Optimizacija ruta vozila za potrebe istraživanja kvalitete mobilne mreže primjenom algoritama za rješavanje problema trgovačkog putnika

Dukić, Zoran

Master's thesis / Diplomski rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Transport and Traffic Sciences / Sveučilište u Zagrebu, Fakultet prometnih znanosti**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:119:790638

SVEUČILIŠTE U ZAGREBU

FAKULTET PROMETNIH ZNANOSTI

Zoran Dukić

Optimizacija ruta vozila za potrebe istraživanja kvalitete mobilne mreže primjenom algoritama za rješavanje problema trgovačkog putnika

DIPLOMSKI RAD

Zagreb, 2017.

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI
POVJERENSTVO ZA DIPLOMSKI ISPIT

Zagreb, 5. svibnja 2017.

Zavod: **Samostalne katedre**
Predmet: **Optimizacija prometnih procesa**

DIPLOMSKI ZADATAK br. 4400

Pristupnik: **Zoran Dukić (0036452436)**
Studij: Promet
Smjer: Informacijsko-komunikacijski promet

Zadatak: **Optimizacija ruta vozila za potrebe istraživanja kvalitete mobilne mreže primjenom algoritama za rješavanje problema trgovačkog putnika**

Opis zadatka:

U radu je potrebno prikazati na koji način servisna služba provodi istraživanje kvalitete mobilne mreže. Potrebno je objasniti problem trgovačkog putnika i primjenu algoritama za grupiranje.
Uz primjenu algoritama za rješavanje problema trgovačkog putnika potrebno je optimizirati rute vozila servisne službe koja se bavi istraživanjem kvalitete mobilne mreže i napraviti analizu rezultata dobivenih algoritmima za rješavanje problema trgovačkog putnika.

Zadatak uručen pristupniku: 23. svibnja 2017.

Mentor: dr. sc. Juraj Fosin

Predsjednik povjerenstva za diplomski ispit:

Sveučilište u Zagrebu
Fakultet prometnih znanosti

DIPLOMSKI RAD

Optimizacija ruta vozila za potrebe istraživanja kvalitete mobilne mreže primjenom algoritama za rješavanje problema trgovačkog putnika

Vehicle Routes Optimization for the Purpose of Mobile Network Quality Research by Applying Algorithms for Solving Traveling Salesman Problem

Mentor: dr. sc. Juraj Fosin
Student: Zoran Dukić, 0036452436

Zagreb, rujan 2017.

SAŽETAK

Cilj ovog diplomskog rada je pronaći rješenje po kojem se može minimizirati ukupna prijeđena udaljenost za poduzeće P3 Communications, koje se bavi istraživanjem kvalitete mobilne mreže. Da bi se dala generalna slika kvalitete mobilne mreže na nekom geografskom području, potrebno je provesti mjerenja različitih parametara na različitim mjestima, ali i na prometnicama koja povezuju ta mjesta. Često se radi o velikom broju mjesta koje treba posjetiti. Kako bi se obišla sva mjesta u što kraćem vremenu, poduzeće P3 Communications koristi više vozila. U svrhu minimiziranja prijeđene udaljenosti, izrađeno je programsko rješenje koje koristi algoritme za rješavanje problema trgovačkog putnika. Programsko rješenje koristi ukupno dva algoritma za rješavanje problema trgovačkog putnika i jedan algoritam za grupiranje. Prvi korišteni algoritam je algoritam k-srednjih vrijednosti, koji spada u algoritme particijskog grupiranja. Druga dva algoritma su algoritam najbližeg susjeda, koji spada u skupinu heurističkih algoritama, i 2-Opt algoritam koji poboljšava inicijalno rješenje dobiveno algoritmom najbližeg susjeda. Rješenja dobivena programskim rješenjem prikazana su slikama sučelja na kojem je iscrtana ruta kojom se svako vozilo treba kretati.

KLJUČNE RIJEČI : problem trgovačkog putnika; istraživanje kvalitete mobilne mreže; algoritam najbližeg susjeda; 2-Opt algoritam, grupiranje

ABSTRACT

The aim of this Master's Thesis is to find a way how to minimize the total distance traveled by P3 Communications company, a company that conducts mobile network quality research. In order to give a general picture of the quality of a mobile network for some geographic area, it is necessary to perform measurements of different parameters at different places but also on the roads linking these places. It is often a great number of places that needs to be visited. To visit all places in the shortest amount of time, company P3 Communications uses more than one vehicle. For the purpose of minimizing the total traveled distance, a software solution has been developed using algorithms based on solving traveling salesman problem. The solution uses a total number of three algorithms, two which are used for solving traveling salesman problem and one algorithm used for clustering. Algorithm used for clustering is called K-means algorithm, which is one of the partitions algorithms. The other two algorithms are nearest neighbour algorithm, a heuristics algorithm, and 2-Opt algorithm, which improves the initial solution obtained by the nearest neighbour algorithm. The solutions provided by the software are shown on a graphical user interface on which different routes are drawn, for each vehicle separately.

KEY WORDS: traveling salesman problem; mobile network quality research; nearest neighbour algorithm; 2-Opt algorithm

SADRŽAJ

1. UVOD

Problem trgovačkog putnika jedan je od osnovnih problema kombinatorne optimizacije. Poznata interpretacija tog problema je kako pronaći što kraću udaljenost koju trgovački putnik treba prijeći, a da pritom sva mjesta posjeti točno jednom i da se vrati u mjesto iz kojeg je krenuo. Iako se ovaj problem na prvu možda ne čini teškim, treba uzeti u obzir da on ima faktorijelnu složenost, odnosno da se najkraći put između N gradova nalazi negdje unutar prostora stanja koji je jako velik.

U ovom diplomskom radu uzet je za primjer način rada poduzeća P3 Communications koje se bavi istraživanjem kvalitete mobilne mreže. Za što točnije rezultate istraživanja, P3 Communications, korištenjem nekoliko vozila, posjećuje veliki broj lokacija određenog područja i mjeri određene parametre. Korištenjem algoritama za rješavanje problema trgovačkog putnika, poduzeće P3 Communications može optimizirati određene dijelove svojeg poslovanja, što bi u ovom konkretnom slučaju značilo minimizirati prijeđenu udaljenost i time smanjiti troškove kao što su troškovi putovanja ili utrošak vremena kako bi se posjetile sve lokacije.

Programsko rješenje korišteno u ovome radu koristi algoritam za grupiranje i dva algoritma za rješavanje problema trgovačkog putnika. Korišteni algoritam za grupiranje je algoritam k-srednjih vrijednosti, kojim će se odrediti koje sve lokacije svako vozilo treba posjetiti. Jedan od dva algoritama trgovačkog putnika korištena u programskom rješenju je algoritam najbližeg susjeda - heuristički algoritam koji se temelji na pohlepnom dodavanju najbližeg vrha već dodanim vrhovima u ruti. Drugi korišteni algoritam je 2-Opt algoritam koji će poboljšati inicijalno rješenje dobiveno algoritmom najbližeg susjeda. Sva tri algoritma koriste se u svrhu optimizacije poslovanja navedenog poduzeća.

2. NAČIN PROVOĐENJA ISTRAŽIVANJA KVALITETE MOBILNE MREŽE

Kvaliteta mobilne mreže se istražuje kako bi rezultati dobiveni tim istraživanjem telekom operaterima dali stvarni uvid u kvalitetu usluge koju pružaju kao i njihovu konkurentnost na tržištu. Isto tako, na temelju tih rezultata korisnici usluge mogu donijeti „bolje odluke“ kod odabira telekom operatera zato što rezultati istraživanja objektivno uspoređuju performanse različitih aspekata mobilnih mreža različitih operatera.

Kvaliteta mobilne mreže se provodi ispitivanjem različitih parametara kao što su parametri vezani uz kvalitetu podatkovnih usluga ili parametri vezani uz glasovne usluge. Istraživanje se provodi u različitim gradovima, mjestima, ali i na prometnicama koje povezuju ta područja čime se, između ostalog, ispituje i razina pokrivenosti signalom nekog operatera koristeći više mobilnih uređaja. Testiranje se provodi tako da se simuliraju potrebe i navike prosječnog korisnika pa tako mjerenje kvalitete podatkovnog prometa na pametnim telefonima obuhvaća pretraživanje Interneta na najpopularnijim stranicama kao što su YouTube, Facebook, Wikipedia te preuzimanje i slanje datoteka.

Za primjer možemo uzeti način provođenja istraživanja kvalitete mobilne mreže poduzeća P3 Communications na području Republike Hrvatske za tri vodeća telekom operatera (Hrvatski Telekom, Vip i TELE 2). P3 Communications je vodeće europsko poduzeće koje provodi neovisna istraživanja kvalitete mreža i usluga u telekom industriji. Certifikati koje P3 dodjeljuje široko su prihvaćeni kao nepristrani i reprezentativni rezultati testiranja mobilnih mreža zbog toga što se temelje na realnoj usporedbi svih konkurenata na pojedinom tržištu [1]. Na slici 1. možemo vidjeti gradove u kojima su se vršila testiranja glasovnih i podatkovnih usluga na teritoriju Republike Hrvatske iz 2016. godine provedeno od strane poduzeća P3 Communications.

Slika 1: Gradovi u kojima se vršilo testiranje na području RH provedeno od strane poduzeća P3 Communications, [2]

2.1. Ispitivanje kvalitete usluge glasovnih poziva

Za dobivanje rezultata kvalitete usluge glasovnih poziva, poduzeće P3 Communications na različitim geografskim područjima mjeri ukupno tri ključna pokazatelja uspješnosti (Key Performance Indicator – KPI): stopa uspješnosti uspostavljanja veze, vrijeme potrebno za uspostavljanje poziva, prosječna vrijednost kvalitete govora.

- Stopa uspješnosti uspostavljanja veze – upućuje se određeni broj poziva prema odredišnom mobilnom uređaju i računa se broj uspješno uspostavljenih veza u postocima. Vrijednost od 97% predstavlja dovoljnu stopu uspješnosti, vrijednost od 98% predstavlja dobru stopu uspješnosti dok vrijednost od 99% predstavlja odličnu stopu uspješnosti uspostave poziva.
- Vrijeme potrebno za uspostavljanje poziva – računa se vrijeme potrebno od zahtjeva za uspostavom poziva do ostvarivanja samog poziva.

- Prosječna vrijednost kvalitete govora – mjeri se na temelju POLQA (Perceptual Objective Listening Quality Assessment) standarda. POLQA je standard koji procjenjuje obrađeni govorni signal u odnosu na izvorni signal. Uspoređuje svaki uzorak referentnog signala (govornika) sa svakim odgovarajućim uzorkom degradiranog signala (strana slušatelja). Perceptivne razlike između oba signala se zatim boduju [3][4].

Kako bi provedena mjerenja bila što bliža stvarnom korisničkom iskustvu potrebno je držati se određenih preduvjeta:

- Pozivi se upućuju isključivo s jednog mobilnog uređaja prema drugom mobilnom uređaju.
- Ako je to moguće, koristi se ista generacija mobilne mreže prilikom uspostavljanja svih poziva (3G/4G).
- Ispitivani pozivi različitih operatera se obavljaju u istom vremenskom okviru od 115 sekundi.
- Vrijeme trajanja poziva za kojeg se ispituje kvaliteta traje 70 sekundi.
- Za vrijeme trajanja poziva, strani govornika i slušatelja se šalje po jedna elektronička pošta.
- Mjerenja se provode u gradovima, ali i na prometnicama koje ih povezuju [5].

Za potrebe ispitivanja kvalitete glasovnih poziva na teritoriju Republike Hrvatske iz 2016. godine korištena su dva automobila s po dva pametna mobilna uređaja (Samsung Galaxy S5) za svaki telekom operater. Mjerili su se gore navedeni parametri poštujući unaprijed definirane uvjete.

Tablica 1: Bodovna tablica za uslugu glasovnog poziva

Glasovni poziv	Max.	VIP	T-COM	TELE2
Gradovi	240	87%	86%	88%
Mjesta	80	88%	90%	90%
Ceste	80	84%	83%	65%
Ukupno	400	348	346	336

Izvor: [5]

Iz tablice 1. je na temelju provedenih mjerenja vidljivo kako je VIP osvojio najviše bodova (348/400) u utrci najbolje usluge u domeni glasovnih poziva.

Tablica 2: Ključni pokazatelji uspješnosti (KPI) za uslugu glasovnog poziva

Podatkovne usluge	KPI	Mjerna jedinica	VIP	T-COM	TELE2
Gradovi	1	%	99,2	98,9	99,7
	2	Sekunda	6,4	6,4	6,9
	3	MOS-LQ	3,7	3,8	3,7
Mjesta	1	%	99,4	99,6	99,8
	2	Sekunda	6,4	6,3	6,8
	3	MOS-LQ	3,7	3,8	3,7
Ceste	1	%	98,0	97,4	92,8
	2	Sekunda	6,4	6,4	7,0
	3	MOS-LQ	3,7	3,8	3,6

Izvor: [5]

U tablici 2. možemo vidjeti vrijednosti pojedinog operatera za različite pokazatelje uspješnosti (1 - Stopa uspješnosti uspostavljanja veze, 2 – Vrijeme potrebno za uspostavljanje poziva, 3 – Prosječna vrijednost kvalitete govora) za usluge glasovnih poziva. Mjerna jedinica MOS-LQ (Mean opinion score-Listening Quality) govori kolika je prosječna vrijednost kvalitete govora na temelju POLQA standarda. Vrijednosti za MOS-LQ su između 1-5. Što je vrijednost veća to je i veća kvaliteta govora.

2.2. Ispitivanje kvalitete usluge podatkovnog prometa

Za dobivanje rezultata kvalitete usluge podatkovnog prometa, tvrtka P3 Communications na različitim geografskim područjima mjeri ukupno četiri ključna pokazatelja uspješnosti: pretraživanje Interneta, preuzimanje podataka, prijenos podataka, korištenje YouTube servisa [5].

- Pretraživanje Interneta – popularno „surfanje“, izvodi se na način da se pretražuje ukupno 10 web stranica od kojih je nekoliko statičnih, a nekoliko dinamičnih te se mjeri vrijeme potrebno da se stranica učita.
- Preuzimanje podataka – izvodi se na način da se sa testnog servera preuzima datoteka veličine 3 MB (megabyte) i pohranjuje na uređaj sa kojeg se vrši ispitivanje. Ključni pokazatelj je vrijeme koje je potrebno da bi se takva datoteka preuzela.
- Prijenos podataka - izvodi se na način da se na testni server prenosi datoteka veličine 1 MB sa uređaja sa kojeg se vrši ispitivanje. Ključni pokazatelj je vrijeme koje je potrebno da bi se takva datoteka prenijela.
- Korištenje Youtube servisa – izvodi se na način da se pokrene određeni Youtube sadržaj u trajanju od 45 sekundi. Za vrijeme tog trajanja mjere se sljedeći parametri: vrijeme potrebno da se sadržaj učita, broj prekida, prosječna rezolucija.

Kod ispitivanje kvalitete podatkovnih usluga se također treba pridržavati određenih preduvjeta za svaki telekom operater kao što su odabir generacije mobilne mreže (4G ako je dostupna), jednaki broj posjećenih web stranica, jednake veličine datoteka za preuzimanje i prijenos kao i vrijeme trajanja Youtube sadržaja [5].

Tablica 3: Bodovna tablica za uslugu podatkovnog prometa

Podatkovni promet	Max.	VIP	T-COM	TELE2
Gradovi	360	82%	83%	85%
Mjesta	120	76%	82%	86%
Ceste	120	74%	75%	71%
Ukupno	600	476	487	492

Izvor: [5]

Iz tablice 3. se može iščitati kako je u kategoriji podatkovnih usluga na području za koje se vršilo ispitivanje, operater TELE2 ostvario najveći broj bodova. Od mogućih 600 osvojio je 492 boda.

Tablica 4: Ključni pokazatelji uspješnosti za podatkovnu uslugu

Podatkovne usluge	KPI	MAX	VIP	T-COM	TELE2
Gradovi	Pretraživanje weba	126	80%	79%	77%
	Preuzimanje podataka	72	82%	85%	83%
	Prijenos podataka	72	81%	78%	90%
	Youtube	90	85%	92%	92%
Mjesta	Pretraživanje weba	126	78%	82%	80%
	Preuzimanje podataka	72	73%	82%	86%
	Prijenos podataka	72	75%	74%	88%
	Youtube	90	78%	88%	94%
Ceste	Pretraživanje weba	126	73%	75%	64%
	Preuzimanje podataka	72	75%	81%	78%
	Prijenos podataka	72	72%	68%	68%
		90	78%	77%	76%

Izvor: [5]

U tablici 4. možemo vidjeti vrijednosti za svaki ključni pokazatelj uspješnosti po telekom operateru. Može se zaključiti kako operater TELE2 svoju glavnu prednost nad ostalim konkurentima ima u kategoriji prijenosa podataka u gradovima kao i prijenosa i preuzimanja podataka u manjim mjestima.

