ALGORITAM OPTIMIZACIJE KOLONIJOM MRAVA ZA RJEŠAVANJE PROBLEMA USMJERAVANJA

Turjak, Tea

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni preddiplomski studij računarstva

ALGORITAM OPTIMIZACIJE KOLONIJOM MRAVA ZA RJEŠAVANJE PROBLEMA USMJERAVANJA VOZILA

Završni rad

Tea Turjak

Osijek, 2021.

SADRŽAJ

1. UVOD

Problem usmjeravanja vozila (engl. *vehicle routing problem*, VRP) predstavlja problem kombinatorne optimizacije u upravljanju distribucijom dobara između skladišta i konačnih klijenata. Glavni je cilj ovog problema pronaći najkraći put koji vozila trebaju prijeći kako bi obišla sve klijente, odrediti najmanji broj vozila potrebnih za dostavne usluge te odrediti koja vozila posjećuju pojedine klijente. Pri kapacitativnoj inačici problema usmjeravanja vozila, analiziranoj u ovome radu, zahtjevi klijenata su unaprijed poznati, a sva potrebna vozila imaju isti kapacitet i kreću iz istog skladišta.

U suvremenom svijetu, uz razvoj globalizacije, planiranje i organizacija transporta dobara i ljudi veliki je izazov. Najboljim mogućim rješenjem problema usmjeravanja vozila i primjenom u praksi smanjili bi se troškovi transporta, uštedjelo vrijeme i povećalo zadovoljstvo klijenata. Osim za distribuciju dobara, dobivena rješenja moguće je primijeniti i puno šire, primjerice za raspored vožnje školskih autobusa, dostavu pošte i hrane, prikupljanje kućnog otpada i slično.

Pri matematičkom modeliranju problema klijenti, skladište i ceste čine potpuno povezani graf. Distribucija robe organizira se između skladišta i klijenata, koji su predstavljeni čvorovima na grafu, a kretanje vozila odvija se cestama, bridovima grafa, koje povezuju klijente i skladište. Problem pripada kategoriji NP-teških problema vezanih uz kombinatornu optimizaciju jer u sebi sadrži problem trgovačkog putnika kao potproblem. Povećanje broja klijenata/čvorova grafa dovodi do povećanja kompleksnosti problema, a povećanjem kompleksnosti često se smanjuje mogućnost dobivanja optimalnih rješenja. Jedan od mogućih načina rješavanja optimizacijskih problema primjena je novih klasa algoritama koji su poznati pod nazivom metaheuristike. Uvođenjem metaheuristika povećava se mogućnost pronalaženja kvalitetnijih rješenja za opisane probleme unutar relativno kratkog vremena. Algoritam optimizacije kolonijom mrava (engl. *ant colony optimization*, ACO) popularna je metaheuristika usmjerena na diskretne probleme optimizacije koju je moguće primijeniti za rješavanje problema usmjeravanja vozila, a nadahnuta je ponašanjem mrava u potrazi za hranom. Prvobitna inačica algoritma optimizacije kolonijom mrava poznata je pod nazivom mravlji sustav (engl. *ant system*, AS), a proučavanjem i daljnjim istraživanjem razvijen je elitistički mravlji sustav (engl. *elitist ant system*, EAS). Elitistički mravlji sustav predstavlja poboljšani algoritam početnog mravljeg sustava te je korišten u ostvarenom programskom rješenju.

U drugom poglavlju detaljnije je opisan problem usmjeravanja vozila zajedno s objašnjenjem nekih inačica tog problema. Pojašnjen je algoritam optimizacije kolonijom mrava, a posebna

pažnja pridana je elitističkom mravljem sustavu za rješavanje kapacitativnog problema usmjeravanja vozila. Ostvareno programsko rješenje opisano je u trećem poglavlju. Prikazana je struktura programskog rješenja i način uporabe. Četvrto poglavlje obuhvaća eksperimentalnu analizu koja je provedena pomoću ostvarenog programskog rješenja. Opisane su korištene postavke u eksperimentalnom pokusu te su prikazani dobiveni rezultati i njihova analiza.

1.1. Zadatak završnog rada

Zadatak završnog rada je opisati problem usmjeravanja vozila gledajući na njega kao na problem kombinatorne optimizacije, opisati njegovu kapacitativnu inačicu i druge inačice te opisati algoritam optimizacije kolonijom mrava i njegovu primjenu na promatrani problem. U praktičnom dijelu rada potrebno je razviti programsko rješenje koje predstavlja elitistički mravlji sustav, inačicu algoritma optimizacije kolonijom mrava, za rješavanje kapacitativnog problema usmjeravanja vozila. Također, potrebno je provesti eksperimentalnu analizu na nekim primjerima problema iz literature.

2. PROBLEM USMJERAVANJA VOZILA I ALGORITAM OPTIMIZACIJE KOLONIJOM MRAVA

Problem usmjeravanja vozila pripada problemima kombinatorne optimizacije čija je svrha pronaći skupinu najkraćih mogućih ruta koju vozila trebaju prijeći kako bi se obišli svi klijenti zadanog grafa i zadovoljili njihovi zahtjevi. Najčešće se primjenjuje pri dostavljanju dobara ili u rješavanju problema transportnih sustava. Mogući pristupi rješavanja problema uključuju uporabu algoritama lokalne pretrage, heuristike, metaheuristike, te egzaktnih metoda korištenih za rješavanje jednostavnijih primjeraka problema [1].

Algoritam optimizacije kolonijom mrava pripada u skupinu metaheuristika koje se koriste za rješavanje problema usmjeravanja vozila. Neke od inačica algoritma optimizacije kolonijom mrava su mravlji sustav, elitistički mravlji sustav, *Ant-Q*, sustav kolonije mrava (engl. *ant colony system*, ACS), MAX-MIN mravlji sustav (engl. *max-min ant system*, MMAS) te mravlji sustav temeljen na rangiranju (engl. *rank-based ant system*) [2].

2.1. Problem usmjeravanja vozila

Problem usmjeravanja vozila može se opisati kao povezani graf G = (C, E) koji može, ali i ne mora, biti potpuno povezan. Klijenti i skladište su predstavljeni kao skup elemenata od N čvorova, C = $\{c_0, c_1, \ldots, c_N\}$, u kojemu čvor c_0 predstavlja skladište iz kojeg vozila počinju i završavaju svoje putovanje, odnosno rute. Svakom čvoru pridodana je vrijednost q_i koja simbolizira zahtjeve klijenta na tom određenom čvoru. Za čvor c_0 koji predstavlja skladište, vrijednost q_0 iznosi nula budući da nema nikakav utjecaj na promjenu kapaciteta vozila Q pri njihovom putovanju. Lukovi ovoga grafa pripadaju skupu E koji se sastoji od parova čvorova (c_i, c_j), pri čemu je $i \neq j$, a za svaki par čvorova izračunava se vrijednost d_{ij} koja označava udaljenost između promatranog para čvorova c_i i c_j. Cilj problema usmjeravanja vozila je obići sve čvorove zadanog grafa samo jednom, koristeći minimalan broj vozila koja imaju isti početni kapacitet Q na početku svojih ruta [3].

Probleme kombinatorne optimizacije karakterizira činjenica da povećanjem broja čvorova u grafu dolazi do eksponencijalnog porasta vremena koje je potrebno za pronalazak optimalnog rješenja što je obilježje NP-teških problema [4]. Budući da problem usmjeravanja vozila pripada skupini

problema kombinatorne optimizacije i u praksi je prilikom rješavanja ovog problema teško dobiti optimalna rješenja, također pripada skupini NP-teških problema. Iako je na velikom broju čvorova grafa mogućnost pronalaska optimalnog rješenja niska, rješenja dobivena uporabom ACO algoritama smatraju se dovoljno dobrima za daljnju uporabu i analizu.

