, postupci učitavanja podataka, kreiranje početnog rješenja i lokalne pretrage. Za izradu web aplikacije korišten je programski jezik Python, programski jezik JavaScript i mikro web okvir Flask unutar programskog alata PyCharm.

4.1 Učitavanje podataka korisnika

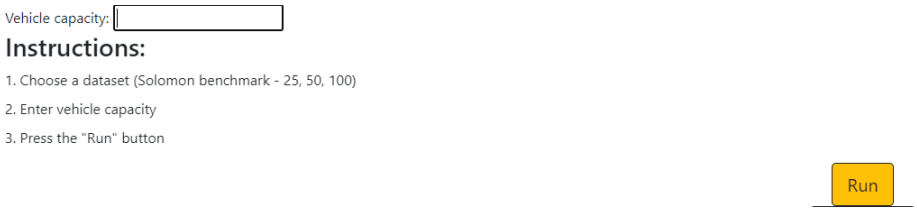
Na početnoj stranici potrebno je iz padajućeg izbornika odabrati željeni testni problem. Korišteni su testni problemi koje je razvio Augerat et al. za CVRP problem, [20].

Padajući izbornik se sastoji od 3 testna problema sa različitim brojevima korisnika kao što je vidljivo na slici 9.



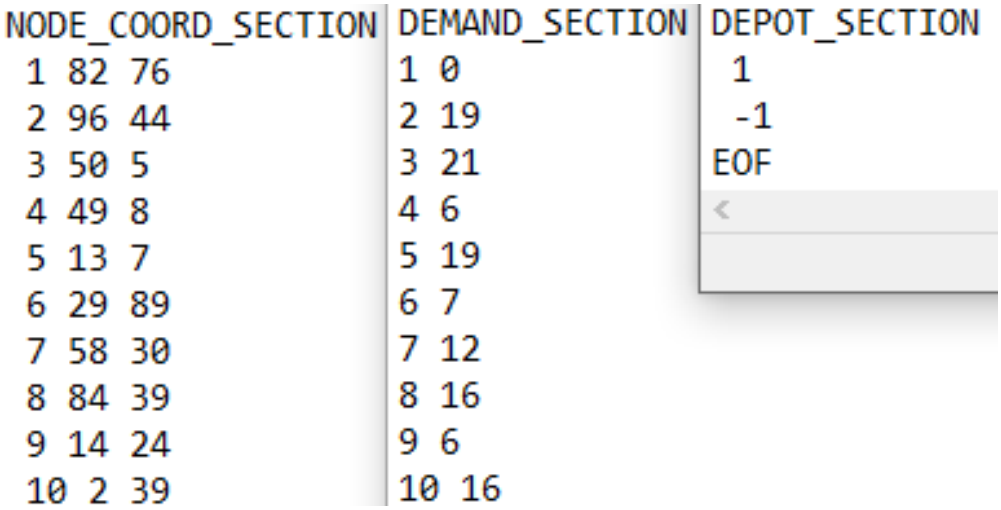
Slika 9. Padajući izbornik sa testnim problemima

Zatim je potrebno unijeti željeni iznos kapaciteta vozila koje će služit kao ograničenje u problemu unutar okvira za unos *Vehicle capacity* te pokrenuti rad aplikacije sa pritiskom na gumb *Run.* Stranica također sadrži i kratke instrukcije objašnjene u koracima od 1 do 3. Prikaz je vidljiv na slici 10.



Slika 10. Okvir za unos, instrukcije i gumb za pokretanje aplikacije

Testni podaci sastoje se od korisnika i skladišta opisanih atributima. Atributi su podijeljeni u sekcije NODE_COORD_SECTION koja sadrži redni broj korisnika i njegove koordinate, DEMAND_SECTION koji sadrži redni broj korisnika i njegovu potražnju te DEPOT_SECTION koji sadrži vrijednosti skladišta. Opis korisnika i njihovih atributa vidljivi su na slici 11.