3. Problem trgovačkog putnika

Problem trgovačkog putnika (Traveling Salesman Problem, TSP) je problem diskretne i kombinatorne optimizacije. Spada u skupinu NP-teških problema (non-polynomial hard problems), a složenost mu je O(N!) odnosno ima faktorijelnu složenost pa je zahtjevan za ljude ali i za računala.
U literaturi se spominju različite formulacije problema trgovačkog putnika, međutim njegov osnovni oblik bismo mogli formulirati na dva načina:

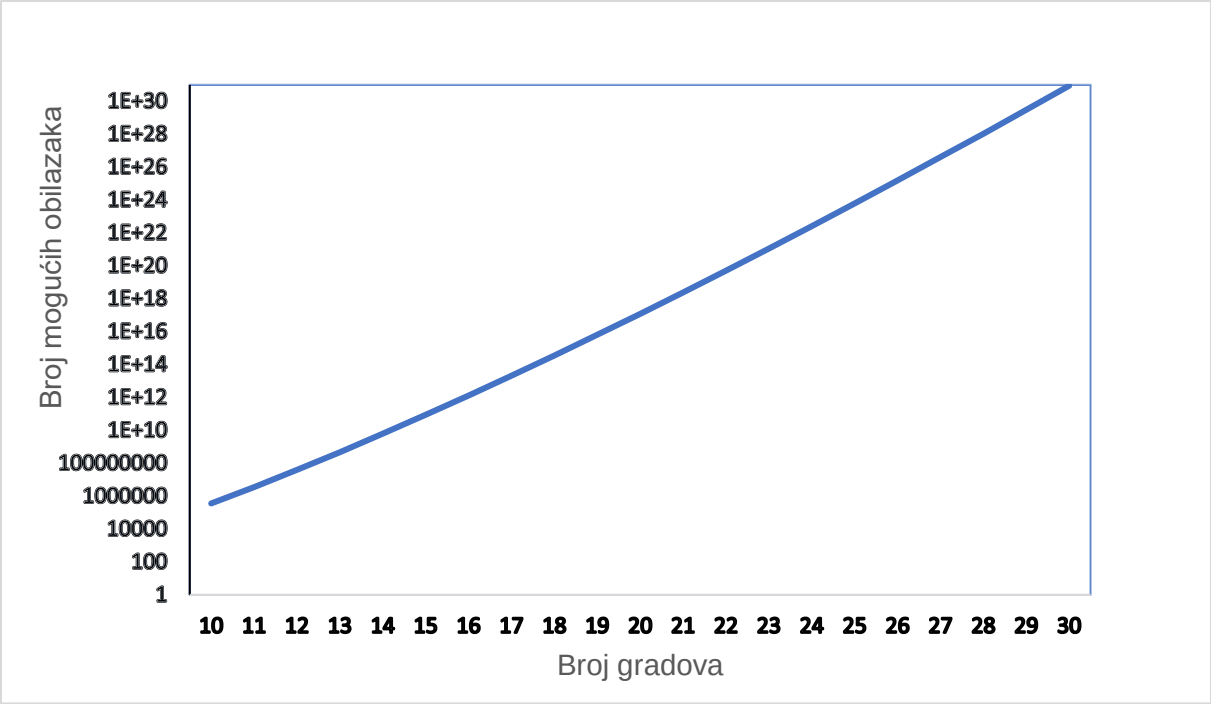
1. Trgovački putnik ima zadan skup vrhova u određenom grafu od kojih svaki mora posjetiti točno jedanput i vratiti se u početni vrh. Udaljenosti među vrhovima su poznate. Postavlja se pitanje kojim bi redoslijedom trgovački putnik trebao obilaziti vrhove tako da ukupna duljina puta bude minimalna.
2. U težinskom grafu treba pronaći Hamiltonov ciklus minimalne težine

Kako bismo bolje razumjeli teoriju problema trgovačkog putnika, potrebno se najprije upoznati sa nekoliko definicija grafova:

- Graf je uređena trojka G = (V, E, φ), gdje je V = V(G) ne prazan skup čije elemente nazivamo vrhovima, E = E(G) je skup disjunktan s V čije elemente nazivamo bridovima i φ je funkcija koja svakom bridu e iz E pridružuje par {u, v}, ne nužno različitih vrhova u,v ∈ V. Graf skraćeno označavamo G = (V, E) ili samo G.
- Hamiltonov ciklus je ciklus u neusmjerenom grafu koji prolazi kroz sve vrhove grafa isključivo jedanput, osim vrha koji je početni i krajnji vrh ciklusa.
- Hamiltonov graf je graf koji sadrži Hamiltonov ciklus. Kada graf sadrži više Hamiltonovih ciklusa i ako želimo pronaći najkraći ciklus onda se taj problem naziva problemom trgovačkog putnika.

Dakle, da bi graf uopće imao rješenje problema trgovačkog putnika, mora imati barem jedan Hamiltonov ciklus, odnosno mora biti Hamiltonov graf.

U potpunim grafovima s n vrhova broj Hamiltonovih ciklusa je (n-1)! (u neusmjerenim grafovima (n-1)!/2, s obzirom da su po dvije rute identične ali inverzne) te rastu eksponencijalno s brojem vrhova (Graf 1).



Graf 1: Složenost problema trgovačkog putnika, [Izradio autor]

Algoritam za rješavanje problema trgovačkog putnika je u prvu ruku jednostavan za izračunati, međutim na primjeru Grafa 1. je vidljivo kako takav izračun za još uvijek mali broj gradova (10-30) daje jako puno kombinacija kako bi se našlo pravo rješenje. Zato takav problem svrstavamo u skupinu NP-teških problema. Riječ je o zadacima koji se ne mogu rješavati u polinomskom vremenu t = Nk, gdje je N dimenzija problema, a k konstanta. NP-teški problemi ne omogućuju jednostavnu provjeru dobivenog rješenja i ne znamo da li postoji bolje [6].

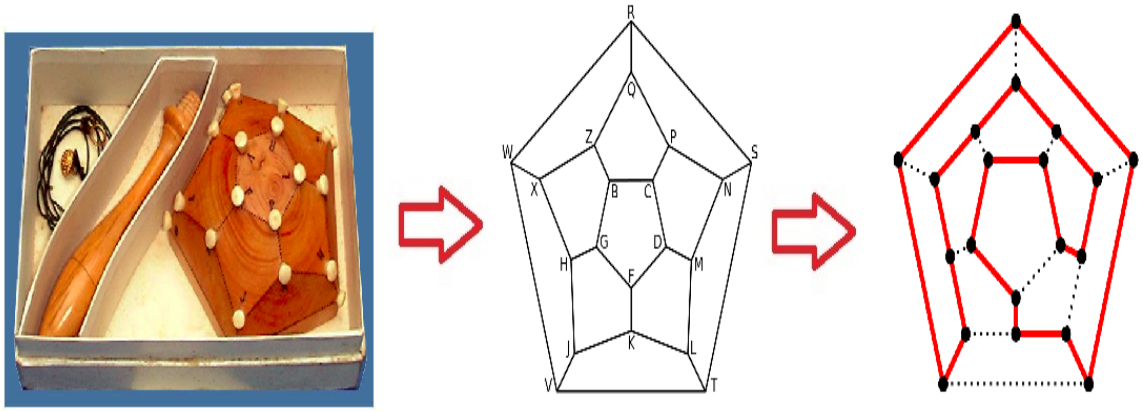
3.1. Povijest trgovačkog putnika

Porijeklo ovog problema seže daleko u povijest. Matematičke probleme slične problemu trgovačkog putnika je počeo razmatrati Euler, kojeg je zanimalo kako bi skakač na šahovskoj ploči posjetio sva 64 mjesta samo jednom (slika 2). Prva doslovna pojava problema trgovačkog putnika bila je u knjizi njemačkog trgovačkog putnika B.F. Voighta, 1832. Autor knjige sugerira kako je pokrivanje što više područja bez posjećivanja ijednog od njih dvaput najvažniji aspekt planiranja putovanja. No ni on nije problem trgovačkog putnika tako imenovao [6].

38	35	62	25	60	23	10	7
63	26	37	34	11	8	59	22
36	39	28	61	24	57	6	9
27	64	33	40	5	12	21	58
50	29	4	13	48	41	56	19
1	14	49	32	53	20	47	44
30	51	16	3	42	45	18	55
15	2	31	52	17	54	43	46

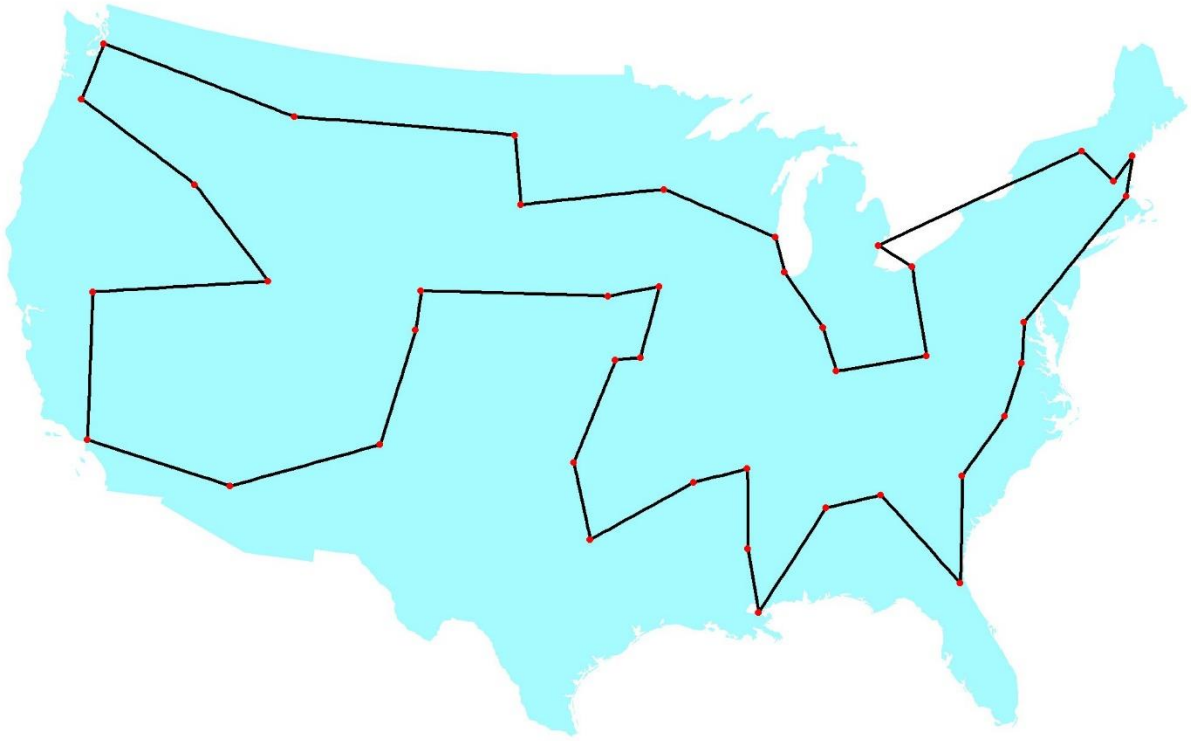
Slika 2: Igra skakača na šahovskoj ploči, [7]

Početkom 20.st. matematičari William Rowan Hamilton i Thomas Kirkman su razmatrali probleme koji se svode na problem trgovačkog putnika. Hamilton je 1857. godine osmislio igru (slika 3.) nazvana „Hamiltonova igra“ (Icosian game) koja je od igrača tražila da odredi stazu između 20 točaka uz korištenje definiranih veza između njih. Opća forma ovog problema se pojavljuje 30-ih godina 20. stoljeća zahvaljujući Karl Mengeru kada je i sam pojam „trgovački putnik“ prvi put upotrebljen [6].



Slika 3: Hamiltonova igra, [8]

Prvi veliki problem trgovačkog putnika koji je bio riješen je bio problem s 42 američka gradova, kojeg su riješili Dantzig, Fulkerson i Johnson (Slika 4.).



Slika 4: Problem trgovačkog putnika s 42 gradova, [9]

Međutim, od 1954. razvijane su efikasnije metode kako bi se problem riješio na većem uzorku gradova pa je tako 2006. godine problem trgovačkog putnika riješen na uzorku od 85.900 gradova.

3.2. Metode rješavanja

Metoda koja se prva nameće kada razmišljamo o rješavanju ovakvih problema je jednostavno pronaći sve moguće Hamiltonove cikluse odnosno sve moguće obilaske, izračunati njihovu duljinu odnosno težinu i odabrati najbolji. Međutim kako smo već komentirali Graf 1. iz kojeg je vidljivo da takva metoda kao rezultat daje veliki broj kombinacija što zahtjeva veliku procesnu snagu računala i veliki utrošak vremena kako bi se našao optimalan Hamiltonov ciklus, razvijane su mnoge aproksimativne metode koje relativno brzo daju prihvatljivo dobro rješenje. Modernim metodama moguće je u razumnom vremenu naći rješenje i za probleme od nekoliko milijuna gradova, koje je s velikom vjerojatnošću vrlo blizu optimalnog rješenja. Te metode mogu biti egzaktne, heurističke i metaheurističke [10].

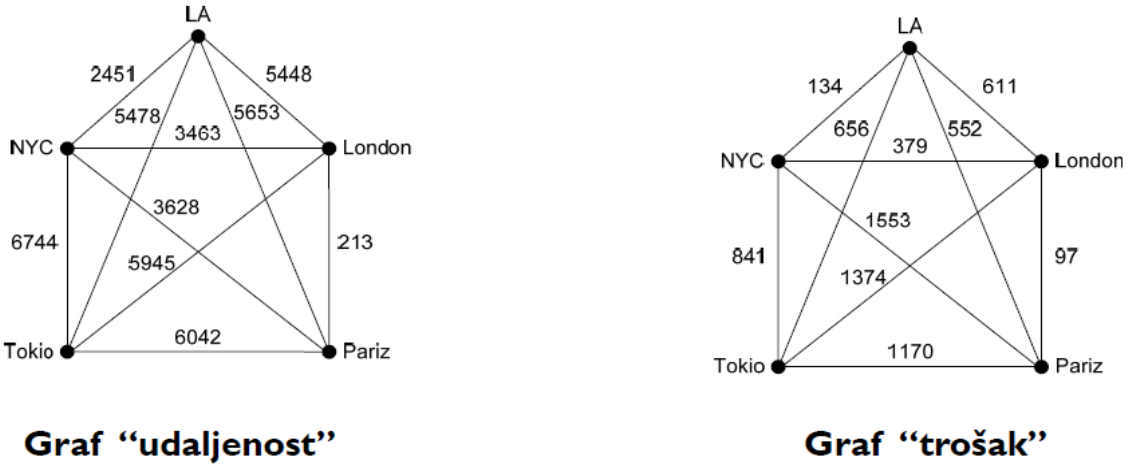
3.2.1 Egzaktne metode rješavanja

Egzaktne metode pronalaze optimalno rješenje, ali zbog svoje velike računalne složenosti ograničene su na manje probleme. Najjednostavniji egzaktni algoritam bio bi da se ispituju sve moguće kombinacije i izabere se najpovoljnija. Ova metoda se naziva „gruba sila“ (Brute Force) i ne efikasna je čak i za instance problema od 15 točaka. Osim Brute Force metode tu su još i metode linearnog programiranja, metoda grananja i ograničavanja (Branch & Bound), te metoda grananja i rezanja (Branch & Cut) [11].

3.2.1.1 Brute Force metoda

Brute Force metoda spada u skupinu egzaktnih metoda za pronalazak optimalnog rješenja. Njome se ispituju sve moguće kombinacije i zatim se izabire najpovoljnija, bilo da je riječ o najmanjoj udaljenosti ili npr. najjeftinijoj ruti. Ovakva metoda je pogodna za mali broj točaka jer uvijek daje optimalno rješenje ali za veliki broj točaka ovakav algoritam treba jako puno vremena kako bi izračunao rutu stoga se koristi samo za mali broj točaka.

Pogledajmo na primjeru određenog grafa (graf „udaljenost“ i graf „trošak“) kako je uz pomoć Brute Force metode moguće pronaći optimalno rješenje.