Potencijalna primjena ovog problema vidljiva je u usmjeravanju školskih autobusa za transport učenika. Svaka škola na raspolaganju ima određen broj autobusa za prijevoz. Autobusi prikupljaju djecu s poznatih autobusnih stanica te ih voze do pripadne škole. Autobusi mogu primiti određen broj učenika i imaju zadanu rutu kojom prikupljaju učenike. Također je određen vremenski period u kojem se autobus zadržava na autobusnoj postaji i vrijeme potrebno da učenici uđu u autobus. Korištenje minimalnog broja autobusa potrebnih za prijevoz učenika, minimizacija vremena potrebnog za prikupljanje učenika i smanjivanje ukupnog vremena vožnje autobusa glavni su fokus ovoga problema [5]. Većina poslova u transportnoj industriji barem je djelomično povezana s problemom usmjeravanja vozila. S obzirom da se mnogi problemi iz stvarnoga života mogu predočiti kao problem usmjeravanja vozila, ovaj je problem iznimno relevantan u svakodnevici.

2.1.1. Neke varijante problema usmjeravanja vozila

Iako se mnogi problemi iz svakodnevnog života vezani uz transport mogu svesti na problem usmjeravanja vozila među njima postoje razlike koje, ako se uključe u model problema, mogu značajno pomoći pri dobivanju kvalitetnijih rješenja. Zbog toga je došlo do razvitka različitih varijanti problema usmjeravanja vozila. Neke od najpoznatijih varijanti problema usmjeravanja vozila su kapacitativni problem usmjeravanja vozila, problem usmjeravanja vozila s vremenskim prozorima, problem usmjeravanja vozila s povratnim prikupljanjem, problem usmjeravanja vozila s prikupljanjem i isporukom i problem usmjeravanja vozila s ograničenom udaljenošću.

Jedna od najpoznatijih i najosnovnijih varijanti problema usmjeravanja vozila, promatrana u ovome radu, je kapacitativni problem usmjeravanja vozila (engl. *capacitated vehicle routing problem*, CVRP). Temelji se na pretpostavkama da su zahtjevi kupaca unaprijed poznati i nepromjenjivi te da se ne mogu dijeliti na više vozila. Određen je minimalan broj vozila potreban za obilazak svih klijenata pri čemu svako vozilo ima isti kapacitet. Minimalan potreban broj vozila utvrđuje se rješavanjem problema pakiranja [6]. Taj broj se dobije tako da se zbroj zahtjeva svih klijenata podijeli s kapacitetom jednog vozila i zaokruži na prvi veći cijeli broj. Prije izračuna potrebne vrijednosti pazi se da nijedan zahtjev klijenta nije veći od vrijednosti kapaciteta jednog

vozila. Svako vozilo svoju rutu započinje i završava u skladištu. Glavna zadaća CVRP-a je pronaći skupinu najkraćih ruta koja vozila prijeđu tako da posjete sve klijente, a da pri tome ne prekorače unaprijed zadani kapacitet vozila [1].

Problem usmjeravanja vozila s vremenskim prozorima (engl. *VRP with Time Windows*, VRPTW) inačica je CVRP problema u kojem je svakom klijentu pridružen vremenski interval $[a_i, b_i]$, poznat pod nazivom vremenski prozor. Taj interval predstavlja vrijeme u kojem vozilo obavlja zadani zahtjev klijenta. Pretpostavke su da vozila počinju svoje putovanje iz skladišta u trenutku kada počinje teći vrijeme, da se zahtjev klijenta obavi u pripadnom vremenskom intervalu te da vozilo mora ostati kod klijenta određeno zadano vrijeme prije nego što nastavi daljnju isporuku [7].

Problem usmjeravanja vozila s povratnim prikupljanjem (engl. *VRP with Backhauls*, VRPB) predstavlja varijantu CVRP problema u kojem su klijenti podijeljeni u dvije liste. Prva lista obuhvaća klijente kojima se dostavljaju paketi, a druga lista sastoji se od klijenata od kojih vozila prikupljaju pakete. Uvjet je da se prvo dostave svi paketi, a zatim se vrši prikupljanje paketa klijenata s druge liste prije nego što se vozilo vrati u skladište [1].

Problem usmjeravanja vozila s prikupljanjem i isporukom (engl. *VRP with Pickup and Delivery*, VRPPD) označava varijantu VRP-a pri kojoj se zahtjevi klijenata sastoje od dva dijela. Prvi dio zahtjeva se odnosi na dostavu paketa klijentu koju vozilo obavi čim dođe do klijenta, a drugi dio zahtjeva su paketi koje vozilo preuzme od klijenta nakon izvršene isporuke [8].

Problem usmjeravanja vozila s ograničenom udaljenošću (engl. *distance-constrained vehicle routing problem*, DVRP) još je jedna od inačica VRP-a. Osim ograničenog kapaciteta vozila, ukupna prijeđena udaljenost svakog pojedinog vozila u dobivenom rješenju mora biti manja ili jednaka najvećoj mogućoj proputovanoj udaljenosti [9].

2.2. Algoritam optimizacije kolonijom mrava

Algoritmi optimizacije kolonijom mrava pripadaju skupini metaheuristika. Metaheuristika definira opći način uporabe heurističkih metoda koje su primjenjive za rješavanje različitih optimizacijskih problema pri čemu se dobivaju rješenja visoke kvalitete. ACO algoritmi koriste se za rješavanje dinamičkih i statičkih optimizacijskih problema. Inspiracija za ACO algoritam bio je način kojim

mrave pronalaze hranu u prirodi pomoću feromona. ACO algoritam se sastoji od kolonije umjetnih mrava u kojoj je suradnja među mravima ključna pri pronalaženju dobrih rješenja problema kombinatorne optimizacije. Problemi kombinatorne optimizacije mogu se svesti na probleme minimizacije ili maksimizacije [6].

Umjetni mravi predstavljaju stohastičku konstruktivnu metodu koja se koristi za izgradnju rješenja promatranog problema. Prije uporabe ACO algoritma za rješavanje problema, potrebno je podatke problema pretvoriti u model koji ACO algoritam može koristiti, odnosno model potpuno povezanog grafa kojim se umjetni mravi mogu kretati. Prilikom obilaska grafa, mravi ostavljaju feromone na lukovima grafa koji im služe u ponovnim obilascima pri traženju boljih rješenja. Također, svaki mrav ima mogućnost pamćenja puta kojim se kretao. Osim feromona, druga bitna informacija pri konstrukciji rješenja je heuristička vrijednost, odnosno heuristička informacija problema koja je poznata prije korištenja algoritma na zadani problem. Heuristička informacija predstavlja procjenu troška koja se postiže odabirom određenog luka grafa prilikom konstrukcije rješenja. Vrijednosti feromona i heurističke informacije koriste se za odabir sljedećeg koraka kretanja mrava na grafu. Mravi pojedinačno imaju malu mogućnost pronalaženja kvalitetnih rješenja, ali u suradnji s drugim mravima, odnosno korištenjem podataka o količini feromona koje su ostavili prethodni mravi, pridonosi se pronalaženju kvalitetnijih rješenja [6].

Glavne procedure ACO algoritma uključuju konstrukciju rješenja od strane mrava i ažuriranje količine feromona na lukovima grafova. Prije izvođenja ACO algoritma potrebno je postaviti broj iteracija algoritma kako bi mravi dobili mogućnost pronalaska rješenja što bolje kvalitete prilikom njegovog izvođenja. Način općeg rada ACO algoritma, prema [6], prikazan je na slici 2.1.