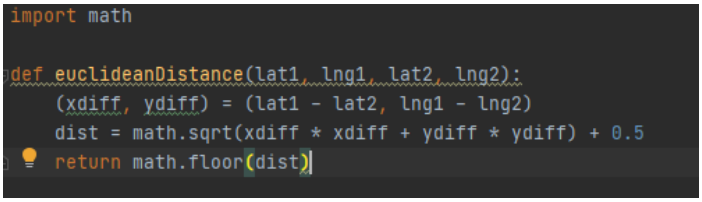


Slika 11. Prikaz sekcija testnih podataka za prvih 10 korisnika i skladišta

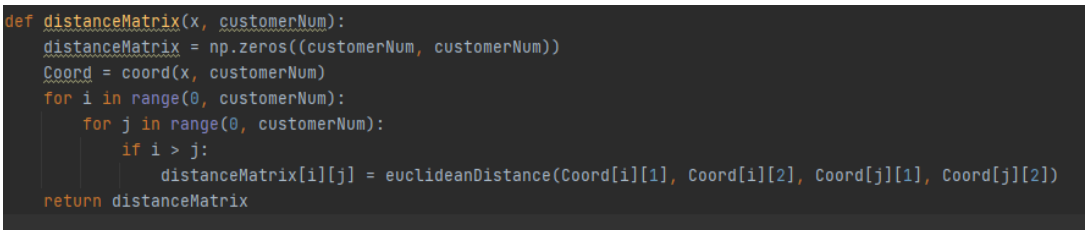
Svaka koordinata korisnika se sprema u listu naziva Coord koja se dalje prosljeđuje u distanceMatrix metodu kako bi napravili matricu udaljenosti za učitane korisnike. Također svaka potražnja se sprema u listu naziva costList koja se čita za svakog korisnika iz DEMAND_SECTION sekcije datoteke.

4.2 Konstrukcija početnog rješenja

Nakon učitavanja podataka korisnika računa se matrica udaljenosti svih korisnika. Svaki korisnik je opisan x i y koordinatom pa se pomoću metode euclideanDistance koja sadrži euklidsku formulu za izračun udaljenosti između dviju točaka u koordinatnom sustavu računa njihova međusobna udaljenost te se međusobna udaljenost unosi u varijablu distanceMatrix. Metoda euclideanDistance vraća međusobnu udaljenost u metrima. Euklidska formulacija koja se nalazi u metodi distance prikazana je u slici 12.



Slika 12. Metoda euclideanDistance s euklidskom formulacijom



Slika 13. Prikaz punjenja matrice distanceMatrix

Slika 13 prikazuje punjenje matrice distanceMatrix s udaljenostima. Prvo se kreira matrica koja je popunjena s nulama veličine ukupnog broja korisnika. Program pomoću *for* petlji prolazi kroz sve korisnike i popunjava matricu udaljenosti sa svim mogućim udaljenostima između korisnika.

Nakon stvaranja i popunjavanja matrice udaljenosti, kreira se varijabla početnog rješenja. Za inicijalizaciju početnog rješenja poziva se metoda koja sadrži pohlepni algoritam za davanje prvotnog rješenja. Metodi se prosljeđuju varijable capacity koja sadrži kapacitet vozila koji je zadan preko stranice, varijabla distanceMatrix koja sadrži matricu udaljenosti između svih korisnika, varijabla costList koja sadrži listu potražnji svakog korisnika te varijabla customerNum koja sadrži ukupan broj korisnika.