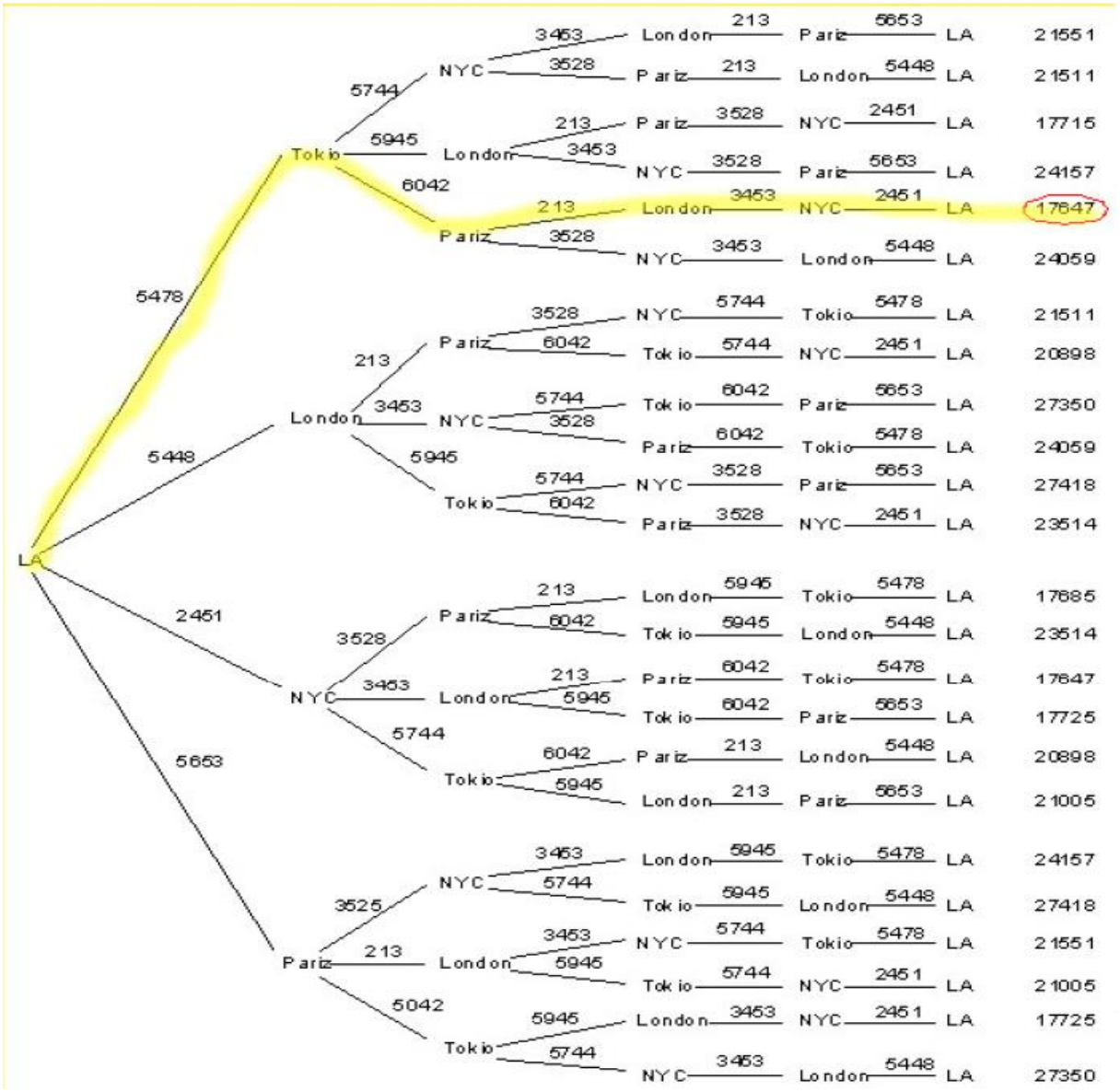


Slika 5: Graf „udaljenost“ i graf „trošak“, [12]

Ako na primjeru slike 5. gdje su udaljenosti između gradova težine na grafu, želimo pronaći optimalan put potrebno je provesti nekoliko koraka:

- pronaći sve moguće Hamiltonove cikluse odnosno sve puteve koji prolaze kroz svaki vrh točno jedanput osim početnog vrha koji je ujedno i završni vrh,
- pronaći duljinu svakog ciklusa dodavanjem težine bridova,
- odabrati ciklus s minimalnom ukupnom težinom. [12]

Nakon odrađenih svih koraka, optimalan ciklus, odnosno ciklus s najmanjim prijeđenim putem, možemo vidjeti na slici 6. označen žutom bojom.



Slika 6: Svi Hamiltonovi ciklusi za zadani primjer, [12]

Ukoliko želimo dobiti optimalnu rutu, ali ovisno o grafu „trošak“ tada je dovoljno slijediti iste korake kao i kod grafa „udaljenosti“.

3.2.1.2 Metoda linearnog programiranja

Linearno programiranje je optimiziranje linearne funkcije cilja, koja je podvrgnuta ograničenjima linearne jednakosti i nejednakosti. Ono traži način kako postići najbolji rezultat upotrebljavajući listu zahtjeva koja je opisana u linearnim jednadžbama. To je formalni postupak optimizacije sustava kod kojih se funkcija cilja i ograničenja mogu izraziti linearnim kombinacijama promjenjivih veličina. Omogućuje uspješno rješavanje problema veličine do 200 gradova [6].

3.2.1.3 Metoda grananja i ograničavanja

Grananje i ograničavanje (Branch and Bound) je opći algoritam za traženje optimalnih rješenja mnogih optimizacijskih problema. Sastoji se od brojanja svih potencijalnih rješenja, pri čemu se veliki podskupovi loših rješenja odbacuju. Loša rješenja su ona koja su unutar nekog intervala, tj. između predefinirane gornje i donje granice. Redoslijed obilaska gradova trgovačkog putnika prvo se rekurzivno grana na dva dijela, te nastaje struktura stabla. Nakon toga se računaju gornja i donja granica za svaki dobiveni podskup. Ako je donja granica nekog vrha bolja od gornje granice nekog drugog vrha, taj drugi vrh se izbacuje iz pretrage. Rekurzija se zaustavlja kad se skup reducira na jedan element ili kad njegova gornja granica postane jednaka njegovoj donjoj granici [6].

3.2.2 Heurističke metode rješavanja

Heurističke metode koriste algoritme koji daju približna rješenja. U relativno kratkom vremenu mogu se dobiti dovoljno dobra rješenja. Ovi algoritmi zahtijevaju manje vremena za izvršavanje ali zato ne garantiraju optimalno rješenje. Najpoznatiji heuristički algoritmi su: algoritam najbližeg susjeda (Nearest Neighbour algoritam), pohlepna heuristika (Greedy heuristics), algoritam umetanja najbližeg, algoritam umetanja najudaljenijeg, algoritam umetanja najjeftinijeg.

3.2.2.1 Algoritam najbližeg susjeda

Algoritam najbližeg susjeda je heuristički algoritam koji se temelji na pohlepnom dodavanju najbližeg vrha već dodanim vrhovima u ruti.

Ideja algoritma je:

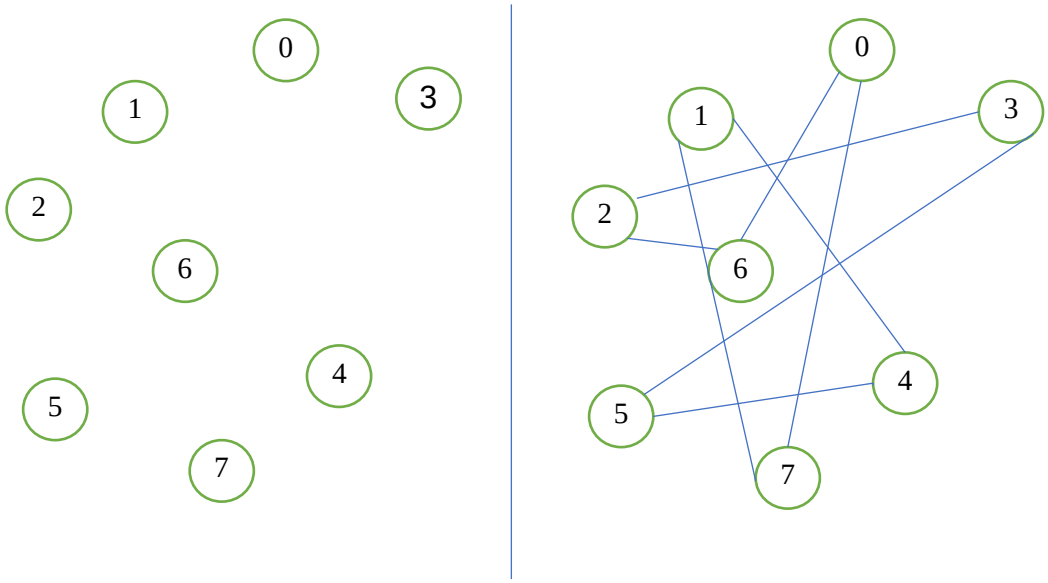
- odabrati početnu točku,
- premjestiti se u najbliži neposjećeni vrh (brid s najmanjom težinom),
- ponavljati prethodni korak dok se ciklus ne završi.[12]

Tablica 5: Nasumično odabrane udaljenosti za 8 vrhova

	0	1	2	3	4	5	6	7
0	/	5	12	4	7	9	6	2
1	5	/	5	8	3	7	4	3
2	12	5	/	3	8	12	5	8
3	4	8	3	/	4	5	7	6
4	7	3	8	4	/	2	3	8
5	9	7	12	5	2	/	6	4
6	6	4	5	7	3	6	/	8
7	2	3	8	6	8	4	8	/

Izvor: [Izradio autor]

Kao početna točka uzima se vrh 0 (može se odabrati bilo koji drugi vrh) te se u tablici 5. traži najbliži vrh početnom vrhu 0. U ovom slučaju to je vrh 7 čija je udaljenost 2. Nakon toga se traži najbliži vrh vrhu 7, a u ovom slučaju to je vrh 2. Zatim se postupak ponavlja sve dok svi vrhovi nisu spojeni s tim da se treba paziti da se svaki vrh obiđe točno jedanput osim vrha 0 koji treba biti i završni vrh.



Graf 2: Rješenje algoritma najbližeg susjeda, [Izradio autor]

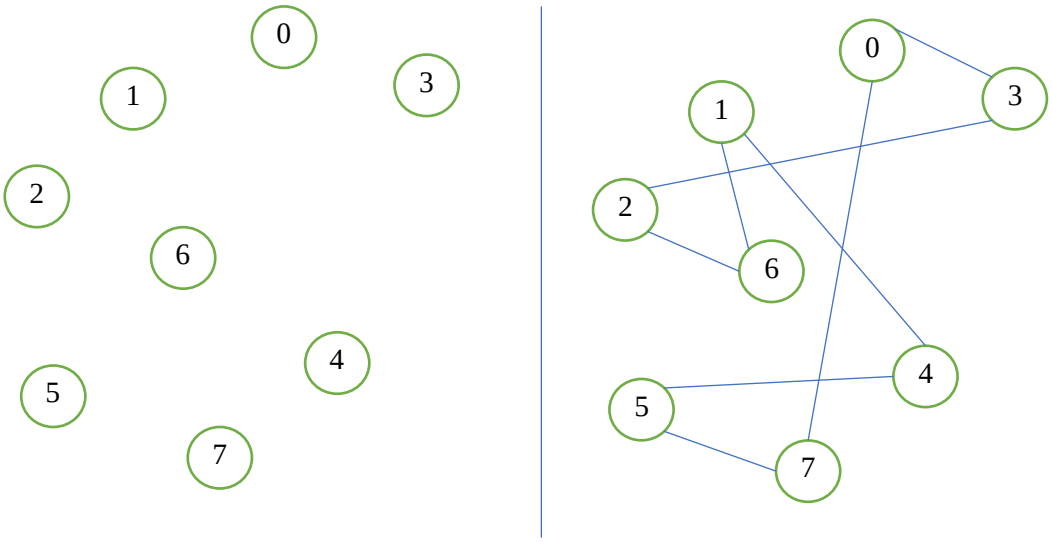
Dobivena ruta je 0,7,1,4,5,3,2,6,0 s ukupnom duljinom 2+3+3+2+5+3+5+6=29. Vremenska složenost ovog algoritma za problem od n gradova je $O(n^2)$. Ovaj algoritam zbog svoje jednostavnosti ne daju uvijek optimalno rješenje.

Tako se primjerice često posjećuje neki grad prerano što onemogućuje pronalazak boljeg rješenja kasnije.

3.2.2.2 Pohlepna heuristika

Pohlepna heuristika je naziv za drugu vrstu algoritma problema trgovačkog putnika. Ovom heuristikom promatra se potpuni graf s *n* vrhova i bridovima duljine C_{ij} između svaka dva para vrhova *i* i *j*. Ruta trovačkog putnika u takvom je grafu Hamiltonov ciklus u kojem je stupanj svih vrhova 2.

Izgradnja rute je u koracima, dodavanjem bridova krenuvši od najmanjeg. Pritom se izostavlja brid koji je već u ruti, koji bi zatvorio Hamiltonov ciklus s manje od *n* vrhova ili bi stvorio vrh trećeg stupnja [13].



Graf 3: Rješenje algoritma pohlepne heuristike, [Izradio autor]

Dobivena ruta je 0,7,5,4,1,6,2,3,0 ukupne duljine 27. Pohlepna heuristike je kompleksnosti $O(n^2 logn)$, dakle nešto sporija od algoritma najbližeg susjeda, s nešto boljim rezultatom.

3.2.2.3 Algoritam umetanja najbližeg

Algoritam umetanja najjeftinijeg (nearest insertion algorithm) je jedan od algoritama umetanja kod kojih se u trenutnu rutu umeće jedan od preostalih vrhova prema određenom kriteriju:

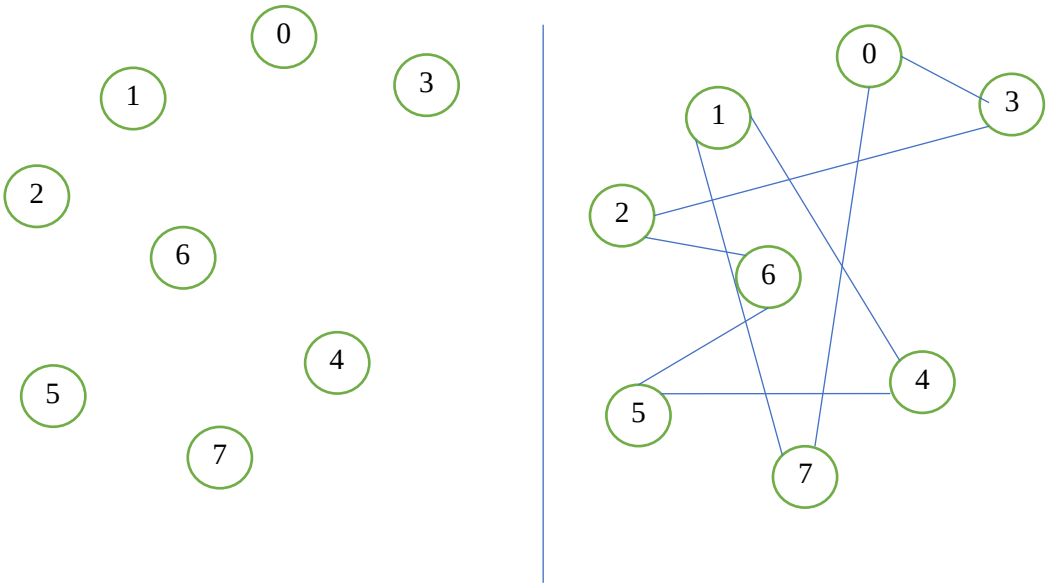
Korak 1: Odaberi vrh i za početni vrh 0,

Korak 2: Pronađi vrh *k* takav da je *c0k* minimalno (vrh najbliži početnom) i formiraj trenutnu rutu (0,*k*,0),

Korak 3: Nađi vrh *h* koji nije u trenutnoj ruti, a najbliži je nekom vrhu u trenutnoj ruti (korak odabira vrha),

Korak 4: Nađi brid (*i,j*) u trenutnoj ruti koja minimizira *cih+chj-cij*. Umetni vrh *h* između *i* i *j* (korak umetanja vrha),

Korak 5: Ponavljaj korak 3 sve dok svi vrhovi nisu u ruti. [13]



Graf 4: Rješenje algoritmom umetanja najbližeg, [Izradio autor]

Dobivena ruta je 0,7,1,4,5,6,2,3,0 a udaljenost je 28.