Linija ***Kod (ACOAlgoritam)***

```
1:      početak
2:        ulaz inicijalizacija_problema, postavljanje_parametara
3:        dok je broj_iteracija <= ukupan_broj_iteracija_algoritma činiti
4:            KonstrukcijaRješenjaOdStraneMrava
5:            AžuriranjeKoličineFeromona
6:        kraj petlje
7:      kraj
```

Sl. 2.1. Prikaz općeg rada ACO algoritma

Svaki mrav se postavlja na početni čvor iz kojeg počinje konstruirati rutu prema ciljanom čvoru. Parametri koji su nužni za određivanje vjerojatnosti odabira sljedećeg čvora, intenzitet isparavanja feromona te parametri α i β detaljnije su opisani u nastavku. Mrav provodi radnje za odabiranje sljedećeg čvora dok ne dođe do čvora koji predstavlja kraj njegovog putovanja. Budući da mrav nema podatke o tome približava li se ciljanom čvoru ili ne, ponavlja proceduru za odabir sljedećeg čvora. Može se dogoditi da mrav posjeti već posjećene čvorove i time stvori petlju. Nakon što mrav dođe do krajnjeg čvora, provjerava se putanja kojom je mrav došao do rješenja. U slučaju pronalaska petlji u obilasku, one se uklanjaju iz dobivenog rješenja i tek tada počinje izračunavanje kvalitete konstruiranog rješenja. Završetkom procedure konstrukcije rješenja od strane mrava odvija se procedura za ažuriranje količine feromona u grafu. Primjer izgradnje rješenja od strane mrava, prema [2], prikazan je na slici 2.2.

Linija ***Kod (KonstrukcijaRješenjaOdStraneMrava)***

```
1:      početak
2:        za i=1 do broj_mrava činiti
3:          PostavljanjeMravaNaPočetniČvor
4:          dok mravNijeDošaoDoCiljanogČvora činiti
5:            OdabirSljedećegČvora
6:          kraj petlje
7:          BrisanjePetljiIzKonstruiranogRješenja
8:          IzračunKvaliteteRješenja
9:        kraj petlje
10:     kraj
```

Sl. 2.2. Prikaz konstrukcije rješenja od strane mrava u ACO algoritmu

Isparavanje feromona odvija se po svim lukovima grafa za zadani intenzitet isparavanja. Nakon toga mravi ostavljaju određenu količinu feromona na lukovima grafa kojima su prošli tijekom putovanja od početnog do završnog čvora. Primjer isparavanja feromona i ostavljanje feromona od strane mrava, prema [6], prikazan je na slici 2.3.

Linija ***Kod (AžuriranjeKoličineFeromona)***

```
1:      početak
2:        IsparavanjeFeromona
3:        PromjenaKoličineFeromona
4:      kraj
```

Sl. 2.3. Prikaz ažuriranja količine feromona u ACO algoritmu

Jedan od prvih problema na kojima su vrednovani ACO algoritmi je problem trgovačkog putnika (engl. *traveling salesman problem*, TSP) koji se može predstaviti pomoću potpuno povezanog grafa. Cilj TSP-a je pronaći najkraći put za obilazak zadanih lokacija te sadrži početnu i krajnju lokaciju putovanja. TSP se smatra jednim od najpogodnijih problema za testiranje novih algoritama osmišljenih u svrhu rješavanja kombinatornih optimizacijskih problema [6].

Mravlji sustav predstavlja prvu inačicu ACO algoritma. Princip rada AS algoritma prilikom rješavanja TSP-a može se podijeliti na dva dijela: postupak kojim mravi konstruiraju svoje rute i ažuriranje količine feromona na prijeđenim lukovima kojima mravi prolaze [6].

Feromonski trag se sastoji od feromona koje mravi ostavljaju na svojemu putu u potrazi za hranom. Mravi ostavljaju feromone po putovima kojima su prošli nakon što konstruiraju svoju rutu. Heuristička informacija η_{ij} ovog algoritma jednaka je obrnuto proporcionalnoj vrijednosti međusobne udaljenosti d_{ij} između pojedina dva čvora (c_i, c_j) [10]

$$\eta_{ij} = \frac{1}{d_{ij}}. \tag{2.1.}$$

Međusobna udaljenost između svih čvorova grafa računa se prema euklidskoj udaljenosti

$$d_{ij} = d(T_i, T_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \tag{2.2.}$$

gdje je $d(T_i, T_j)$ udaljenost između dvije točke u ravnini (T_i i T_j), a svaka točka predstavljena je koordinatama x_i i y_i.

Kako bi bilo moguće započeti izvođenje algoritma, prvobitno se moraju postaviti parametri potrebni za izvođenje AS algoritma te početne vrijednosti matrica feromona, heurističke vrijednosti i vjerojatnosti odabira sljedećeg klijenta. Početno stanje feromona za sve članove matrice, prema [6], definirano je kao:

$$\tau_0 = m/C^{nn} \tag{2.3.}$$

gdje je τ_0 početna količina feromona koji se nalazi na svim stazama promatranog grafa, parametar m broj umjetnih mrava u koloniji, a C^{nn} kvaliteta rješenja koja je potrebna za izračun početnog stanja matrice feromona. Vrijednost C^{nn} određuje se izračunavanjem kvalitete rute koja započinje

iz početnog čvora, a zatim se uvijek odabire čvor s najbližom udaljenošću dok se ne obiđu svi čvorovi grafa (algoritam najbližeg susjeda). Izračunom kvalitete C^{nn} rute omogućeno je stvaranje matrice početnih vrijednosti feromona i matrice vjerojatnosti odabira klijenata.

Prema [6], matrica vjerojatnosti odabira sljedećeg čvora izračunava se prema izrazu:

$$p_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{j \in N_i^k}[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}, \tag{2.4.}$$

gdje je p_{ij}^{k} vjerojatnost da mrav k koji se nalazi kod čvora c_i odabere čvor c_j, τ_{ij} količina feromona koja se nalazi na putu između čvorova c_i i c_j, η_{ij} heuristička vrijednost između čvorova c_i i c_j, α parametar koji označava koliko koncentracija feromona utječe na odabir čvora, β parametar koji označava koliko heuristička vrijednost utječe na odabir čvora, a N_i^k predstavlja listu čvorova koji još nisu posjećeni.

Feromonski trag i heuristička informacija pridodaju se lukovima grafa jer prilikom odabira novog čvora kojeg će mrav obići, prvo se odabire put kretanja mrava. Mrav se prvobitno postavlja nasumično na početni čvor, a nakon toga izvodi se procedura za odabir sljedećeg čvora. Od izračunatih vjerojatnosti odabira sljedećeg čvora slučajnim odabirom se izabere čvor kojeg će mrav posjetiti. Prema [6], preporučene vrijednosti parametara pri uporabi AS-a za rješavanje TSP-a su da je vrijednost parametra α jednaka 1, vrijednost parametra β od 2 do 5, vrijednost parametra ρ jednaka 0.5, dok je vrijednost parametra m jednaka vrijednosti broja čvorova grafa N. Nakon što svi mravi kreiraju svoje rute, dolazi do mijenjanja količine feromona na lukovima grafa.

Prije nego što dođe do postavljanja feromona na lukove kojima su prošli mravi, prvo se odvija isparavanje feromona pri kojemu parametar ρ označava intenzitet isparavanja feromona. Isparavanje je predstavljeno kao:

$$\tau_{ij} = (1 - \rho)\tau_{ij}, \forall (i, j) \in E, \tag{2.5.}$$

gdje je E skup svih lukova potpuno povezanog grafa G = (C, E) [6].

Nakon isparavanja feromona može se krenuti na ažuriranje količine feromona na stazama. Prema AS-u, ažuriranje feromona provodi se prema izrazu:

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}, \tag{2.6.}$$

$$\Delta\tau_{ij}^{k} = \frac{1}{C^{k}}, \forall(i,j) \in T^{k}\ , \tag{2.7.}$$

gdje je $\Delta\tau_{ij}^{k}$ količina feromona koju ostavljaju mravi na prijeđenim lukovima rute T^{k} u trenutnoj iteraciji algoritma, C^{k} kvaliteta rješenja rute T^{k} [6].