Skripta pohlepnog algoritma se sastoji od 3 metode: nextNode, createRoute i greedy. Algoritam kreće od greedy metode tako da u njoj se kreira prazna lista pod imenom routes koja će se punit rutama kojima će se vozilo prolaziti i praznu listu notVisited koja predstavlja listu svih neposjećenih korisnika. Algoritam će raditi dok god svi korisnici iz liste neposjećenih korisnika ne budu posjećeni to jest dok god je vrijednost u list notVisited veća od 0. Vozilo kreće s kreiranjem varijable route koja poziva metodu createRoute. U createRoute metodi postavlja se početno polazište s vrijednostima jednakima učitanim iz DEPOT_SECTION sekcije datoteke. Zatim kako bi se krenulo u posjetu korisnika poziva se metoda nextNode. Metoda nextNode prolazi kroz listu neposjećenih korisnika i traži najbližeg korisnika od trenutačne pozicije, a da pri tome ne prelazi preko ograničenja kapaciteta vozila. Nakon dolaska kod korisnika kapacitet vozila se povećava za njegovu potražnju. Posjećenog korisnika se miče iz liste neposjećenih korisnika te ga se sprema u listu posjećenih korisnika trenutnog vozila. Ukoliko je kapacitet vozila jednak nuli ili bi ostali korisnici prekoračili ograničenje vozila, vozilo se vraća u skladište i lista rute se dodaje u listu svih ruta. Zatim se ponovno kreira novo vozilo sa novom praznom listom ruta, kreće se iz skladišta i postupak se tako ponavlja dok svi korisnici iz liste neposjećenih korisnika ne budu posjećeni. Algoritam vraća listu ruta, broj potrebnih vozila te ukupnu prijeđenu udaljenost.

4.4 Postupak poboljšanja rješenja

Nakon što se dobije početno rješenje vrše se optimizacijski postupci za poboljšanje rezultata. Postavljeno je da se izvrši pretraživanje susjedstva u vremenskom periodu od 5 sekundi za svaki postupak.

4.4.1 Promjenjivo pretraživanje susjedstva

Promjenjivo pretraživanje susjedstva (eng. „Variable neighborhood search“, VNS) je metaheuristička metoda koju su Mladenović i Hansen predložili 1997. godine. Ona se koristi za rješavanja niza problema kombinatorne optimizacije i problema globalne optimizacije. Istražuje udaljena susjedstva postojećeg rješenja i odatle prelazi na novo ako i samo ako je postignuto poboljšanje, [21].

VNS je izgrađen na sljedećim pretpostavkama, [21]:

- Lokalni minimum s obzirom na jednu strukturu susjedstva nije nužno lokalni minimum za drugu strukturu susjedstva,
- Globalni minimum je lokalni minimum s obzirom na sve moguće strukture susjedstva,
- Za mnoge probleme, lokalni minimumi s obzirom na lokalna susjedstva relativno su blizu jedan drugome.

4.4.2 Promjenjivo spuštanje susjedstva

Promjenjivo spuštanje susjedstva (eng. „Variable neighborhood descent“, VND) je metoda za promjenu susjedstva na deterministički način. U opisu algoritma pretpostavljamo da je zadano početno rješenje *x*. Većina heuristika lokalnog pretraživanja u fazi spuštanja koristi vrlo mali broj susjedstva. Konačno rješenje treba biti lokalni minimum u odnosu na svih *kmax* struktura susjedstva, stoga su šanse za postizanje globalnog minimuma veće kada se koristi VND nego s jednom strukturom susjedstva, [21].

VNS skripta sadrži metodu VNS koja kada se pozove pokreće unutarnju štopericu i kopira rute početnog rješenja te kreira nove rute pozivom metode VND. VND metoda također kopira rute početnog rješenja te se izvršava dok god ima poboljšanja. Unutar VND metode koriste se tri operatora za optimizaciju ruta: twoOpt operator, inter relocate operator i inter swap operator. Rad ovih operatora je već objašnjen prije u radu. Ukoliko operator nađe poboljšanje, nove poboljšane rute se spremaju. Kada poboljšanja više nema VND metoda vraća novo nastale rute.

Zatim se VNS metoda izvršava dok god ne istekne vremenski period od 60 sekundi. Shake metoda koristi operatore inter swap i inter relocate za svako susjedstvo te radi slučajan odabir jednog od operatora za izmjenu ruta i vraća nazad novo nastale rute. Zatim se ponovno vrši optimizacija promjenjenih ruta pomoću VND metode. Dobiveno rješenje se uspoređuje sa prijašnjim rješenjem i rješenje koje je bolje postaje novo najbolje rješenje i njegove rute se spremaju. Postupak se ponavlja za svih 10 susjedstva ili broj susjedstva koji je bio moguć za proći u ograničenom vremenu. Metoda VNS vraća najbolju moguću rutu.