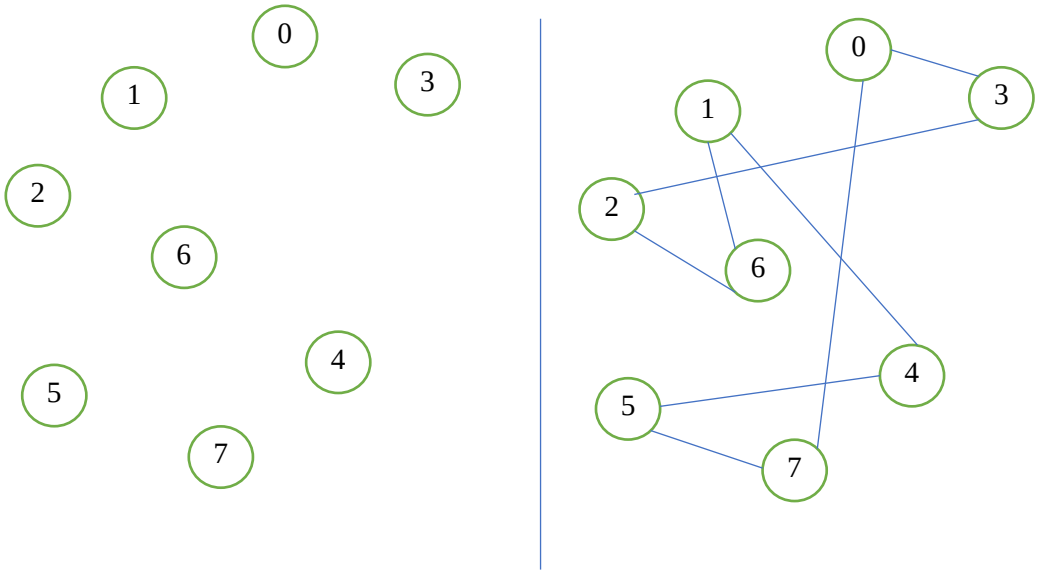
3.2.2.4 Algoritam umetanja najudaljenijeg

Algoritam umetanja najudaljenijeg (farthest insertion algorithm) je također algoritam umetanja pri čemu je kriterij za odabir vrha koji se dodaje trenutnoj ruti najveća udaljenost od nekog vrha trenutne rute.

Algoritam je sljedeći:

- Odaberi vrh i za početni vrh 0
- Pronađi vrh k takav da je c0k maksimalno (vrh najdalji početnom) i formiraj trenutnu rutu (0,k,0).

- Nađi vrh h koji nije u trenutnoj ruti, a najudaljeniji je od nekog vrha u trenutnoj ruti (korak odabira vrha).
- Nađi brid (i,j) u trenutnoj ruti koja minimizira cih+chj-cij.Umetni vrh h između i i j (korak umetanja vrha).
- Ponavljaj korak 3 sve dok svi vrhovi nisu u ruti. [13]



Graf 5: Rješenje algoritmom umetanja najudaljenijeg, [Izradio autor]

Dobivena ruta je 0,7,5,4,1,6,2,3,0, a udaljenost je 27. Algoritam umetanja najudaljenijeg generalno treba dati bolji rezultat od algoritma umetanja najbližeg zato jer algoritam umetanja najbližeg zbog svoje „pohlepne“ karakteristike u svakom koraku teži minimalnom povećanju troška, što u konačnici ne rezultira maksimalnom uštedom.

3.2.2.5 Algoritam umetanja najjeftinijeg

Kod algoritma za umetanje najjeftinijeg (cheapest insertion algorithm) postupak je isti kao i kod algoritma za umetanje najbližeg, osim što se za umetanje ne odabire najbliži vrh već onaj vrh za koji je trošak umetanja minimalan. Kompleksnost ovog algoritma je $O(n^2 logn)$.

3.2.3 Metaheurističke metode rješavanja

Metaheurističke metode rješavanja često pronalaze prihvatljivo rješenje u kratkom vremenu i primjenjive su na velikom broju kombinatornih optimizacijskih problema. One pokušavaju pronaći ravnotežu između diverzifikacije i intenzifikacije

lokalne pretrage. Lokalno pretraživanje je heuristička strategija za rješavanje problema kod koje se trenutno rješenje nastoji poboljšati na način da se izabere ono rješenje iz okoline koje je bolje od trenutnog i to se izabrano rješenje proglašava novim trenutnim rješenjem. Mehanizam diversifikacije nastoji istražiti što veći prostor rješenja dok mehanizam intenzifikacije želi što više iskoristiti pojedino područje u potrazi za lokalnim optimumima. Glavni problem metaheurstika upravo je u balansu ta dva mehanizma – nije se dobro previše zadržavati u pojedinom području pretrage a također nije dobro ga niti pre brzo napuštati bez da se pronađe najmanji lokalni ekstrem [14].

Mnogi metaheurističke algoritme uzimaju kao podvrstu heurističkih metoda. Oni kao i heurističke metode ne garantiraju pronalazak optimuma ali daju zadovoljavajuća rješenje relativno brzo. Jedina razlika je što metaheuristike nalaze inspiraciju u prirodi čija iskustva preuzimaju, simuliraju i na taj način daju rješenja. Za njihov rad potrebno je koristiti neku od konstruktivnih heuristika, a ponekad koriste i spomenute heurstičke metode lokalne pretrage za poboljšavanje rješenja. Najpoznatiji primjeri metahustističkih metoda su simulirano kaljenje (Simulated Annealing), genetski algoritmi (Genetic Algorithm), te metode optimizacije mravljom kolonijom (Ant Colony Optimization) [11].

3.2.3.1 Simulirano kaljenje

Simulirano kaljenje je heuristički algoritam koji oponaša pronalazak stanja minimalne energije u procesu kaljenja metala. To se postiže oponašanjem učinka temperature u gibanju čestica kroz stanja različitih energija i postepenim smanjivanjem te temperature. U simuliranom kaljenju gleda se ponašanje samo jedne čestice (rješenja) u procesu kaljenja. U svakoj iteraciji, za stanje „*s“* iz prethodne iteracije, bira se proizvoljno susjedno stanje s'. Ukoliko je energija stanja s' manja od energije stanja s, čestica prelazi u stanje s′. Ako je stanje s′ stanje više energije, s prelazi u s′ uz vjerojatnost koja ovisi o temperaturi. Algoritam kreće od neke zadane "visoke" temperature te ju kroz iteracije smanjuje do "apsolutne nule". Funkcija vjerojatnosti prelaska iz stanja niže u stanje više energije može biti bilo koja funkcija uz ograničenja da je ta funkcija pozitivna te da njena vrijednost opada s padom temperature i porastom razlika energija između stanja. Time se nastoji izbjeći loša karakteristika pohlepnog algoritma: zaustavljanje u lokalnom minimumu koji je puno lošiji od globalnog minimuma [15].

3.2.3.2 Genetski algoritmi

Genetski algoritmi podskup su evolucijskih algoritama. Inspiracija ovakvih algoritama pronađena je u prirodnom produktu evolucije. Osnovni cilj genetskih algoritama je pronalaženje rješenja nekog problema koje tradicionalne determinističke metode ne mogu riješiti. Genetski algoritmi svoju pretragu započinju od cijelog niza potencijalnih rješenja. Ta su rješenja najčešće generirana slučajnim odabirom te predstavljaju početnu populaciju. Početnoj se populaciji određuje dobrota koja je mjera kvalitete i govori koliko je neko rješenje dobro – bolja rješenja imati će veću dobrotu. Dobrota se određuje funkcijom cilja. Kada se odredi dobrota, može započeti proces križanja.

Početna populacija rješenja se iterira kroz generacije, a primjenjuje se princip preživljavanja najboljeg. U svakoj se generaciji odabiru bolja rješenja i odbacuju ona lošija. Odabrana rješenja su podvrgnuta genetskim operatorima. Prvo se križaju, a zatim s određenom vjerojatnošću i mutiraju. Tako nastaju nova potencijalna rješenja koja predstavljaju novu generaciju genetskog algoritma. Ta nova rješenja su u pravilu bolja nego stara rješenja od kojih su nastala. Završetak rada genetskog algoritma ne znači da je pronađeno optimalno rješenje već samo da je nađeno dovoljno dobro rješenje koje može ali i ne mora biti najbolje [6].

3.2.3.3 Optimizacija kolonijom mrava

Promatranjem mrava u prirodi otkriveno je da kad pronađu hranu iza sebe ostavljaju trag feromona. Drugi mravi koji osjete feromon prate ih do hrane i vraćaju se do mravinjaka. Ukoliko za nekim mravom osjete jači feromon radije idu za njim nego za tragom feromona koji je slabiji. Na taj je način i nastao algoritam temeljen promatranjem mrava u prirodi. Umjetni mravi (softverski agenti) za razliku od mravlje jedinke u prirodi imaju mogućnost pamćenja i odlučivanja. Pretražuju prostor u svim smjerovima i spremaju trenutno stanje u matricu. Što je put kraći, to stanje ostaje duže u matrici i s vremenom se na najkraćem putu nakupi najveća koncentracija feromona. Taj put ujedno je i najkraći put.

4. Primjena algoritama za grupiranje

Na primjeru poduzeća P3 Communication iz 2016. godine za potrebe testiranja kvalitete mobilne mreže korišteno je više vozila. Svako vozilo je trebalo proći određena područja i ispitati kvalitetu mobilne mreže. Primjenom algoritama za grupiranje svakom vozilu se dodjeljuju lokacije koja treba posjetiti kao što je objašnjeno u poglavlju 4.3. Kasnije, u poglavlju 5. ovog rada će se za svaku od tih grupa izvršiti algoritam za rješavanje problema trgovačkog putnika te će se za svako vozilo znati optimalna ruta. Osim primjera koji će se obraditi u ovom poglavlju, potrebno je naglasiti da algoritmi za grupiranja imaju sve veću primjenu u modernome svijetu. Jedna od tih primjena je kod analiziranja velikih količina podataka. Veliki servisi kao što su Facebook, Yahoo, Google, Amazon prikupljaju podatke svojih korisnika na temelju određenih obilježja. Podaci se čuvaju u velikim distribuiranim bazama podataka. Postavlja se pitanje kako analizirati tako velike količine podatak i iz njih dobiti bitne zaključke. Problem je još veći kada znamo da su podaci u bazama spremljeni bez neke organizacije. Dakle, prvi korak u analizi tako velike baze neuređenih podataka je grupiranje. Grupiranje podataka omogućuje podjelu podataka na manje cjeline slične po određenim svojstvima npr. grupiranje velikog broja kutija po veličini u manje grupe. Nakon grupiranja može se odabrati bilo koja manja grupa te se nad svim elementima u manjoj grupi mogu primijeniti iste metode analize. Osim što grupiranje podataka otkriva sličnosti među elementima koji su se grupirali, također dobivamo i važne statističke zaključke, npr: koliko grupa postoji, koliko članova ima svaka grupa, kakva su njihova svojstva i slično.

4.1. Metode za grupiranje

U postupku modeliranja rješenja za problem grupiranja koristi se više različitih metoda. Dvije metode su se pokazala kao općenitije i mogu poslužiti kao predložak za rješavanje većeg broja problema grupiranja: radi se o hijerarhijskim i particijskim metodama grupiranja [16].

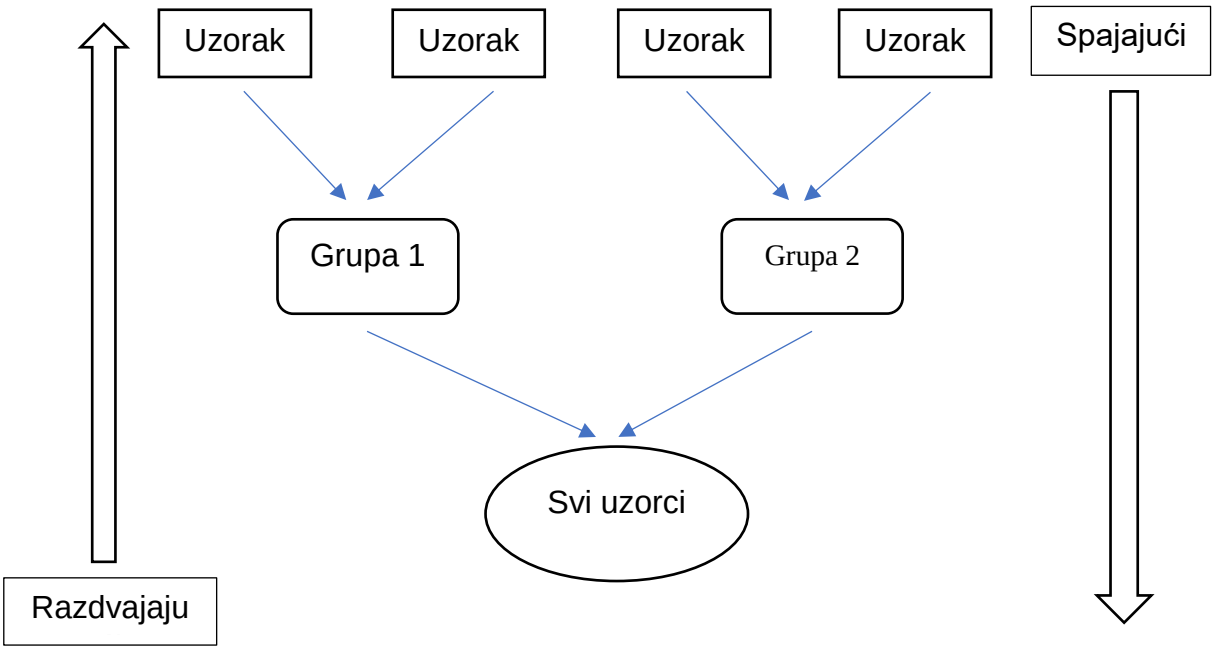
Postoji još jedna podjela algoritama za grupiranje: algoritmi sa spajajućim postupkom i oni sa razdvajajućim postupkom. Ta je podjela vezana uz samu strukturu i način rada algoritma. Algoritmi sa spajajućim postupkom počinju tako da svaki uzorak stave u zasebnu grupu, a zatim postupno spajaju grupe sve dok se ne zadovolji uvjet zaustavljanja. Algoritmi sa razdvajajućim postupkom počinju tako da sve uzorke grupiraju u jednu grupu a zatim tu grupu dijele sve dok se ne zadovolji uvjet zaustavljanja. Ova podjela jednako vrijedi i za hijerarhijske i za particijske algoritme [17].

4.1.1 Hijerarhijsko grupiranje

Kod hijerarhijskog grupiranja podaci nisu unaprijed smješteni u grupe nego postoje dva načina inicijalne raspodjele podataka:

- Svaki uzorak je grupa
- Svi uzorci se nalaze unutar jedne grupe

Postupak grupiranja može se odvijati u dva smjera ovisno o inicijalnoj raspodjeli podataka. Kada je svaki uzorak nova grupa tada se postupak grupiranja nastavlja spajanjem uzoraka s bliskim svojstvima (spajajući postupak). U svakom sljedećem koraku grupiranje se ponavlja te se grupe sa sličnim svojstvima spajaju u jednu novu grupu. Ako se svi uzorci nalaze unutar jedne grupe onda se u svakom sljedećem koraku radi razdvajanje koje se primjenjuje puno rjeđe nego postupak spajanja. Razdvajanje se radi sve dok ne dobijemo željeni broj grupa ili željenu točnost. Navedeni postupak se može prikazati kao stablo gdje su vrhovi grupe, a veze povezuju grupe koje su nastale ili su spojene iz vrhova djece. Tako kreirani graf kao struktura stabla se naziva dendogram. Na grafu grafu 6. dendogramom je prikazan postupak grupiranja.



Graf 6: Hijerarhijski proces grupiranja, [Izradio autor]

Spajajući postupak hijerarhijskog procesa grupiranja se može promatrati pomoću matrice udaljenosti. Sve jedinice u matrici su u grupama veličine jedan, a zatim se vrši spajanje u veće grupe koje su blizu jedna druge. Postoji više načina kako vršiti grupiranje, a jedan od jednostavnijih načina je metodom najbližeg susjeda.

Tablica 6: Udaljenosti između jedinica promatranja

	A	B	C	D	E
A	-				
B	2	-			
C	3	5	-		
D	6	4	3	-	
E	8	10	5	2	-

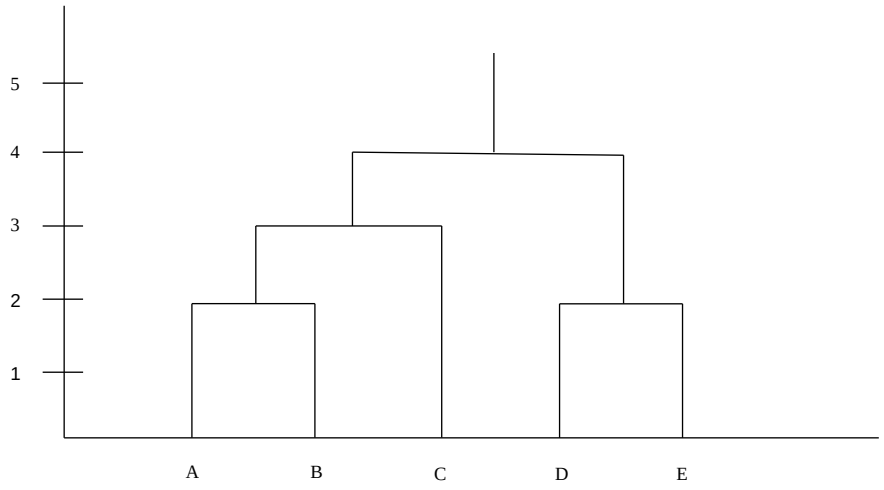
Izvor: [Izradio autor]

Tablica 7: Grupiranje na osnovi udaljenost najbližih susjeda

Udaljenost	Grupe
0	A,B,C,D,E
2	(A,B),C,(D,E)
3	(A,B,C),(D,E)
4	(A,B,C,D,E)

Izvor:[Izradio autor]

U tablici 6 su date udaljenosti između ukupno pet uzoraka (A,B,C,D,E) promatranja. Kasnije na temelju tih udaljenosti u tablici 7 radimo grupiranje na način da se dva uzorka, odnosno dvije grupe, spajaju u jednu ukoliko je određeni uzorak promatranja iz jedne grupe najbliži određenom uzorku iz druge grupe. U prvom koraku se definiraju grupe što je u ovom slučaju ukupno pet grupa. Zatim se u sljedećem koraku uzorak A i B spajaju u jednu grupu kao i uzorak D i E. U trećem koraku se uzorak C pridodaje grupi u kojoj se nalaze uzorci A i B da bi se u zadnjem koraku na temelju udaljenosti koja iznosi 4 za uzorak B i D svi uzorci grupirali u jednu zajedničku grupu. Graf 7. prikazuje dendogram u kojem se vidi kako je vršeno grupiranje po udaljenosti najbližih susjeda.