Elitistički mravlji sustav predstavlja unaprjeđenje AS algoritma. Specifičnost EAS algoritma je da uz prisustvo mrava koji ostavljaju normalnu količinu feromona na prijeđenim rutama, kao u AS inačici, postoji elitini mrav koji u svakoj iteraciji postavlja određenu i veću količinu feromona na najbolju pronađenu rutu tijekom iteracije algoritma. Prema [6], preporučene postavke parametara za izvođenje EAS algoritma iste su kao i za AS inačicu. Razlika je u početnoj količini feromona koja se postavlja kao vrijednost matrice feromona prema izrazu:

$$\tau_{0} = (e + m)/C^{nn} \tag{2.8.}$$

gdje je preporučena vrijednost parametra e prema [6] za EAS algoritam jednaka broju čvorova grafa N.

Ažuriranje količine feromona na lukovima grafa prema EAS odvija se kao i za AS. Prvo se odvija isparavanje feromona prema (2.5.), dok se količina feromona koji mravi ostavljaju odvija prema:

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}{}_{ij}^{k} + e\Delta\tau_{ij}^{bs}, \tag{2.9.}$$

$$\Delta\tau_{ij}^{bs} = \frac{1}{C^{bs}}, \forall(i,j) \in T^{bs}, \tag{2.10.}$$

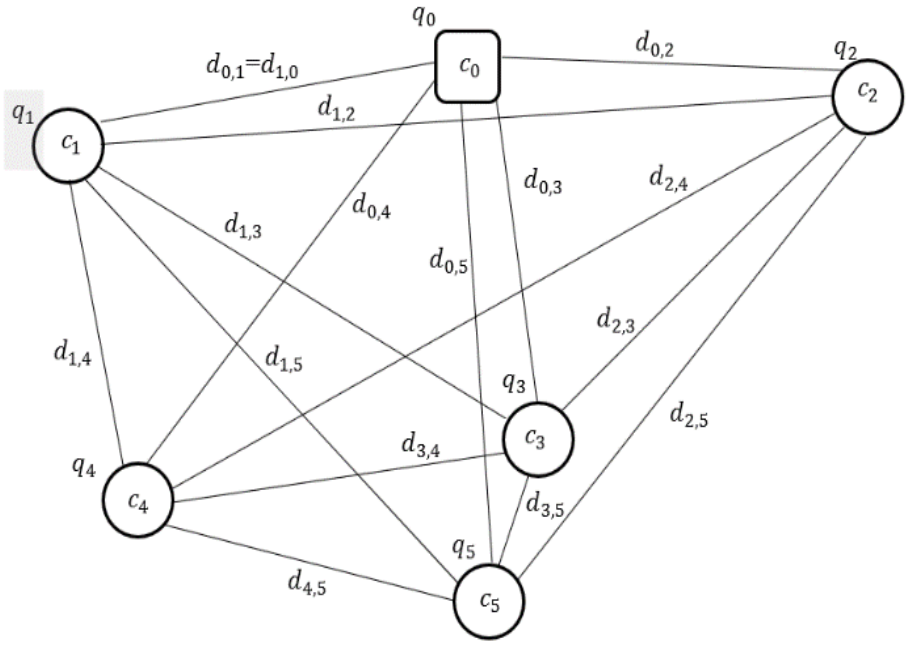
gdje je $\Delta\tau_{ij}^{bs}$ količina feromona koju ostavlja elitni mrav na prijeđenim lukovima rute T^{bs} s najboljom kvalitetom koja je pronađena u iteraciji algoritma, e težina feromona koju elitni mrav ostavlja na najboljoj ruti čija je preporučena vrijednost jednaka broju čvorova promatranog problema N, a C^{bs} je kvaliteta rješenja T^{bs} rute. Istraživanjima se pokazalo da usporedbom AS i EAS algoritma, EAS algoritam daje rješenja bolje kvalitete od AS algoritma [6].

2.2.1. Primjena elitističkog mravljeg sustava za rješavanje kapacitativnog problema usmjeravanja vozila

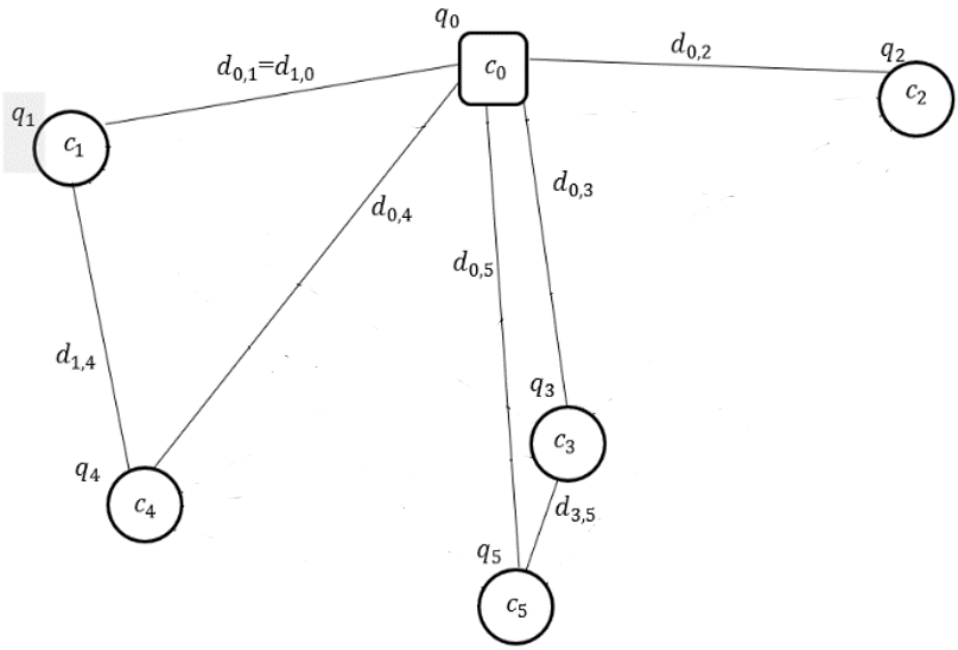
Uporaba ACO algoritama pokazala se vrlo učinkovita prilikom rješavanja problema usmjeravanja vozila prema [10] zbog čega se u ovome radu promatra utjecaj EAS algoritma kao pristupa rješavanja CVRP-a.

Kako bi se EAS algoritam mogao primijeniti na zadani problem, broj mrava korištenih u algoritmu postavlja se na minimalan potreban broj vozila. Svi mravi koriste istu matricu feromona prilikom odabira sljedećeg čvora te se na istoj obavlja proces ažuriranja feromona nakon konstrukcije svih ruta u iteraciji algoritma. Svaki mrav se ponaša kao da rješava TSP unutar svoje rute. Za razliku od TSP-a, umjesto da traži krajnji čvor u ruti i obilazi sve raspoložive čvorove, mrav svoj obilazak čvorova završava kada zahtjev klijenata na budećem mogućem čvoru premaši zadani dozvoljeni kapacitet kojima vozila raspolažu. Kada mrav više ne može podržati zahtjeve klijenata, vraća se u skladište i završava svoj obilazak. Time drugi mrav započinje svoj obilazak. Mravi koji konstruiraju rute kojima bi vozila prolazila ostavljaju količinu feromona koju bi normalni mrav ostavio na prijeđenim lukovima grafa. Količina feromona elitnog mrava dodaje se na najbolju dotad napravljenu rutu u svim iteracijama algoritma, odnosno rutu s najmanjom udaljenošću koju su mravi napravili.

Prikaz primjera potpuno povezanog grafa koji predstavlja mogući problem usmjeravanja vozila, prema opisu problema u poglavlju 2.1, prije nego što je na njemu primijenjen odabrani postupak rješavanja vidi se na slici 2.4, a primjer dobivene moguće rute nakon primjene algoritma na zadani problem prikazan je na slici 2.5.