5. Rezultati

Za prikaz rada programa koristi se podatkovni set „A-n32-k5“ gdje kapacitet vozila je ograničen na 75. Usporedit će se rezultati istraživanja 5 susjedstva.

```
(First solution) Number of vehicles:  6
(First solution) Total distance:  1268.0
```

Slika 14. Početno rješenje

Slika 14 prikazuje početno rješenje koje se dobije primjenom pohlepnog algoritma. Broj potrebnih vozila je 6 te ukupna pređena udaljenost je 1268.

```
[ 0 30 16  1 12  0]  Distance:  73.0 , Demand:  72
[ 0 14 24 27  0]  Distance:  64.0 , Demand:  47
[ 0 26  7 13 17 31 21  0]  Distance:  156.0 , Demand:  74
[ 0 19  3 23 28  9 22  0]  Distance:  265.0 , Demand:  73
[ 0 18  8 11  4  2  6  0]  Distance:  227.0 , Demand:  73
[ 0 20  5 25 10 15 29  0]  Distance:  194.0 , Demand:  71
Neighborhood:  0 Total distance: 979.0 Number of vehicles: 6
```

Slika 15. Rješenje pretrage prvog susjedstva

Slika 15 prikazuje rute koje je vozilo prošlo te udaljenost svake rute kao i potražnju svake rute. Te na samom kraju ukupnu prijeđenu udaljenost i broj vozila potrebnih da bi se riješio problem.

$$1268 - 979 = 289 \tag{2}$$

Odmah nakon prve iteracije vidljiva je velika ušteda na ukupnoj prijeđenoj udaljenosti prikazana jednadžbom 1. Ukupna ušteda je 289.

```
[ 0 30 16  1 12  0]  Distance:  73.0 , Demand:  72
[ 0 24 27  0]  Distance:  59.0 , Demand:  44
[ 0 21 31 17 13  7 26  0]  Distance:  156.0 , Demand:  74
[ 0 18  8 28  4 11  9 22  0]  Distance:  238.0 , Demand:  75
[ 0 14  6 23  3  2 19  0]  Distance:  209.0 , Demand:  74
[ 0 20  5 25 10 15 29  0]  Distance:  194.0 , Demand:  71
Neighborhood:  1 Total distance: 929.0 Number of vehicles: 6
```

Slika 16. Rješenje pretrage drugog susjedstva

Slika 16 prikazuje pretragu drugog susjedstva iz koje je vidljivo da u usporedbi sa pretragom prvog susjedstva da je došlo do dodatnog poboljšanja krajnjeg rješenja. Iako se udaljenost 2. rute povećala, udaljenost 4. i 5. rute značajno se smanjila što na kraju rezultira u smanjenje ukupne udaljenosti za 50 koja je vidljiva iz jednadžbe 3. Broj potrebnih vozila ostaje isti.

$$979 - 929 = 50 \tag{3}$$

```
[ 0 30 16  1 12  0]  Distance:  73.0 , Demand:  72
[ 0 24 27  0]  Distance:  59.0 , Demand:  44
[ 0 26  7 13 17 31 21  0]  Distance:  156.0 , Demand:  74
[ 0 18 11  4 23  2  6  0]  Distance:  228.0 , Demand:  75
[ 0 14 22  9  8 28  3 19  0]  Distance:  267.0 , Demand:  74
[ 0 29 15 10 25  5 20  0]  Distance:  194.0 , Demand:  71
Neighborhood:  2 Total distance: 977.0 Number of vehicles: 6
```

Slika 17. Rješenje pretrage trećeg susjedstva

Slika 17 prikazuje pretragu trećeg susjedstva iz kojeg vidljivo da je prva ruta i druga ruta ostaju nepromijenjive u odnosu sa prvom i drugom rutom od pretrage drugog susjedstva , ali izmjene u ostatku ruta su rezultirale u lošijem krajnjem rješenju.