Graf 7: Grupiranje najbližih susjeda, [Izradio autor]

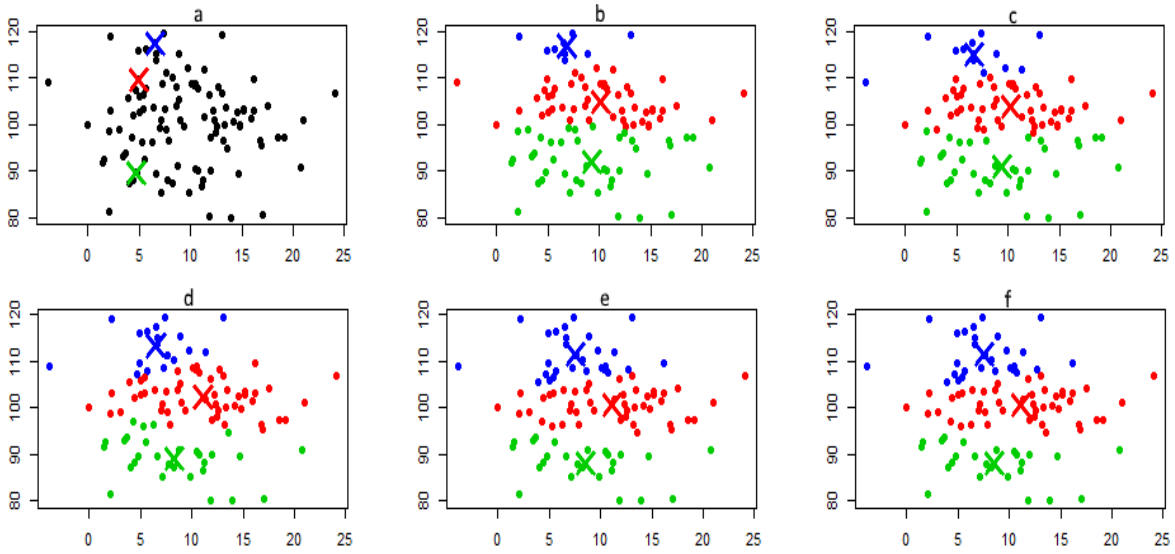
Ova metoda nije pogodna za velike skupove podataka. Vremenska složenost joj je O($n^2 \log n$), dok je prostorna O(n^2). Ukoliko se promotri postupak grupiranja tada je jasno da ako neki uzorak pripada nekoj grupi ne postoji mogućnost da prijeđe u neku drugu grupu unutar istog nivoa hijerarhije [18].

4.1.2 Particijsko grupiranje

Kod particijskog grupiranja kao rezultat dobivamo podjelu jednog uzoraka u više grupa. Particijsko grupiranje se koristi kod velikih skupova podataka zato jer bi korištenje hijerarhijskog grupiranja nad skupom velikih podataka računski bilo neizvedivo. Kao najveći problem particijskog grupiranja smatra se određivanje broja željenih grupa. Grupe kod particijskog grupiranja se stvaraju na način da se optimizira kriterijska funkcija. Kako pretraživanje skupa svih podjela na grupe radi pronalaska optimalne vrijednosti računski nije izvedivo, u praksi se algoritmi za particijsko grupiranje pokreću nekoliko puta sa različitim početnim stanjima te se najbolja dobivena konfiguracija grupa koristi kao rezultat grupiranja. Najjednostavniji i najpoznatiji algoritam grupiranja jest algoritam k-srednjih vrijednosti (k-means algorithm) [18].

4.2. Algoritam k-srednjih vrijednosti

Algoritam k-srednjih vrijednosti spada u algoritme particijskog grupiranja. Radi na način da se prvo nasumično definira k uzoraka tako da svaki bude u posebnoj grupi. Svaki taj uzorak predstavlja centroid pojedine grupe. Za svaki preostali uzorak p pronalazi se centroid prema kojem je uzorak p najbliži te se dodaje uzorak p u grupu tog centroida. Zatim se rekalkulira položaj centroida te grupe uz uzorak p te se radnja ponavlja dok podaci ne konvergiraju. [18]



Slika 7: Proces algoritma k-sredina, [19]

Na slici 7 je prikazan postupak grupiranja korištenjem algoritma k-sredina. Na slici a) su prikazana tri nasumično odabrana centroida. Nakon odabira centroida, preostali uzorak se dodjeljuje pojedinom centroidu ovisno o tome koji mu je centroid od odabrana tri bliži kao što je prikazano na slici b). Na slici c) je prikazana rekalkulacija položaja centroida s obzirom na uzorke koji se nalaze u njegovoj grupi. Postupak se provodi do faze potpune konvergencije koja je prikazana na slici f). Ponovnom provedbom ovog algoritma rezultat mi vjerojatno može biti drugačiji od prikazanog na slici 7.

Grupiranje je optimizirano ukoliko je postignuta minimalna suma kvadrata udaljenost između centroida i uzoraka unutar svake grupe: arg min $\sum_{i=1}^{k}\sum_{X_J \in Si} ||x_j - \mu_i||^2$ - gdje je μ_i „sredina“ tj. centroid skupa S_I, a mjera $||x_j - \mu_i||^2$ je Euklidska distanca.

Ovaj algoritam je popularan zato jer ga je lagano implementirati i zato jer mu je vremenska ovisnost O(n), gdje je n broj uzoraka

4.3. Programsko rješenje korištenjem algoritma k-srednjih vrijednosti

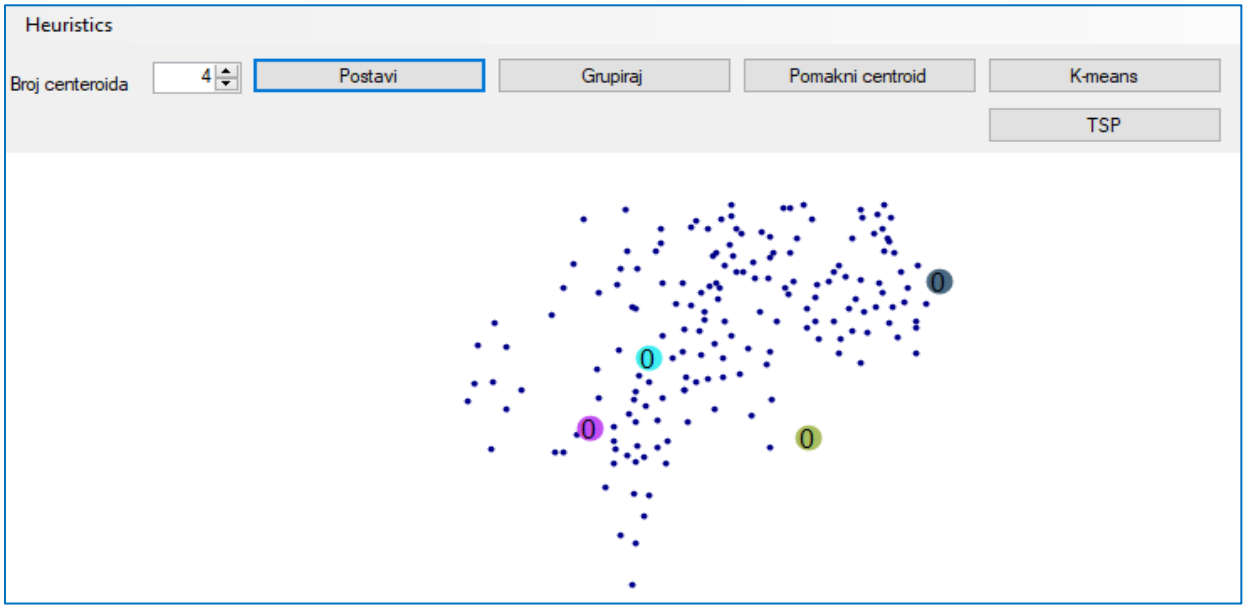
U poglavlju 2.1 objašnjeno je kako i na koji način poduzeće P3 Communications izvodi provođenje istraživanja kvalitete mobilne mreže. Npr. za provođenje kvalitete glasovnih usluga iz 2016. godine poduzeće P3 Communication koristilo je dva osobna automobila kako bi se što prije i efikasnije provela istraživanja. Isto tako moglo je biti korišteno tri, četiri ili više automobila za provođenje istraživanja a sve u cilju bržeg obavljanja zadatka. U slučaju kada se koristi više od jednog automobila za provođenje istraživanja najprije je potrebno odrediti broj vozila koja će obaviti testiranje, zatim je svakom vozilu potrebno dodijeliti lokacije koje mora posjetiti i nastaviti s dodjeljivanjem lokacija dok god nisu sve lokacije pridružene određenom vozilu ili dok se kvadratna pogreška tj. kvadratna udaljenost značajno ne smanji nakon određenog broja iteracija.

Za primjer provođenja istraživanja kvalitete mobilne mreže možemo uzeti područje Slavonije za koju je odabrano ukupno 176 mjesta sa stvarnim koordinatama. U nastavku je prikazano programsko rješenje uporabom algoritma k-srednjih vrijednosti.

4.3.1 Korištenje programskog rješenja za dodavanje centroida

Kod algoritma k-srednjih vrijednosti najprije je potrebno obaviti postupak postavljanja centroida odnosno u slučaju ispitivanja kvalitete mobilne mreže to bi bio

odabir broja vozila koja će se koristiti. Za primjer od 176 mjesta odabrat ćemo ukupno četiri vozila koji se slučajno dodjeljuju unutar prostora kao što je prikazano na slici 8. Svako vozilo je prikazano drugom bojom i brojem koji prikazuje broj lokacija koji su pridruženi pojedinom vozilu. U ovom koraku još nismo pridruživali lokacije vozilima tako da je broj za svako vozilo 0. Kada bismo ponovili postavljanje vozila odnosno centroida u prostor tada bi i njihov položaj u prostoru bio nešto drugačiji.



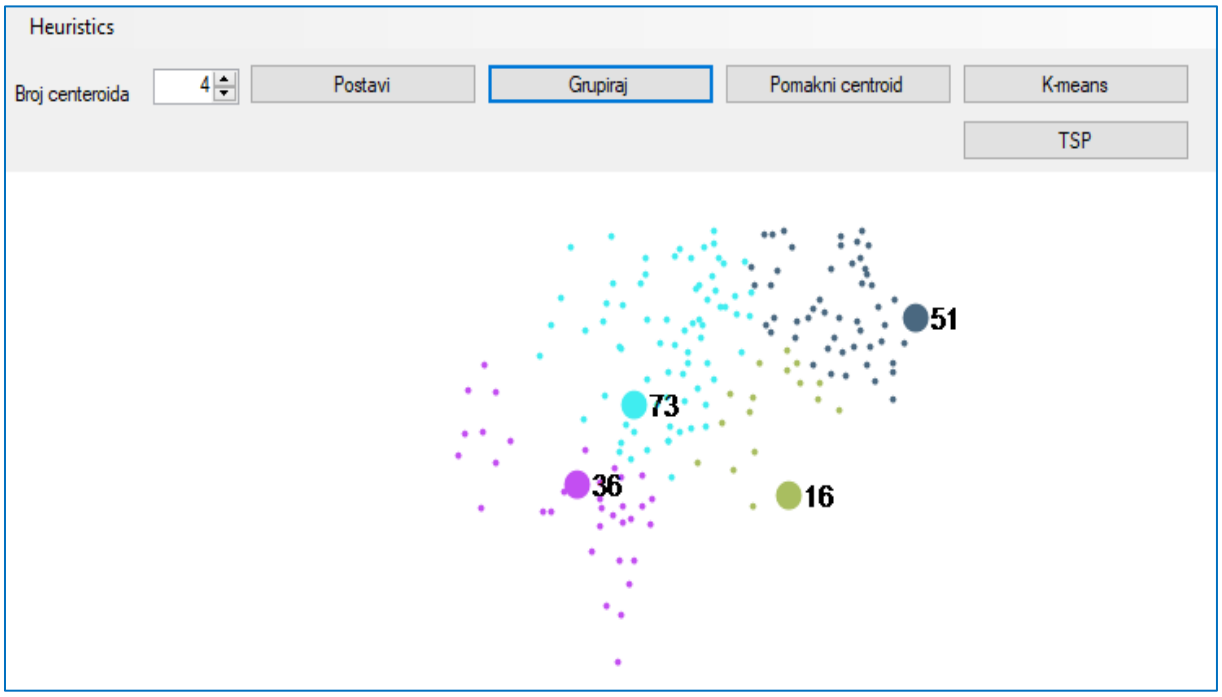
Slika 8: Postavljanje centroida korištenjem programskog rješenja, [Izradio autor]

Graf 8: Dijagram toka dodavanja centroida, [Izradio autor]

Na grafu 8. možemo vidjeti postupak koji se vrši u algoritmu korištenjem alata Visual Studio 2015, uporabom C# programskog jezika korištenjem Windows Forms tehnologije. U prvom koraku slijedi kreiranje centroida na način da se od korisnika traži da odabere željeni broj centroida odnosno u ovom slučaju vozila. Nakon kreiranja centroida oni se spremaju u privremenu memoriju i čekaju na izvršenje sljedećeg koraka. Svakom centroidu se slučajno dodjeljuje koordinata kao i boja kako bismo ih mogli razlikovati. Na kraju se centroidi pozicioniraju u prostor s obzirom na prije odabrani broj centroida, koordinatu i boju.

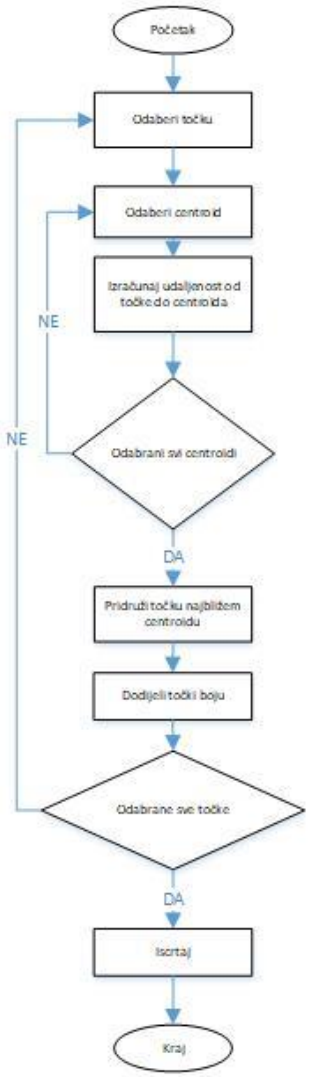
4.3.2. Korištenje programskog rješenja za grupiranje

Nakon obavljenog postupka postavljanja centroida, sljedeći korak je grupiranje. Grupiranje se odvija s obzirom na broj prethodno odabranih centroida i udaljenosti lokacija do svakog pojedinog centroida.



Slika 9: Grupiranje lokacija korištenjem programskog rješenja, [Izradio autor]

Na slici 9. možemo vidjeti da su lokacije pridružene prethodno odabranim centroidima odnosno vozilima. Tako u ovom slučaju imamo ukupno četiri grupe (centroidi sa pridruženim lokacijama) koje možemo razlikovati po boji. Prva grupa je označena ljubičastom bojom i ona predstavlja prvo vozilo kojem je trenutno dodijeljeno 36 lokacija. Druga grupa je označena svijetlo plavom bojom i ona predstavlja drugo vozilo kojem je trenutno dodijeljeno 73 lokacija. Treća grupa je označena zelenom bojom i ona predstavlja treće vozilo kojem je trenutno dodijeljeno 16 lokacija. Četvrta grupa je označena tamno plavom bojom i ona predstavlja četvrto vozilo kojem je trenutno dodijeljeno 51 lokacija.