Sl. 2.4. Prikaz mogućeg problema usmjeravanja vozila



Sl. 2.5. Prikaz mogućeg rješenja nakon izvršenja algoritma

3. OSTVARENO PROGRAMSKO RJEŠENJE

Programsko rješenje ostvareno je u objektno orijentiranom programskom jeziku C# u razvojnom okruženju Microsoft Visual Studio 2019. Microsoft Visual Studio 2019 integrirano je razvojno okruženje kreirano za izradu računalnih i mobilnih aplikacija, kreiranje internetskih stranica i drugih usluga koje su vezane uz razvoj softvera. Console App (.NET Core) predložak korišten je u svrhu izrade ovog programskog rješenja.

Ostvareno programsko rješenje omogućuje učitavanje podataka potrebnih za inicijalizaciju problema iz tekstualne datoteke, postavljanje parametara algoritma i broja ponavljanja izvršenja programa koji se upisuju preko dostupne konzole programa. Rezultati izvođenja algoritma zapisuju se u zadanu tekstualnu datoteku.

3.1. Način rada programskog rješenja

Nakon pokretanja programa, korisnik odabire ime tekstualne datoteke koja sadrži informacije o promatranom CVRP problemu. Zatim unosi vrijednosti parametara α, β i ρ, potrebnih za provedbu EAS algoritma, te određuje broj iteracija algoritma, broj ponavljanja izvođenja algoritma i ime datoteke u koju se zapisuju rezultati dobiveni tijekom rada programa.

Nakon unosa početnih podataka učitavaju se podatci iz datoteke potrebni za formuliranje problema koji obuhvaćaju ukupan broj čvorova, zbroj svih zahtjeva klijenata, poznato optimalno rješenje problema i zadani kapacitet vozila. Minimalan potreban broj vozila za zadani problem izračunava se rješavanjem problema pakiranja. Stvaraju se objekti koji predstavljaju klijente i skladište s pripadnim atributima i definiraju zadani problem. Nakon definiranja osnovnih postavki problema, kreiraju se matrice udaljenosti i heurističkih vrijednosti.

Matrica udaljenosti koristi se prilikom izračuna vrijednosti kvalitete C^{nn} rute. Primjenjuje se za odabir najbližeg klijenta u odnosu na klijenta kod kojeg se vozilo trenutno nalazi. Međusobna udaljenost između svih čvorova koji su predstavljeni objektima računa se prema euklidskoj udaljenosti točaka.

Izračunom kvalitete C^{nn} rute omogućeno je kreiranje matrice početnih vrijednosti feromona i matrice vjerojatnosti odabira klijenata. Stvaranjem ovih podataka počinje izvršavanje EAS algoritma za zadani broj iteracija.

Kreira se lista lokacija posjećenih klijenata koja se koristi za praćenje lokacija kojima vozila prolaze te lista neposjećenih klijenata. Vozilo započinje svoj obilazak i odvija se odabir klijenta kojeg će vozilo posjetiti. Nakon što su izračunate sve vjerojatnosti odabira neposjećenih klijenata prema izrazu (2.4.), redom se izračuna zbroj svih dobivenih vjerojatnosti prema:

$$sum = \sum p_{ij}^{k} \tag{3.1.}$$

Time se dobiva interval $[0, sum]$. Iz intervala se slučajnim odabirom generira broj r prema kojemu se, s obzirom na poredak vjerojatnosti u zbrajanju, odabire onaj klijent kojemu se r broj nađe u pripadnom podintervalu. Primjerice za:

$$0 - p_{i2}^{k} - (p_{i2}^{k} + p_{i3}^{k}) - \cdots - sum, \tag{3.2.}$$

ako je

$$r \in < p_{i2}^{k}, (p_{i2}^{k} + p_{i3}^{k})], \tag{3.3.}$$

pripadanje broja r navedenom podintervalu označava odabir klijenta koji se nalazi na lokaciji s indeksom 3.

Provjerava se može li vozilo podnijeti zahtjev klijenta, a ako može, indeks klijenta briše se iz liste neposjećenih klijenata te se zapisuje u listu posjećenih klijenata. Taj se proces ponavlja sve dok vozilo više ne može podnijeti zahtjeve klijenata. U slučaju da vozilo više nije u mogućnosti posjećivati klijente, završava svoju rutu i vraća se u skladište, a sljedeće vozilo počinje svoj obilazak. Kada su iskorištena sva vozila ili ako su posjećeni svi klijenti, izračunava se kvaliteta dobivene rute. U prvoj iteraciji algoritma, prva stvorena ruta postaje najbolje rješenje, a u idućim iteracijama, uspoređuju se dobivene kvalitete ruta i prema njima se odabire ruta najbolje kvalitete, u ovom slučaju ruta s kvalitetom najmanje vrijednosti.

Poslije usporedbe kvalitete rute, slijedi promjena količine feromona na stazama grafa. Prvo dolazi do isparavanja feromona na svim lukovima prema izrazu (2.5.). Dodavanje određene količine feromona odvija se prema izrazu (2.9.). Kada algoritam dosegne maksimalan broj iteracija, kreira se tekstualna datoteka u koju se zapisuje kvaliteta najbolje napravljene rute tijekom svakog izvršenja programa.

Nakon što se program izvrši zadani broj puta, dohvaćaju se podatci spremljeni u datoteci u kojoj su se zapisivale najbolje kvalitete ruta dobivene tijekom svakog izvođenja programa i nad njima se vrši deskriptivna statistika. Za potrebe ovog rada programsko rješenje sadrži kod koji omogućuje provođenje deskriptivne statistike nad dobivenim podatcima. Izračunava se prosječna kvaliteta dobivenih ruta, standardna devijacija, pronalazi se najbolja i najlošija kvaliteta rute te se računa odstupanje prosječne kvalitete ruta od dokazanog optimuma za promatrani problem. Prosječna kvaliteta dobivenih ruta računa se prema:

$$S_{avg} = \frac{\sum_{i=1}^{p} s_i}{p}, \tag{3.4.}$$

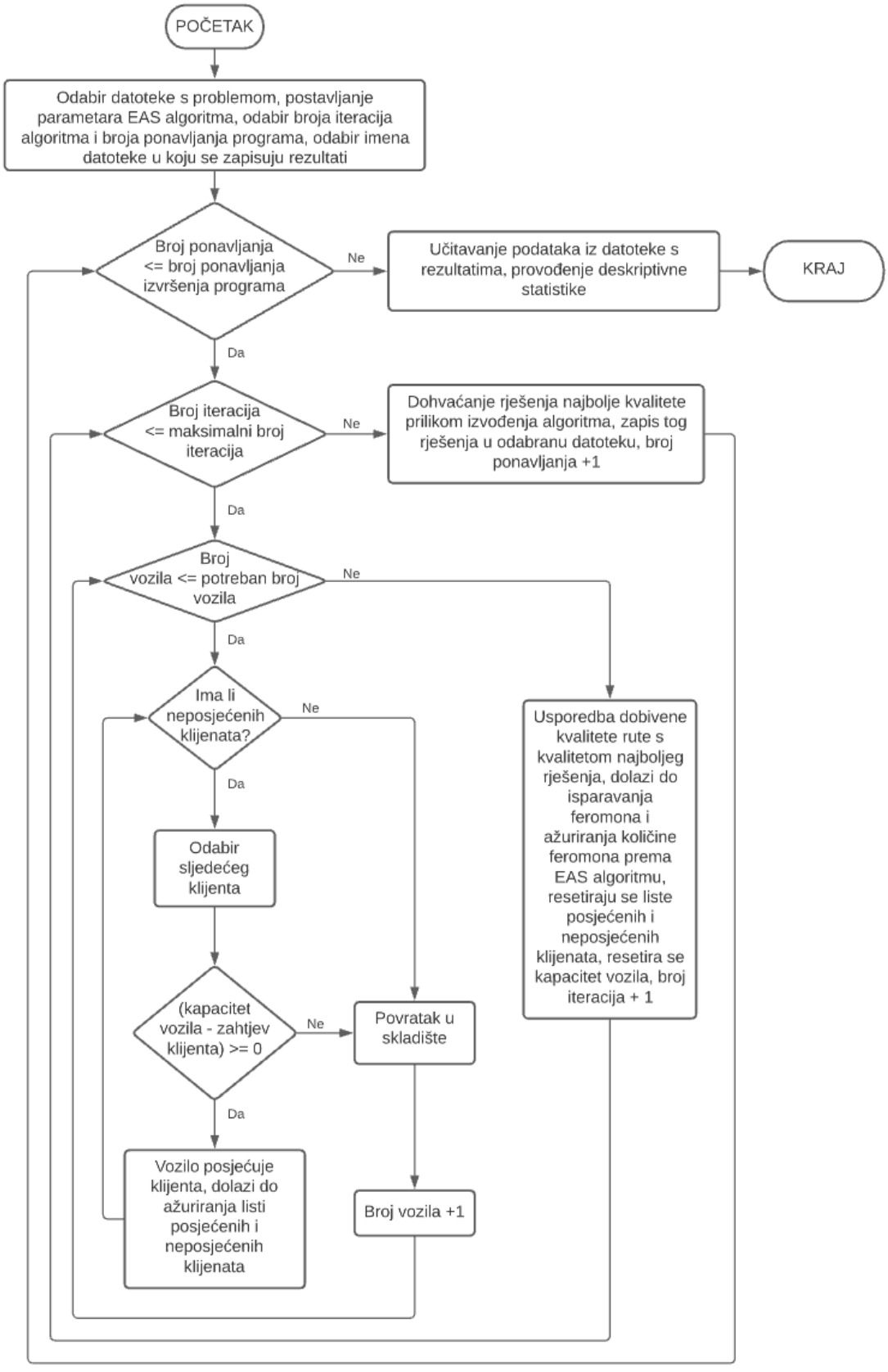
gdje je S_{avg} prosječna kvaliteta dobivenih ruta tijekom izvođenja pokusa, a s_i kvaliteta rute dobivena u jednom izvođenju programa koji se izvodi p puta. Standardna devijacija računa se kao:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{p}(s_i - S_{avg})^2}{p}}, \tag{3.5.}$$

gdje je σ standardna devijacija. Odstupanje prosječne vrijednosti kvalitete ruta od poznatog optimalnog rješenja promatrane instance računa se prema:

$$\Delta = \left|\frac{UB - S_{avg}}{UB}\right| \cdot 100\,, \tag{3.6.}$$

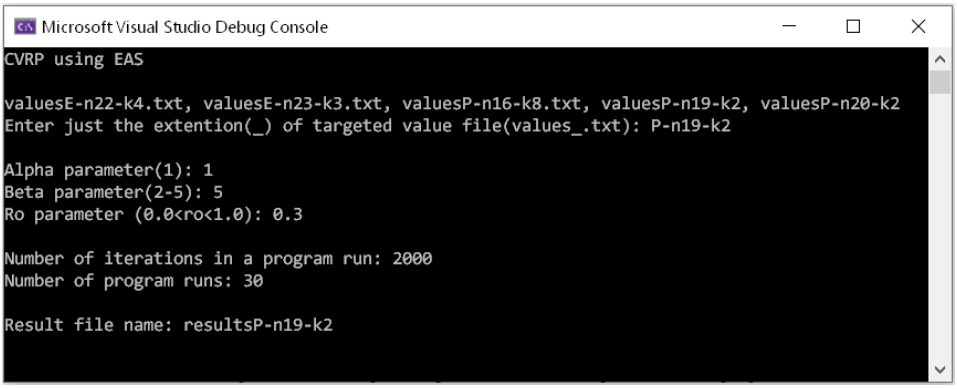
gdje Δ označava odstupanje S_{avg} od optimalnog rješenja instance UB u postotcima. Dijagram toka programskog rješenja prikazan je na slici 3.1.



Sl. 3.1. Dijagram toka programskog rješenja

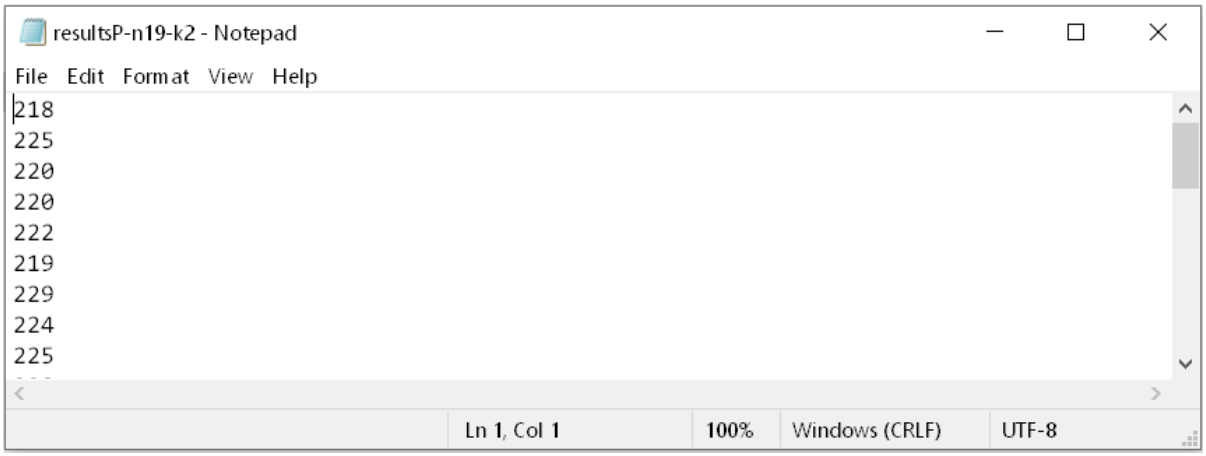
3.2. Prikaz i način uporabe programskog rješenja

Pokretanjem programskog rješenja, omogućena je interakcija s konzolom koja se koristi za unošenje podataka potrebnih za rad programskog rješenja. Korisnik odabire željenu datoteku s podatcima o problemu, parametre za EAS algoritam, broj iteracija algoritma, broj izvođenja algoritma i ime datoteke za spremanje podataka. Primjer upisa podataka prikazan je na slici 3.2.



Sl. 3.2. Prikaz upisa podataka na konzoli

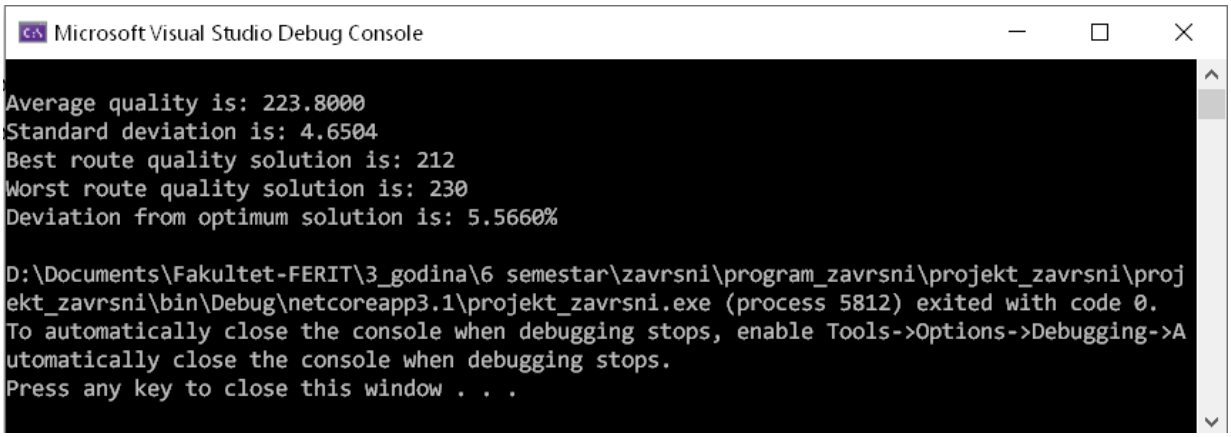
Kada program završi s radom rezultati EAS algoritma zapisani su u tekstualnoj datoteci pod odabranim imenom. Primjer rezultata koji su zapisani u tekstualnoj datoteci vidljiv je na slici 3.3.