$$929 - 977 = -48$$

(4)

Iz jednadže 4 vidljivo je da rješenje druge pretrage ostaje kao najbolje rješenje. Broj potrebnih vozila ostaje isti.

```
[ 0 30 16  1 12  0]  Distance:  73.0 , Demand:  72
[ 0 24 27  0]  Distance:  59.0 , Demand:  44
[ 0 26  7 13 17 31 21  0]  Distance:  156.0 , Demand:  74
[ 0 14 22 18  8  4 28 23  3  6  0]  Distance:  245.0 , Demand:  74
[ 0 19  2 11  9  0]  Distance:  279.0 , Demand:  75
[ 0 20  5 25 10 15 29  0]  Distance:  194.0 , Demand:  71
Neighborhood:  3 Total distance: 1006.0 Number of vehicles: 6
```

Slika 18. Rješenje pretrage četvrtog susjedstva

Slika 18 prikazuje rješenje pretrage četvrtog susjedstva koje i dalje ne nalazi bolje rješenje od pretrage drugog susjedstva kao što je vidljivo iz prikaza 5. Štoviše rješenje pretrage četvrtog susjedstva je najgore do sad izuzetak prvog rješenja. Broj vozila ostaje isti.

$$929 - 1006 = -77$$

(5)

```
[ 0 30 16  1 12  0]  Distance:  73.0 , Demand:  72
[ 0 14 24 27  0]  Distance:  64.0 , Demand:  47
[ 0 26  7 13 17 31 21  0]  Distance:  156.0 , Demand:  74
[ 0 22 18  8 28 23  2  3  6  0]  Distance:  231.0 , Demand:  73
[ 0 19  4 11  9  0]  Distance:  280.0 , Demand:  73
[ 0 20  5 25 10 15 29  0]  Distance:  194.0 , Demand:  71
Neighborhood:  4 Total distance: 998.0 Number of vehicles: 6
```

Slika 19. Rješenje pretrage petog susjedstva

Slika 19 prikazuje rješenje pretrage petog susjedstva. Iako zadnja pretraga ne daje nalošije rješenje isto tako ne daje ni najbolje unatoč svim izmjenama u rutama. Broj potrebnih vozila ostaje isti.

$$929 - 998 = -69 \tag{6}$$

Nakon pretrage 5. susjedstva zaključujemo kako je druga pretraga pokazala najbolji rezultat te unatoč raznim izmjenama redoslijeda ruta te izradom novih ruta u drugim pretragama nije došlo do daljnjeg poboljšanja. Ovakvo rješenje je moguće i vjerojatno da se poveća broj pretrage susjedstva ili vremensko ograničenje rezultati bi bili vjerojatno bolji.

6. Zaključak

Završni rad prikazuje razvoj web aplikacije za rješavanje problema usmjeravanja vozila s ograničenim kapacitetom. Gledajući kako se CVRP smatra osnovnim VRP problemom, u radu su i prikazani ostale nadogradnje i varijante vezane za problem usmjeravanja vozila. Valja napomenuti kako rad nije kompletno funkcionalan jer fali vizualizacija rješenja.

Kroz poglavlja rada opisani su postupci i metode dobivanja početnog rješenja, načini njegovog poboljšanja te konačno najbolje moguće rješenje u zadanim ograničenjima. Za dobivanje početnog rješenja korišten je pohlepni algoritam te je to rješenje poboljšano metaheuristikom pretraživanja lokalnog susjedstva u nadogradnji sa varijantom lokalne pretrage padajućeg pretraživanja susjedstva koje koristi inter i intra operatore za poboljšanje, odnosno izmjene u rutama vozila. Navedena rješenja su prikazana na jednom od mogućih testnih problema s zadanim ograničenjima. Prikazane su izmjene koje su nastale prilikom istraživanja svakog susjedstva te poboljšanja ili pogoršanja u odnosu na početno rješenje i trenutačno najbolje rješenje. Pokazano je koliko zapravo puno se može uštediti s implementacijom optimizacijskih metoda te koliki bi zapravo veliki utjecaj one imale u stvarnom životu gdje su vozila ograničena s maksimalnim dosegom prije punjenja.