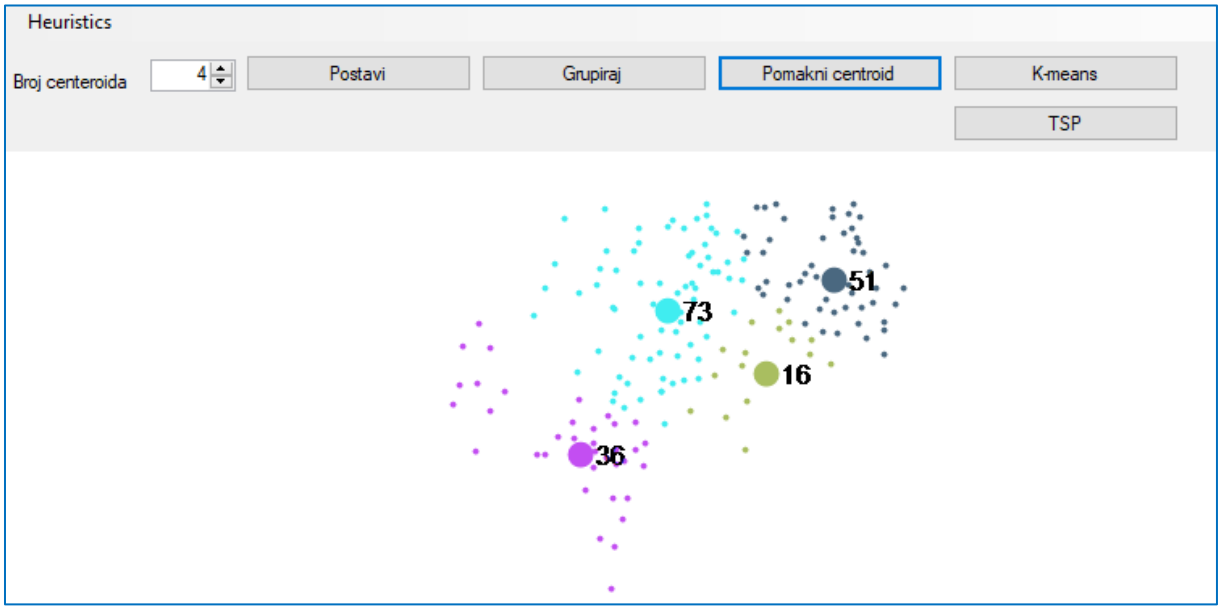


Graf 9: Dijagram toka grupiranja, [Izradio autor]

Kao i kod grafa 8. u kojem se vidi postupak dodavanja centroida, korištenjem istih alata i tehnologija algoritam za grupiranje je objašnjen u grafu 9. Algoritam od ukupno 176 lokacija uzima prvu lokaciju po redu iz prostora i za njega računa udaljenost do svakog pojedinog centroida. Nakon što je izračunata udaljenost, lokacija se pridružuje svom najbližem centroidu te mu se dodjeljuje boja jednaka boji centroida kojem je pridruženo. U tom trenutku centroid i dodijeljena mu lokacija predstavljaju jednu grupu. Isti postupak treba ponoviti i za sve preostale točke odnosno lokacije i pridružiti ih najbližem centroidu. Nakon završenog algoritma imamo ukupno četiri grupe sa pridruženim im lokacijama kao što je već objašnjeno na slici 9.

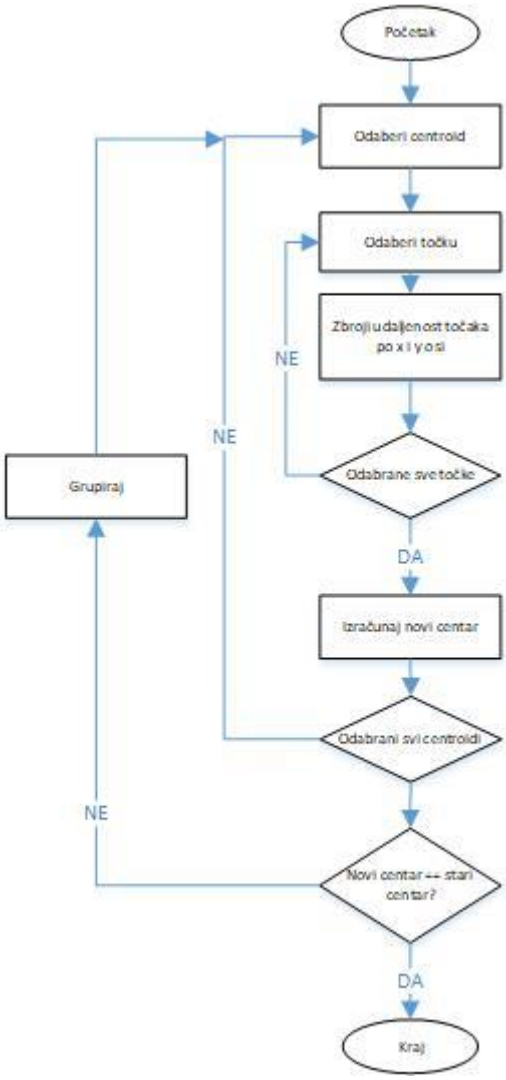
4.3.3. Pomicanje centroida korištenjem programskog rješenja

Nakon obavljenog postupka grupiranja, sljedeći korak je korak pomicanja centroida unutar dodijeljene grupe. Ovaj korak se izvodi kako bi se dobila što optimalnija raspodjela točaka po centroidima, a to konkretno u ovom slučaju znači što optimalnija raspodjela udaljenosti koje svako vozilo mora proći. Kao što je vidljivo sa slike 9. određeni centroidi odnosno vozila se ne nalaze u centru svoje grupe. Ukoliko bi algoritam završio svoje izvođenje bez ovog koraka tada bismo imali neravnomjernu raspodjelu područja po vozilima. Na slici 10. se može vidjeti kako se svaki centroid pomaknuo u smjeru središta svoje grupe.



Slika 10: Pomicanje centroida korištenjem programskog rješenja, [Izradio autor]

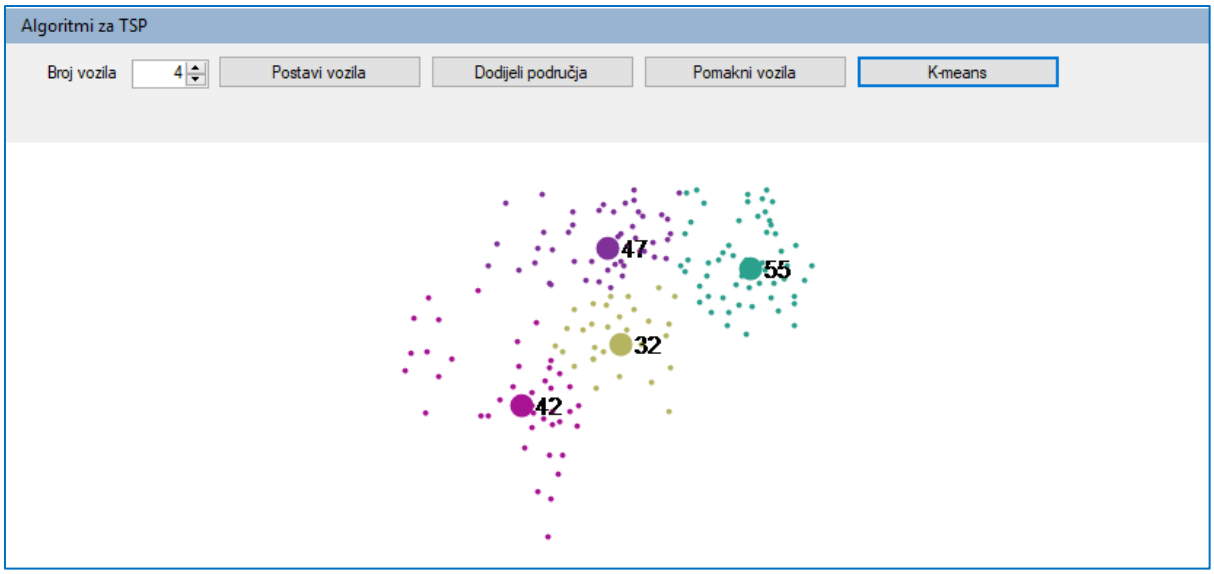
Ukoliko se grafički usporede rješenja dobivena sa slike 9 i slike 10. vidljivo je da nakon postupka pomicanja centroida, svaki centroid odnosno vozilo i dalje ima jednaki broj pridruženih lokacija. Razlog tome je što se algoritam za grupiranje mora ponovo izvršiti kako bi se s obzirom na trenutnu raspodjelu centroida u prostoru ponovo izvršilo računanje udaljenosti svake točke do centroida. Algoritam za grupiranje je već objašnjen u grafu 9. Na grafu 10. možemo vidjeti kako radi algoritam za pomicanje centroida.



Graf 10: Dijagram toka pomicanja centroida, [Izradio autor]

Za svaki centroid se za svaku njegovu pridruženu točku zbrajaju udaljenosti do x i y osi. Nakon toga se računa novi centar za svaki centroid na principu aritmetičke sredine. Ukupan zbroj po x osi za sve točke se dijeli sa ukupnim brojem točaka tog centroida i isti postupak se ponavlja za y os. Za novi centar se uzimaju novo izračunate koordinate x i y osi. Ukoliko se novi centar centroida razlikuje od starog centra izvodi se algoritam za grupiranje koji je objašnjen u grafu 9. Algoritam prestaje sa izvršavanjem kada je svaki centroid pozicioniran u središte svoje grupe.

Konačno rješenje grupiranja je prikazano na slici 11. nakon što se algoritam za grupiranje objašnjen na grafu 9. izvršavao sve do trenutka kada više nije postojalo bolje rješenje za zadani problem. Za ovaj primjer bilo je odabrano četiri vozila, iako je bilo moguće odabrati bilo koji drugi broj vozila.



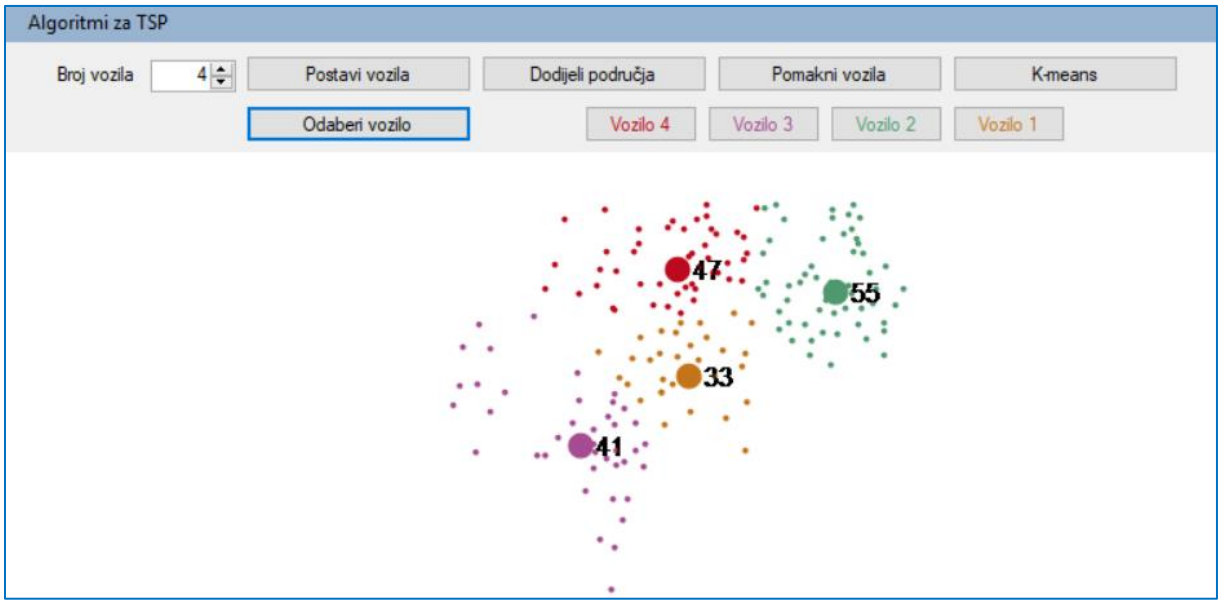
Slika 11: Završetak algoritma za grupiranje, [Izradio autor]

5. Primjena algoritama TSP-a u svrhu istraživanja kvalitete mobilne mreže

U prethodnom poglavlju je korištenjem programskog rješenja objašnjen postupak grupiranja vozila i područja odnosno dodjeljivanja svakom vozilu određen broj lokacija koje mora posjetiti. Nastavno na rezultate dobivene algoritmom za grupiranje primjenjuju se algoritmi za rješavanje problema trgovačkog putnika. Ti algoritmi će pokušati pronaći optimalnu rutu vozila kako bi povezala sve lokacije, a da pri tome prijeđena udaljenost bude što kraća. Nad svakim vozilom se posebno provode dva algoritma: algoritam najbližeg susjeda čiji je način rada opisan u poglavlju 3.2.2.1 i 2-Opt algoritam koji je jedan od najpoznatijih algoritama lokalnog pretraživanja. 2-Opt algoritam će se provesti nakon izvršavanja algoritma najbližeg susjeda u svrhu poboljšanja inicijalnog rješenja.

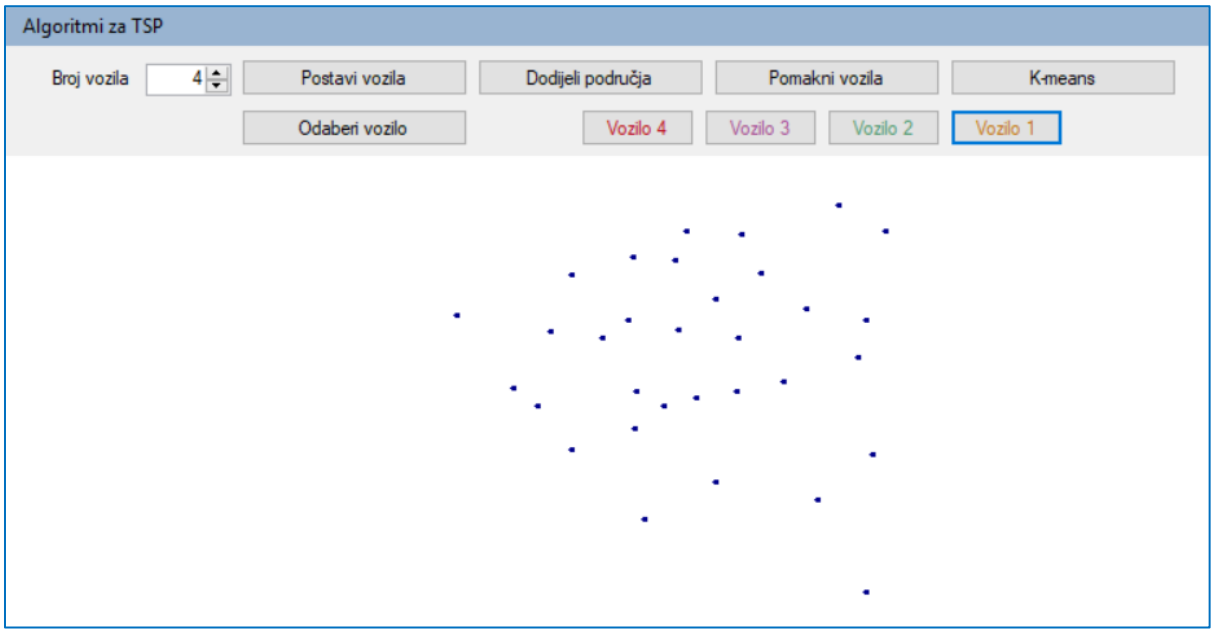
5.1. Primjena algoritma najbližeg susjeda korištenjem programskog rješenja

Nastavno na primjeru poduzeća P3 Communication za koje smo proveli grupiranje, provodi se algoritam najbližeg susjeda za svako vozilo posebno. Prije nego li algoritam najbližeg susjeda počne sa izvršavanjem, potrebno je omogućiti dohvat svakog vozila posebno kao što je prikazano na slici 12.



Slika 12: Mogućnost dohvata pojedinog vozila, [Izradio autor]

Primjer koji se obrađuje se odnosi na ukupno četiri vozila, stoga postoje i četiri različita dugmeta koja se mogu odabrati: vozilo 1, vozilo 2, vozilo 3 i vozilo 4. Odabirom bilo kojeg vozila otvorit će se skup točaka odnosno lokacija samo za odabrano vozilo kao što je prikazano na slici 13.



Slika 13: Prikaz lokacija za pojedino vozilo, [Izradio autor]

Trenutno odabrano vozilo je vozilo 1 s pripadajućim mu lokacijama. Vozilo 1 ima ukupno dodijeljeno 33 lokacije nad kojima se provodi algoritam najbližeg susjeda.