Sl. 3.3. Prikaz zapisa rezultata nakon izvršenja programa u tekstualnoj datoteci

Za potrebe eksperimentalne analize na konzoli se ispisuje prosječna kvaliteta ruta dobivenih EAS algoritmom, standardna devijacija, najbolja i najlošija postignuta kvaliteta rute te odstupanje

prosječne kvalitete ruta od poznatog optimuma za promatrani problem. Primjer ispisa podataka potrebnih za eksperimentalnu analizu vidljiv je na slici 3.4.



Sl. 3.4. Prikaz ispisa podataka za eksperimentalnu analizu na konzoli

4. EKSPERIMENTALNA ANALIZA

Eksperimentalnom analizom nastoji se ispitati učinkovitost primjene EAS algoritma za rješavanje problema usmjeravanja vozila. U svrhu eksperimentalne analize, koristi se pet različitih primjera CVRP-a čiji su podatci dostupni na internetskoj stranici [11]. Osnovni podatci promatranih instanci uključuju broj klijenata N, broj potrebnih vozila K, kapacitet pojedinog vozila Q i poznato optimalno rješenje instance UB. Podatci koji opisuju pojedinu instancu prikazani su tablicom 4.1.

Tablica 4.1. Podatci instanci CVRP-a korištenih za eksperimentalnu analizu

Problem	**1**	**2**	**3**	**4**	**5**
Instanca	E-n22-k4	E-n23-k3	P-n16-k8	P-n19-k2	P-n20-k2
N	21	22	15	18	19
K	4	3	8	2	2
Q	6000	4500	35	160	160
UB	375	569	450	212	216

4.1. Postavke eksperimenta

Pri proučavanju ponašanja EAS algoritma, navedene CVRP instance testirane su s nekoliko različitih kombinacija zadanih parametara EAS algoritma. Parametri algoritma koji su fiksni tijekom izvođenja eksperimentalne analize su parametar α i broj iteracija algoritma tijekom jednog izvođenja programa. Također je nepromjenjiv broj ponavljanja izvršenja algoritma. Vrijednosti navedenih parametara prikazane su u tablici 4.2.

Tablica 4.2. Fiksne vrijednosti parametara koje se zadaju prije izvođenja programa

Parametar	**α**	**Broj iteracija algoritma**	**Broj ponavljanja izvršenja algoritma**
Vrijednost	1	2000	30

Prema [6], parametar α jednak je 1, a vrijednosti ostalih fiksnih parametara postavljene su kako bi se skupilo dovoljno kvalitetnih podataka za eksperimentalnu analizu. Vrijednost e zadana je prema broju klijenata pojedine instance te se ne mijenja tijekom izvođenja eksperimenta. Parametri β i ρ nisu fiksni i mijenjani su tijekom izvođenja pokusa. Parametrima β i ρ odabrane su po tri različite vrijednosti koje se koriste tijekom rada algoritma nad pojedinom instancom. Za parametar β korištene su vrijednosti 2, 3 i 5, a za ρ parametar vrijednosti iznose 0.1, 0.3 i 0.5 jer se za ρ parametar uzimaju vrijednosti u rasponu od 0 do 1 prema [1]. Kombinacije parametara za eksperiment prikazane su u tablici 4.3.

Tablica 4.3. Kombinacije promjenjivih parametara za izvođenje pojedine instance problema

β	**2**	**2**	**2**	**3**	**3**	**3**	**5**	**5**	**5**
ρ	0.1	0.3	0.5	0.1	0.3	0.5	0.1	0.3	0.5

Prilikom pokretanja programskog rješenja nad zadanom instancom problema i parametrima vrši se deskriptivna statistika nad dobivenim podatcima. Izračunava se prosječna vrijednost kvalitete rješenja S_{avg}, standardna devijacija σ i odstupanje vrijednosti prosječnog rješenja od optimuma Δ i pronalazi se najbolje S_{best} i najlošije rješenje S_{worst} među dobivenim podatcima.

4.2. Rezultati

Nakon provedene eksperimentalne analize u tablici 4.4. prikazani su dobiveni rezultati. U navedenoj tablici istaknuti su najbolji dobiveni rezultati pojedinih instanci problema sa zadanim parametrima.

Tablica 4.4. Rezultati eksperimentalne analize

	β	**ρ**	S_{avg}	**σ**	S_{best}	S_{worst}	**Δ [%]**
Problem 1	2	0.1	449.9333	21.1281	411	497	19.9822
	2	0.3	469.1333	20.4691	421	503	25.1022
	2	0.5	488.4333	21.0122	440	525	30.2489
	3	0.1	421.2667	14.4751	390	451	12.3378
	3	0.3	429.9667	20.9276	380	476	14.6578
	3	0.5	439.7000	17.1545	406	472	17.2533
	5	**0.1**	**395.5667**	**9.0910**	**380**	**413**	**5.4844**
	5	0.3	406.9333	13.1071	380	430	8.5156
	5	0.5	411.0000	16.8206	383	458	9.6000
Problem 2	2	0.1	732.8667	38.5752	649	795	28.7991
	2	0.3	771.1333	45.3297	653	875	35.5243
	2	0.5	784.5000	40.5058	700	856	37.8735
	3	0.1	645.0667	19.7719	597	677	13.3685
	3	0.3	682.9000	33.3010	620	764	20.0176
	3	0.5	712.4667	43.2171	625	783	25.2138
	5	**0.1**	**602.3333**	**9.6793**	**578**	**621**	**5.8582**
	5	0.3	616.4333	21.1608	577	682	8.3363
	5	0.5	634.3667	24.8924	580	692	11.4880
Problem 3	2	0.1	451.7667	2.2757	450	459	0.3626
	2	0.3	452.1667	2.7090	450	461	0.4815
	2	0.5	455.6333	5.1412	450	467	1.2519
	3	**0.1**	**450.3333**	**1.4682**	**450**	**458**	**0.0741**
	3	0.3	451.4667	2.8657	450	461	0.3259
	3	0.5	452.6333	3.5636	450	461	0.5852
	5	0.1	450.9333	2.4486	450	461	0.2074
	5	0.3	451.5000	2.2913	450	459	0.3333
	5	0.5	452.3000	3.3877	450	461	0.5111
Problem 4	2	0.1	245.9667	7.2087	230	262	16.0220
	2	0.3	247.1667	7.9418	226	257	16.5881
	2	0.5	246.9667	7.6659	224	259	16.4937
	3	0.1	230.7333	4.9795	222	240	8.8365
	3	0.3	234.0000	4.9933	221	246	10.3774
	3	0.5	231.0000	7.0427	212	241	8.9623
	5	**0.1**	**223.1333**	**4.6385**	**212**	**229**	**5.2516**
	5	0.3	224.0333	4.4383	212	230	5.6761
	5	0.5	224.6000	3.7929	216	230	5.9434
Problem 5	2	0.1	255.6333	10.2972	232	273	18.3488
	2	0.3	255.9557	7.9519	234	270	18.5031
	2	0.5	258.2333	8.7471	236	276	19.5525
	3	0.1	234.8333	6.6737	219	247	8.7191
	3	0.3	233.7667	7.000	218	249	8.2253
	3	0.5	236.6333	4.2149	228	245	9.5525
	5	**0.1**	**221.0667**	**3.6872**	**216**	**231**	**2.3457**
	5	0.3	221.8000	3.2187	217	229	2.6852
	5	0.5	222.0000	3.5496	216	229	2.7778

Analizom rezultata eksperimenta u tablicama uočavaju se veća odstupanja srednje vrijednosti od poznatog optimuma kod instanci s većim brojem klijenata što je posljedica težine problema promatrane instance. Kada se promatraju kombinacije parametara za sve instance, 8 od 9 rezultata prikazuje najveće odstupanje srednje vrijednosti od optimuma kod problema 2 koji ima najveći broj klijenata u ovome eksperimentu. Ako instanca ima manji broj klijenata, veća je mogućnost manjeg odstupanja srednje vrijednosti od optimalnog rješenja što je uočeno kod problema 3.