Završni rezultat rada prikazuje uštedu u ukupnoj prijeđenoj udaljenosti kako bi se poslužili svi korisnici koje se može pokazati vrlo vrijednim u današnjim prometnim problemima gdje je cijena goriva sve veća i veća te maksimalni domet električnih vozila isto tako sve veći. Razvojem novih tehnologija kao i sve veća želja za implementacijom električnih vozila u svakodnevni život stvaraju nove problematike kojima se moraju prilagoditi metode optimizacije ili razviti nove metode. Električna vozila su trenutačno teža od standardnih vozila s manjim kapacitetom prijevoza tereta zbog ograničenja infrastrukture čineći rješavanje CVRP problema jednim od prioriteta dok tehnologija električnih vozila ne napreduje. Primjenom rada u stvarnom svijetu može se postići ekonomska ušteda u prijevozu što je cilj svakog poslovanja koje se bavi prijevozom robe i dobara.

7. Literatura

[1] M. Abdul-Hak, N. Al-holou, Y. Bazzi, M. A. Tamer, Predictive Vehicle Route Optimization in Intelligent Transportation Systems, 2019, Preuzeto sa: https://www.sciencepublishinggroup.com/journal/paperinfo?journalid=390&doi=10.11648/j.ijdst.20190501.13 [Pristupljeno: rujan, 2022]

[2] G. Dantzig, J. Ramser, The Truck Dispatching Problem, 1959 Preuzeto sa: https://andresjaquep.files.wordpress.com/2008/10/2627477-clasico-dantzig.pdf [Pristupljeno: rujan, 2022]

[3] G. Clarke, J. W. Wright, Scheduling of Vehicles from a Central Depot to a Number of Delivery Points, 1964 Preuzeto sa: https://pubsonline.informs.org/doi/epdf/10.1287/opre.12.4.568 [Pristupljeno: rujan 2022]

[4] Wikipedia. Vehicle Routing Problem. Preuzeto sa: https://en.wikipedia.org/wiki/Vehicle_routing_problem [Pristupljeno: kolovoz, 2022]

[5] Wikipedia. Vehicle Routing Problem Figure. Preuzeto sa: https://en.wikipedia.org/wiki/File:Figure_illustrating_the_vehicle_routing_problem.png [Pristupljeno: kolovoz, 2022]

[6] Wikipedia. Travelling Salesman Problem. Preuzeto sa: https://en.wikipedia.org/wiki/Travelling_salesman_problem [Pristupljeno: kolovoz, 2022]

[7] Wikipedia. GLPK solution of traveling salesman problem. Preuzeto sa: https://en.wikipedia.org/wiki/File:GLPK_solution_of_a_travelling_salesman_problem.svg [Pristupljeno: kolovoz, 2022]

[8] Dongoo L., Jaemyung A., Vehicle Routing Problem with Vector Profits (VRPVP) with Max-Min Criterion, Studija, Korea Advanced Institute if Science and Technology (KAIST) Preuzeto sa: https://arxiv.org/ftp/arxiv/papers/1710/1710.10550.pdf [Pristupljeno: kolovoz, 2022]

[9] G. Desaulniers, J. Desrosiers, A. Erdmann, M.M.Solomon, F.Soumis, The Vehicle Routing Problem, VRP with Pickup and Delivery, Chapter 9, 2002 Preuzeto sa: https://www.researchgate.net/profile/Jacques-Desrosiers/publication/200622146_VRP_with_Pickup_and_Delivery/links/0deec528e7769dcf1d000000/VRP-with-Pickup-and-Delivery.pdf [Pristupljeno: kolovoz, 2022]