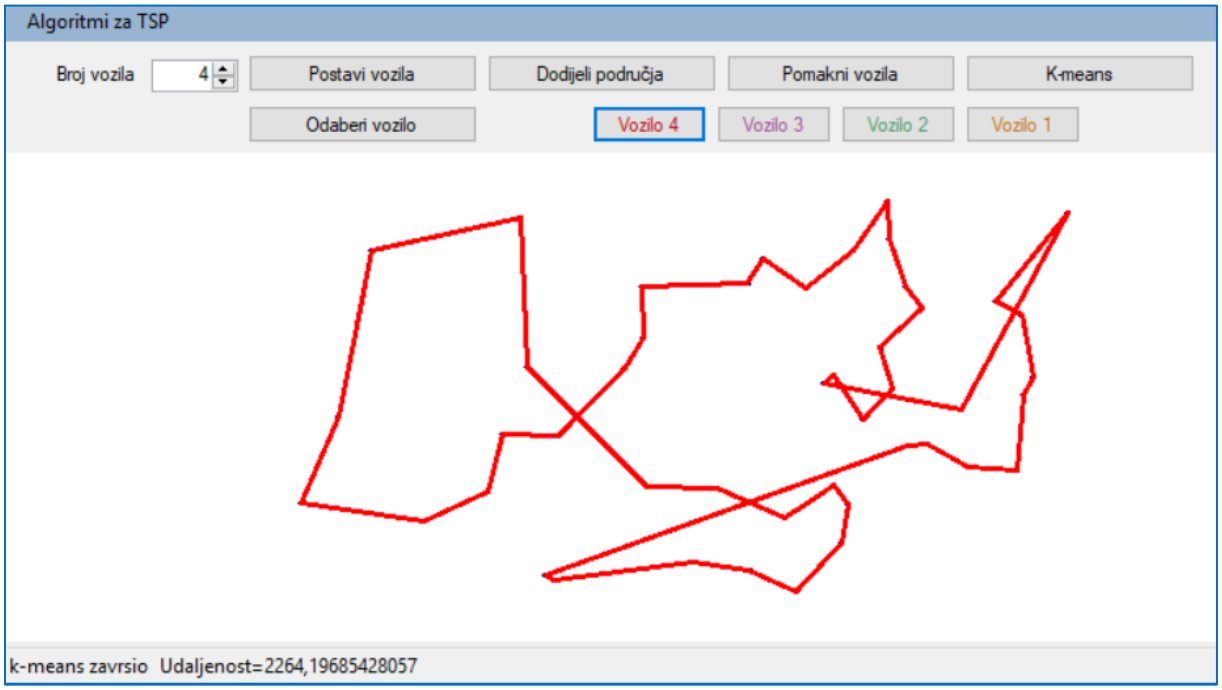
Kao što je objašnjeno u poglavlju 3.2.2.1 algoritam najbližeg susjeda uvijek traži najbliži neposjećeni vrh i ponavlja taj korak dok god postoji vrh koji nije posjećen. Rezultati izvršavanja algoritma najbližeg susjeda za vozilo 1 prikazani su na slici 14.

Slika 14: Izvršavanje algoritma najbližeg susjeda za vozilo 1, [Izradio autor]

Vozilo 1 posjetilo je sve lokacije iako je sa slike 14. zbog malog broja lokacija koje vozilo 1 treba posjetiti lako uočljivo da dobiveno rješenje nije optimalno. Kao i za vozilo 1, algoritam najbližeg susjeda provodi se i za preostala vozila a rezultati za svako pojedino vozilo su prikazani na slikama 15,16 i 17.

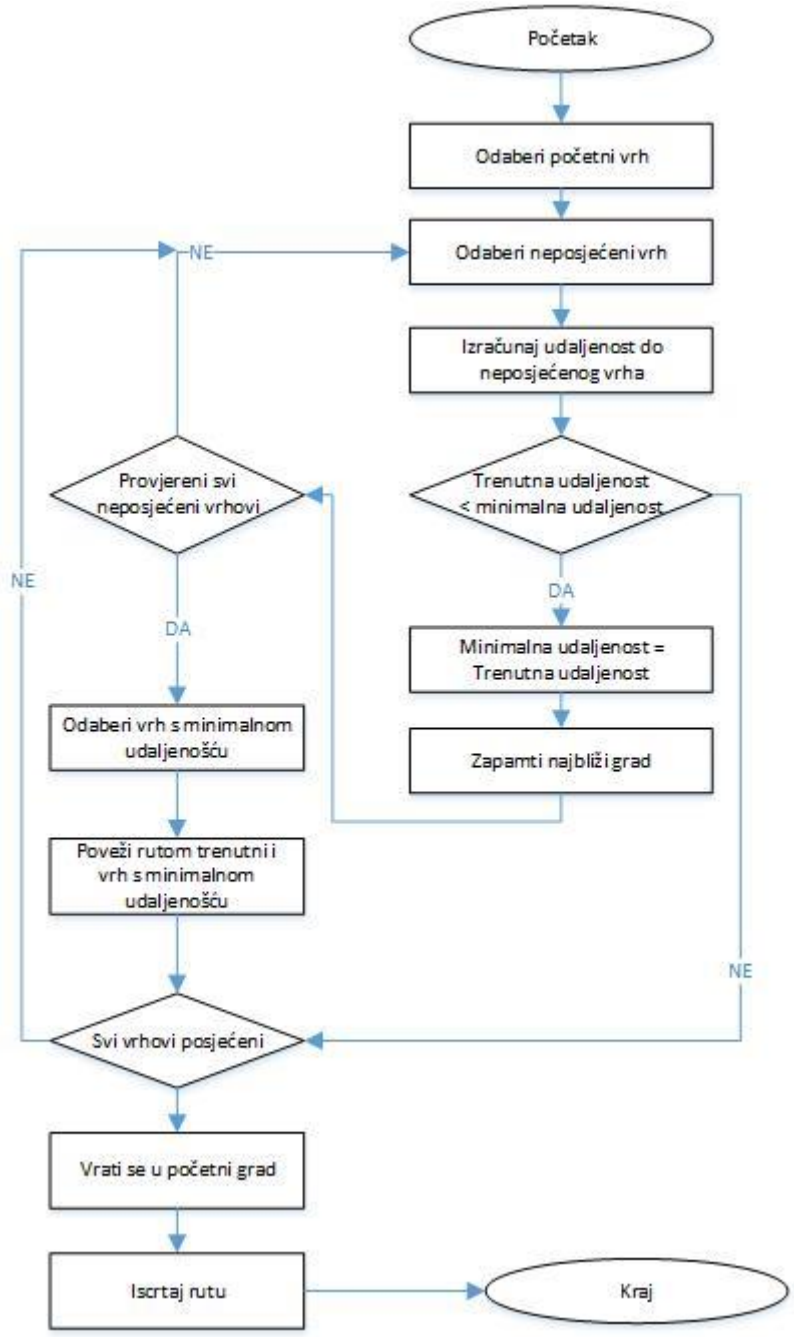
Slika 15: Izvršavanje algoritma najbližeg susjeda za vozilo 2, [Izradio autor]

Slika 16: Izvršavanje algoritma najbližeg susjeda za vozilo 3, [Izradio autor]



Slika 17: Izvršavanje algoritma najbližeg susjeda za vozilo 4, [Izradio autor]

Na grafu 11. dan je prikaz rada programskog rješenja za algoritam najbližeg susjeda.



Graf 11: Dijagram toka algoritma najbližeg susjeda, [Izradio autor]

Nakon odabranog početnog vrha sljedeći je korak nasumični odabir neposjećenog vrha. U trenutku kada algoritam sadrži početni vrh i neposjećeni vrh, računa se udaljenost dviju točaka u ravnini određenim Kartezijevim koordinatama po sljedećem izrazu:

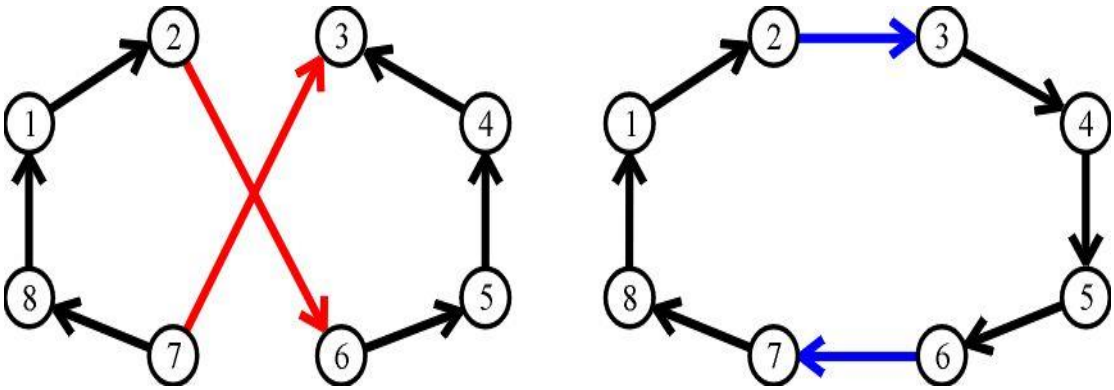
$\sqrt{(\boldsymbol{x.početni\ vrh} - \boldsymbol{x.neposjećeni\ vrh})^2 + (\boldsymbol{y.početni\ vrh} - \boldsymbol{y.neposjećeni\ vrh})^2}$

. Prilikom prvog prolaska petlje koja sadrži izračunatu udaljenost, minimalna udaljenost je prva izračunata udaljenost od početnog vrha do odabranog neposjećenog vrha.

U trenutku kada se petlja izvršila onoliko puta koliko postoji neposjećenih vrhova, dodaje se u rješenje ruta od početnog vrha do vrha minimalne udaljenosti. U sljedećem koraku se vrh s minimalnom udaljenošću postavlja kao početni vrh te se algoritam izvršava dok god postoji neposjećeni vrh za kojeg nije izračunata udaljenost. Kada se svi vrhovi nalaze u rješenju, na kraju rute se dodaje početni grad. Nakon izvršavanja algoritma iscrtava se ruta čiji se prikaz može vidjeti na slikama 14. 15. 16. i 17.

5.2. 2-Opt algoritam

2-Opt algoritam jedan je od najpoznatijih algoritama lokalnog pretraživanja koji se koristi kod problema trgovačkog putnika. Radi na principu unaprjeđenja rute iterativno, brid po brid, odnosno bira dva brida u ciklusu, uklanja ih i dodaje dva nova brida. Spajanje se vrši tako da se zadrže uvjeti obilaska, a dobiveni se obilazak dalje koristi samo ako je kraći od polaznog. Algoritam 2-Opt poboljšava trenutno rješenje sve do kada ne zaglavi u lokalnom optimumu.



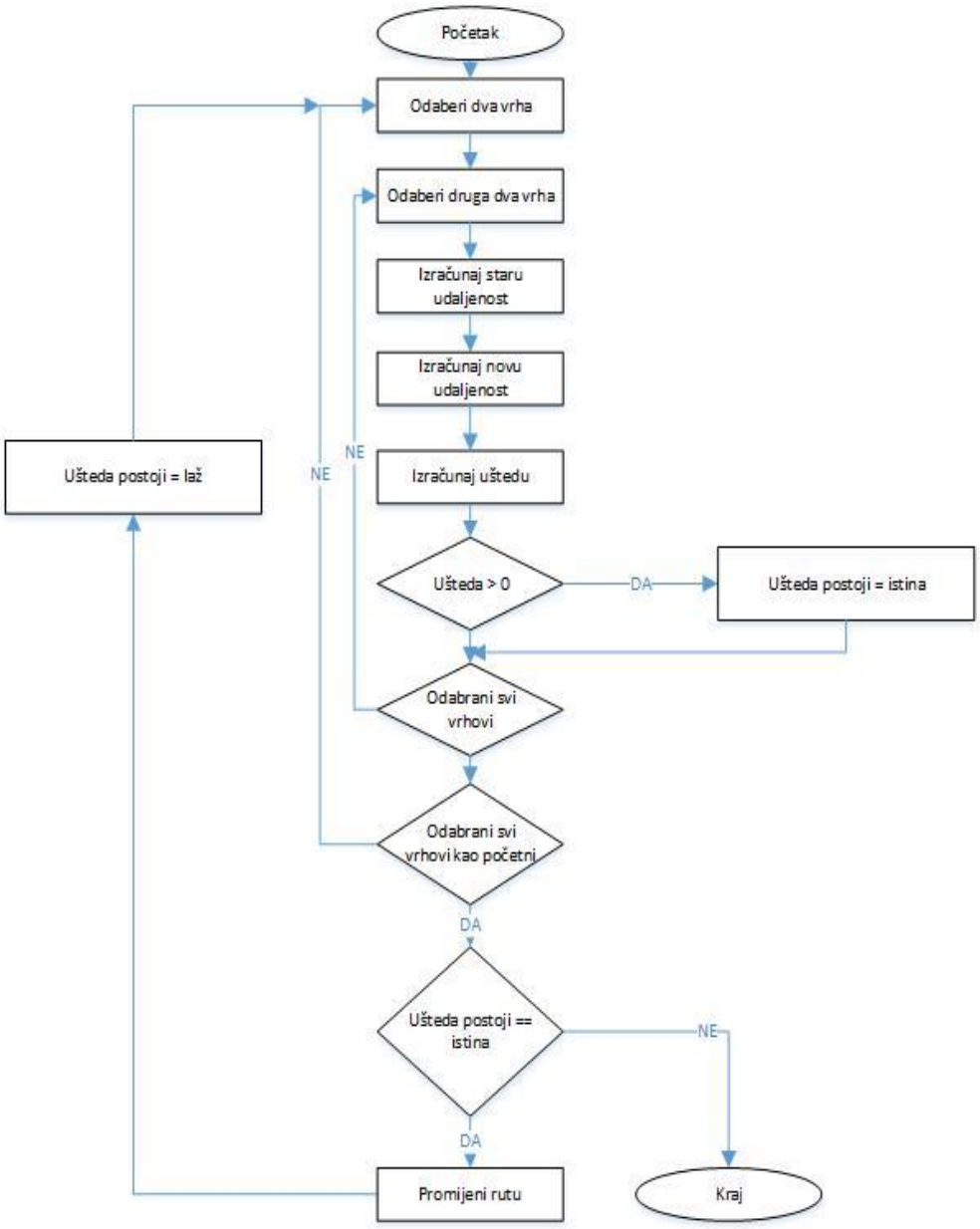
Slika 18: Primjer poboljšanja rute korištenjem 2-Opt algoritma, [20]

Slika 18. prikazuje inicijalnu rutu i njenu poboljšanu verziju. Bridovi koji spajaju vrhove 2-6 i 7-3 (označeni crvenom bojom) uklonjeni su i zamijenjeni novim bridovima 2-3 i 6-7 (označeni plavom bojom). Također, ruta koja spaja vrhove 6-5-4-3 je nakon poboljšanja korištenjem 2-Opt algoritma obrnutog redoslijeda.

5.3. Primjena 2-Opt algoritma korištenjem programskog rješenja

U poglavlju 5.1 korišten je algoritam najbližeg susjeda na primjeru poduzeća P3 Communication za ukupno četiri vozila. Algoritam 2-Opt izvršiti će se nad inicijalnim rješenjima dobivenim algoritmom najbližeg susjeda za svako vozilo posebno.

Na grafu 12. dan je prikaz rada programskog rješenje za algoritam 2-Opt.



Graf 12: Dijagram toka 2-Opt algoritma, [Izradio autor]

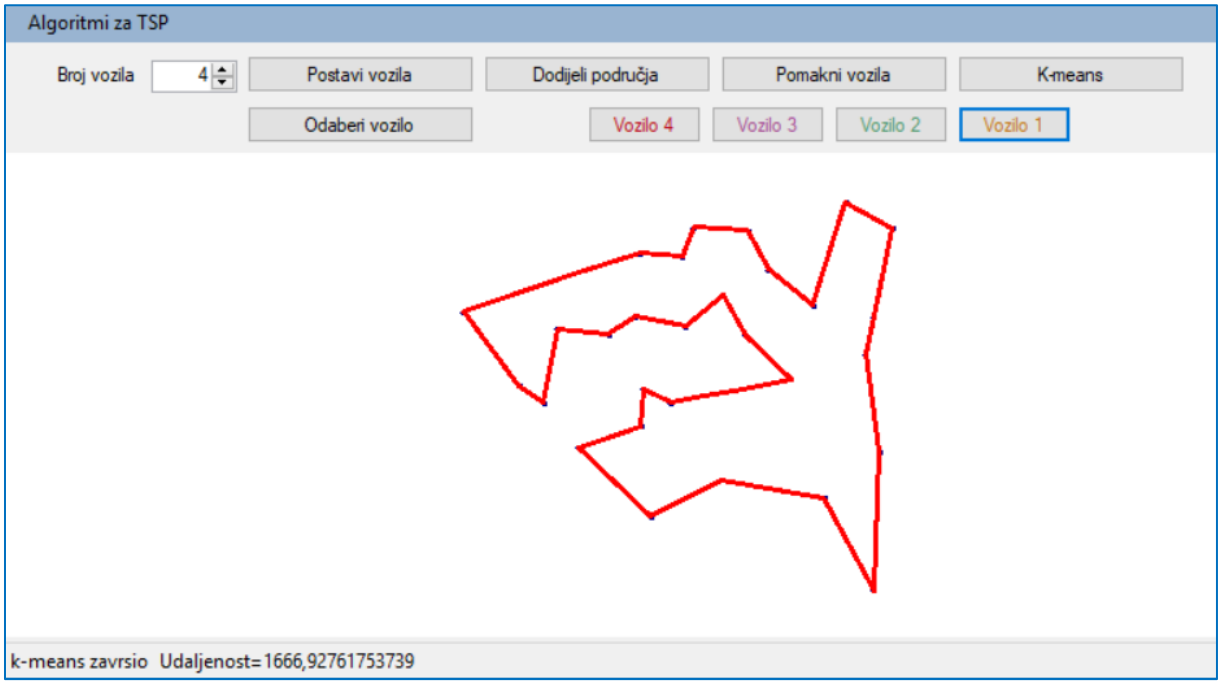
Nakon što su odabrana ukupno četiri vrha (A,B,C,D) iz rute dobivene algoritmom najbližeg susjeda, računa se udaljenost do svakog vrha posebno. Takvu udaljenost nazivamo „stara udaljenost“. Način izračuna udaljenosti dan je u poglavlju 5.1. s tim da se u ovom slučaju udaljenost računa za po svaki par vrhova posebno (dva para) i onda se udaljenosti međusobno zbroje na način: stara udaljenost = udaljenost(A->B) + udaljenost(C->D).