Promatrajući utjecaj parametra ρ, intenzitet isparavanja feromona prema izrazu (2.5.), manja odstupanja prosječne vrijednosti od optimalnog rješenja dobivena su u kombinacijama u kojima je vrijednost parametra ρ manja od 0.5 što je vidljivo kod problema 1, 2, 3 i 5. Kada je intenzitet isparavanja feromona prevelik, može doći do preferiranja pojedinih ruta i time je smanjena vjerojatnost da algoritam nađe bolja rješenja.

Budući da parametar β potencira vrijednosti heurističke informacije prema izrazu (2.4.), što je veća vrijednost parametra β, to je utjecaj vrijednosti heurističke informacije pri računanju vjerojatnosti odabira pojedinog klijenta manji. Kombinacije parametara u kojima je prisutan parametar β s najvećom vrijednošću pokazuje najmanja odstupanja srednje vrijednosti od optimuma što je vidljivo u problemima 1, 2, 4 i 5.

5. ZAKLJUČAK

Problem usmjeravanja vozila je problem kombinatorne optimizacije čiji je cilj pronaći najbolje moguće dostavne rute vozila od skladišta do svih klijenata. VRP-u se, zbog složenosti, pristupa korištenjem metaheuristike kao mogućeg učinkovitog načina njegovog rješavanja. Poznata metaheuristika za rješavanje VRP-a je algoritam optimizacije kolonijom mrava koji je inspiriran suradnjom u mravljoj koloniji u potrazi za hranom. Za programsko rješenje kapacitativnog problema usmjeravanja vozila korišten je elitistički mravlji sustav kao efikasna inačica algoritma optimizacije kolonijom mrava. Programsko rješenje ostvareno je u C# programskom jeziku. Eksperimentalna analiza, provedena pomoću ostvarenog programskog rješenja, prikazuje ovisnost kvalitete rješenja o ulaznim parametrima EAS algoritma za različite instance CVRP-a. Analiza, očekivano, u većini promatranih slučajeva potvrđuje da programsko rješenje daje kvalitetnije rezultate u instancama problema s manjim brojem klijenata i pri sporijem isparavanju feromona. Također je uočeno da veća vrijednost parametra koji određuje utjecaj heurističke informacije pridonosi rješenjima bolje kvalitete. U budućem radu eksperimentalna analiza mogla bi se nadopuniti testiranjem dodatnih kombinacija parametara i korištenjem primjera problema s većim brojem klijenata što bi dalo detaljnije rezultate i bolji uvid u ponašanje ovog algoritma. Moguća proširenja programskog rješenja obuhvaćala bi nadopunjavanje programa ugradnjom dodatnih algoritama koji koriste druge, iz literature poznate, verzije ACO algoritama. Time bi se u eksperimentalnoj analizi omogućila usporedba kvalitete rješenja dobivenih različitim algoritmima. Nove mogućnosti pružila bi i programska rješenja usmjerena na više različitih verzija problema usmjeravanja vozila.

LITERATURA

[1] P. Toth i D.Vigo, The vehicle routing problem, Philadelphia, USA: SIAM, 2002.

[2] A. P. Engelbrecht, Computational Intelligence: An Introduction, Second Edition, Wiley Publishing, 2007.

[3] Y. Bin, Y. Zhong-Zhen i Y. Baozhen, "An improved ant colony optimization for vehicle routing problem", *"European Journal of Operational Research"*, svez. 196, pp. 171-176, 2009.

[4] K. Leyton-Brown, H. H. Hoss i F. Hutter, "Understanding the Empirical Hardness of NP-Complete Problems", *"Communications of the ACM"*, svez. 57, br. 5, pp. 98-107, 2014.

[5] L. Y. O. Li i Z. Fu, "The school bus routing problem: A case study", *"Journal of the Operational Research Society"*, svez. 53, pp. 552-558, 2002.

[6] M. Dorigo i T. Stützle, Ant Colony Optimization, USA: Bradford Company, 2004.

[7] S. N. Kumar i R. Panneerselvam, "A survey on the vehicle routing problem and its variants", *"Intelligent Information Managment"*, svez. 4, br. 3, pp. 66-74, 2021.

[8] E. M. Toro, A. H. Escobar i M. Granada, "Literature review on the vehicle routing problem in the green transportation context", *"Luna Azul"*, svez. 42, pp. 362-387, 2016.

[9] Z. H. Ahmed, "A Lexisearch Algorithm for the Distance-Constrained Vehicle Routing Problem", *"International Jorunal of Mathematical and Computional Methods"*, svez. 1, pp. 165-174, 2016.

[10] M. Huang i P. Ding, "An improved ant colony algorithm and its application in vehicle routing problem", *"Mathematical Problems in Engineering"*, 2013.

[11] I. Xavier, "Capacitated Vehicle Routing Problem Library" (CVRLIB), "Hurricane Media", 2014. [Mrežno]. Available: http://vrp.atd-lab.inf.puc-rio.br/index.php/en/. [Pokušaj pristupa 2 8 2021].

SAŽETAK

Problem usmjeravanja vozila je problem kombinatorne optimizacije nastao na temelju potrebe za optimizacijom transporta u suvremenom svijetu. Jedan od načina za rješavanje ovog problema primjena je algoritma optimizacije kolonijom mrava kao popularne metaheuristike. Detaljnije je promatran kapacitativni problem usmjeravanja vozila zajedno s elitističkim mravljim sustavom na kojima se temelji ostvareno programsko rješenje problema. Programsko rješenje kreirano je u programskom jeziku C#. Objašnjen je način rada programskog rješenja i njegova uporaba. Provedena eksperimentalna analiza prikazala je kako različite vrijednosti parametara EAS algoritma utječu na kvalitetu pronađenih rješenja kapacitativnog problema usmjeravanja vozila.

Ključne riječi: elitistički mravlji sustav, kapacitativni problem usmjeravanja vozila, metaheuristika, optimizacija kolonijom mrava, problem kombinatorne optimizacije

ABSTRACT

An artificial ant colony algorithm for the vehicle routing problem

The vehicle routing problem is a combinatorial optimization problem based on the need for transport optimization in the modern world. One way of solving this problem is the use of ant colony optimization algorithm as a popular metaheuristic. Capacitated vehicle routing problem along with elitist ant system is observed in more detail and it's also a basis for the realized software solution. The realized software solution was written in the C# programming language. The mode of operation of the software solution and its use are explained. The conducted experimental analysis shows how different input parameter values of the EAS algorithm affect the found solution quality of the capacitated vehicle routing problem.

Key words: elitist ant system, capacitated vehicle routing problem, metaheuristic, ant colony optimization, combinatorial optimization problem

ŽIVOTOPIS

Tea Turjak rođena je 10. rujna 1999. godine u Našicama. Školovanje započinje 2006. u Osnovnoj školi Dore Pejačević Našice koje završava 2014. godine. Iste godine upisuje prirodoslovno-matematičku gimnaziju u Srednjoj školi Isidora Kršnjavog Našice. Srednjoškolsko obrazovanje završava 2018. godine nakon uspješno položene državne mature te upisuje sveučilišni preddiplomski studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku.

PRILOZI /na CD-u/

1. Završni rad u doc, docx i PDF formatu
2. Projekt programskog rješenja
3. Podatci korišteni za eksperimentalnu analizu