[10] Xiaobing G., Yan W., Shuhai L., Ben N., Vehicle Routing Problem with Time Windows and Simultaneous Delivery and Pick-Up Service Base on MCPSO, Znanstveni rad, Shenzhen University, 2012 Preuzeto sa: https://www.hindawi.com/journals/mpe/2012/104279/ [Pristupljeno: kolovoz, 2022]

[11] Carić T. Unaprijeđenje organizacije transporta primjenom heurističkih metoda. Disertacija. Sveučilište u Zagrebu, Fakultet prometnih znanosti; 2004

[12] D. Sariklis, S. Powell, The Journal of the Operational Research Society, Vol.51, No.5, A heuristic method for the open vehicle routing problem, 2000 Preuzeto sa: https://www.jstor.org/stable/254187 [Pristupljeno: kolovoz, 2022]

[13] C. Jacobs-Blecha, M. GoetSchalckx, The Vehicle Routing Problem with Backhauls: Properties and Solution Algorithms, Georgia Tech Resarch Corporation, 1992 Preuzeto sa: https://www2.isye.gatech.edu/~mgoetsch/cali/VEHICLE/VRPB/VRPB.HTM [Pristupljeno: kolovoz, 2022]

[14] F. Ribic, Izrada programskog sučelja za rješavanje problema vremenski ovisnog usmjeravanja vozila, Sveučilište u Zagrebu, Fakultet prometnih znanosti, Završni rad, 2020

[15] G. Radanović, Pregled heurističkih algoritama, Sveučilište Zagreb, Fakultet elektotehnike i računarstva, 2007, Preuzeto sa: http://www.zemris.fer.hr/~golub/ga/studenti/seminari/2007_radanovic/index.html [Pristupljeno: kolovoz, 2022]

[16] Programiz. Greedy Algorithm. Preuzeto sa: https://www.programiz.com/dsa/greedy-algorithm [Pristupljeno: kolovoz, 2022]

[17] T. Erdelić, Solving the electric vehicle routing problem using a hybrid adaptive large neighborhood search method, Doktorski rad, University of Zagreb, Faculty of Transport and Traffic Sciences, 2021

[18] S. Watanabe, K. Sakakibara, A Multiobjectivization Approach for Vehicle Routing Problems, Intra Route operators for CVRPTW, 2006 Preuzeto sa: https://www.researchgate.net/figure/Intra-Route-operators-for-CVRPTW_fig3_221228536 [Pristupljeno: kolovoz, 2022]

[19] Mathworks. What is Simulated Annealing? Preuzeto sa: https://www.mathworks.com/help/gads/what-is-simulated-annealing.html [Pristupljeno: kolovoz, 2022]

[20] CVRP Instances. Preuzeto sa: http://www.bernabe.dorronsoro.es/vrp/index.html?/Problem_Instances/CVRPTWInstances.html [Pristupljeno: rujan, 2022]

[21] Wikipedia. Variable neighborhood search. Preuzeto sa: https://en.wikipedia.org/wiki/Variable_neighborhood_search#cite_note-13 [Pristupljeno: rujan, 2022]

8. Popis slika

Sveučilište u Zagrebu
Fakultet prometnih znanosti
Vukelićeva 4, 10000 Zagreb

IZJAVA O AKADEMSKOJ ČESTITOSTI I SUGLASNOSTI

Izjavljujem i svojim potpisom potvrđujem da je završni rad (*vrsta rada*) isključivo rezultat mojega vlastitog rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu, a što pokazuju upotrijebljene bilješke i bibliografija. Izjavljujem da nijedan dio rada nije napisan na nedopušten način, odnosno da je prepisan iz necitiranog rada te da nijedan dio rada ne krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.

Svojim potpisom potvrđujem i dajem suglasnost za javnu objavu završnog/diplomskog rada pod naslovom Interaktivna web aplikacija za rješavanje problema usmjeravanja vozila, u Nacionalni repozitorij završnih i diplomskih radova ZIR.

Student/ica:

U Zagrebu, rujan,2022.

Lucian Vrbanc
(ime i prezime, *potpis*)