„Nova udaljenost“ se računa istom formulom kao i - „stara udaljenost“, ali s drugačijim redoslijedom vrhova unutar rute odnosno varijabli unutar formule na način:

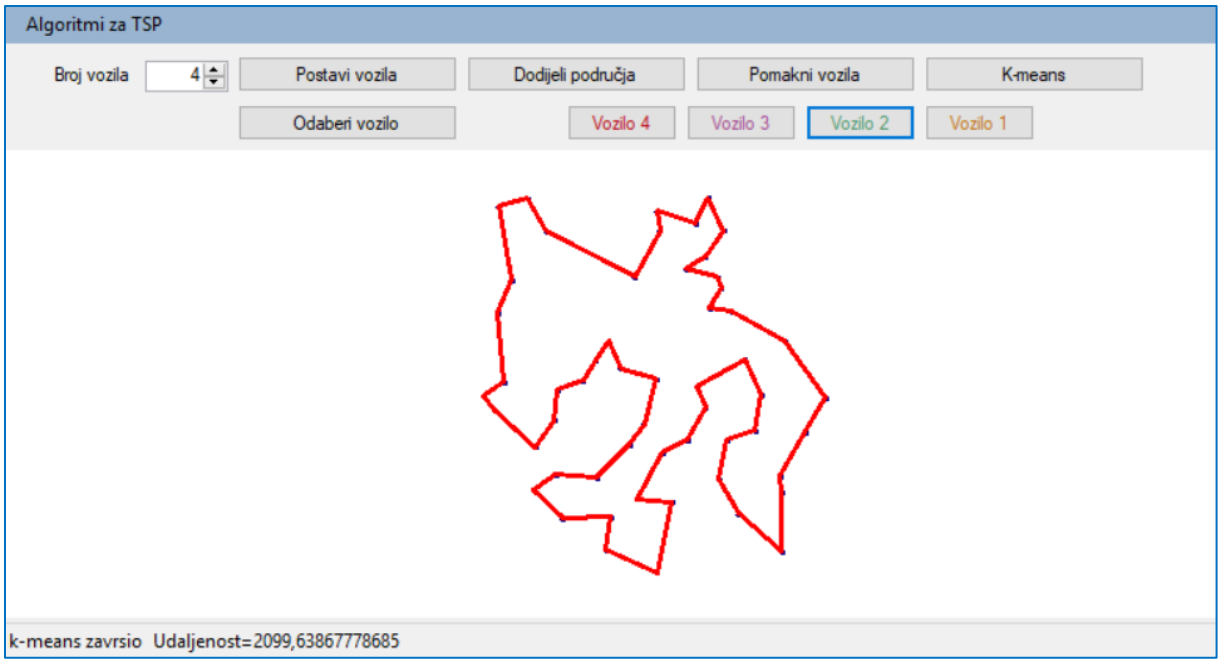
$\sqrt{(A.x - C.x)^2 + (A.y - C.y)^2}$, odnosno računa se udaljenost od prvog vrha do trećeg vrha. Isti postupak se još jednom ponavlja za rutu B->D.

Dakle nova udaljenost je: nova udaljenost = udaljenost(A->C) + udaljenost(B->D).

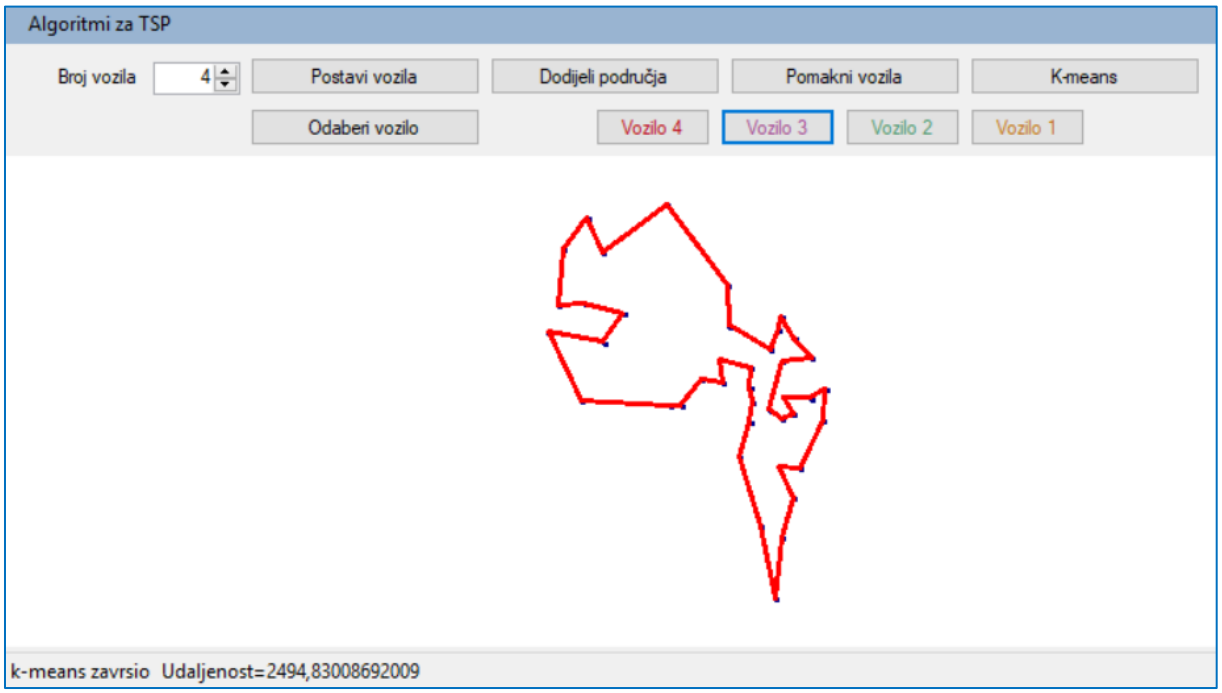
Nakon izračunatih udaljenosti računa se „ušteda“ aritmetičkom operacijom oduzimanja na način: ušteda = stara udaljenost – nova udaljenost. Ukoliko su odabrani svi vrhovi, provjerava se postoji li „ušteda“. Ukoliko „ušteda“ postoji, trenutna ruta dobivena algoritmom najbližeg susjeda se mijenja i algoritam se ponovo izvršava. Ukoliko algoritam više ne pronalazi novu rutu koja će doprinijeti dodatnoj uštedi, algoritam se prekida. Na slikama 19. 20. 21. i 22. prikazana je ruta za svako vozilo posebno nakon izvršavanja algoritma 2-Opt.



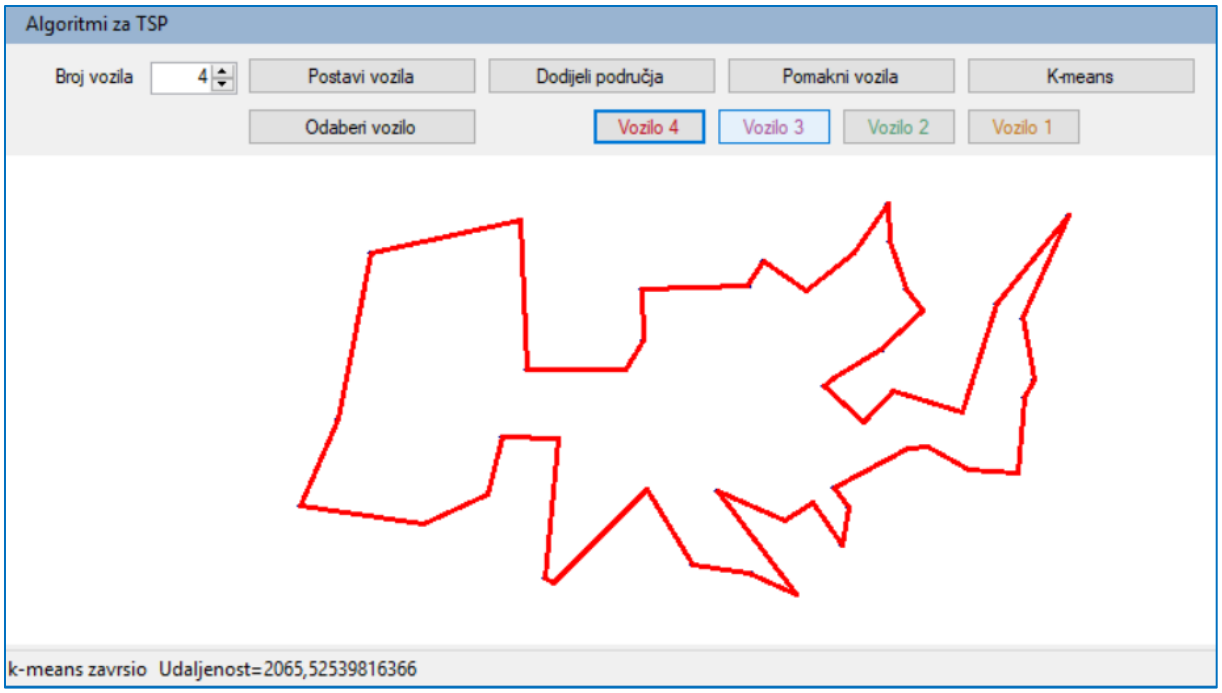
Slika 19: Izvršavanje 2-Opt algoritma za vozilo 1, [Izradio autor]



Slika 20: Izvršavanje 2-Opt algoritma za vozilo 2, [Izradio autor]



Slika 21: Izvršavanje 2-Opt algoritma za vozilo 3, [Izradio autor]



Slika 22: Izvršavanje 2-Opt algoritma za vozilo 4, [Izradio autor]

6. Analiza rezultata dobivenih algoritmima za rješavanje problema TSP-a

Algoritmi za rješavanje problema trgovačkog putnika su korišteni na primjeru poduzeća P3 Communication koje se bavi istraživanjem kvalitete mobilne mreže. P3 Communication svoje istraživanje provodi na način da vozilom prolazi kroz sva područja u kojima je potrebno ispitati kvalitetu mobilne mreže te mjeri određene parametre koji su objašnjeni u poglavljima 2.1 i 2.2. Kako je navedeni problem moguće riješiti korištenjem više vozila, najprije je bilo potrebno obaviti postupak grupiranja, odnosno svakom vozilu pridružiti područja koja treba posjetiti. Postupak grupiranja je objašnjen kroz poglavlje 4. Zatim se nad svakim područjem izvršio algoritam najbližeg susjeda. S obzirom da je algoritam najbližeg susjeda „pohlepni“ algoritam i gleda samo najbolje trenutno rješenje, dobivena ruta nije optimalna te se nad inicijalnim rješenjem dobivenim algoritmom najbližeg susjeda izvršio 2-Opt algoritam koji je poboljšao trenutno rješenje.

6.1. Analiza rezultata dobivenih algoritmom najbližeg susjeda

Algoritam najbližeg susjeda izvršio se nad ukupno 176 lokacija u Republici Hrvatskoj. Za svaku lokaciju korištena je stvarna koordinata iako nisu uzimane stvarne udaljenosti među njima. Sve lokacije se nalaze u istočnom dijelu Hrvatske odnosno u Slavoniji. Koordinata svake lokacije pretvorena je u decimalni broj tako da se svaka lokacija može iscrtati unutar sučelja omeđenog x i y koordinatnim sustavom. Kada su se sve lokacije iscrtala na sučelju, izračunala se udaljenost do svake lokacije po principu računanja udaljenosti dviju točaka u ravnini određenim Kartezijevim koordinatama. Dakle, udaljenosti među lokacijama su zapravo jedinične vrijednosti po x i y osi. Algoritam je uspoređivao rezultate udaljenosti do svake lokacije. Zatim se od polazišne lokacije do lokacije s najmanjom udaljenosti iscrtala ruta. Od lokacije s najmanjom izračunatom udaljenosti dobiveno u prethodnom koraku do svih preostalih neposjećenih lokacija računala se udaljenost i iscrtala se ruta do najbliže lokacije. Postupak računanja udaljenosti i iscrtavanja rute ponavljao se sve dok se nije dobio zatvoreni ciklus, odnosno sve dok se ruta nije ponovo vratila u lokaciju iz kojeg je krenula.

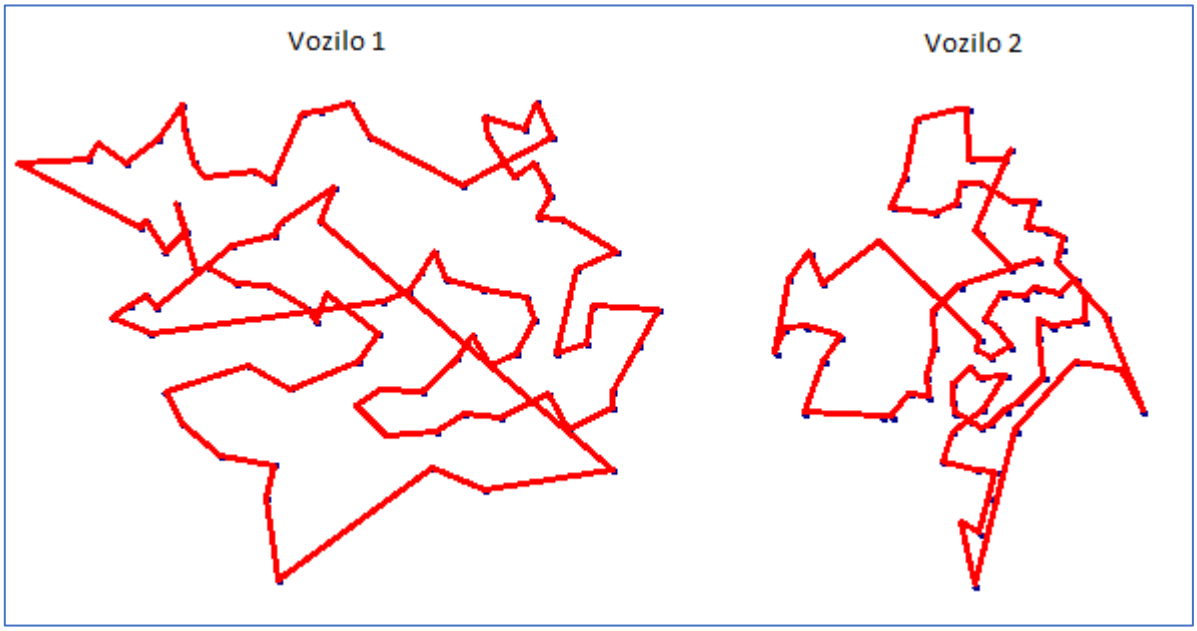
Ukupni rezultati prijeđene udaljenosti za sva četiri vozila prikazani su u tablici 8. zaokruženi na tri decimale.

	Vozilo 1	Vozilo 2	Vozilo 3	Vozilo 4	Ukupno
Prijeđena udaljenost	2093,40	2499,17	3031,65	2264,20	9888,43
Broj posjećenih lokacija	33	55	41	47	176

Tablica 8: Prijeđena udaljenost za primjer sa 4 vozila, [Izradio autor]

Na primjeru iz tablice 8. vozilo 2 posjetilo je najviše lokacija – ukupno 55 lokacija, dok je vozilo 3 prešlo najveću udaljenost – ukupno 3031,654.

Za navedeni primjer poduzeća P3 Communication mogao se koristiti i neki drugi broj vozila. Za primjer sa 2 vozila bilo bi potrebno ponovno provesti algoritam za grupiranje i rješavanje problema trgovačkog putnika. Rezultati algoritma najbližeg susjeda za primjer sa 2 vozila prikazani su na slici 23.



Slika 23: Algoritam „najbliži susjed“ za ukupno 2 vozila, [Izradio autor]

Ukupni rezultati prijeđene udaljenosti u primjeru kada su korištena dva vozila prikazani su u tablici 9.

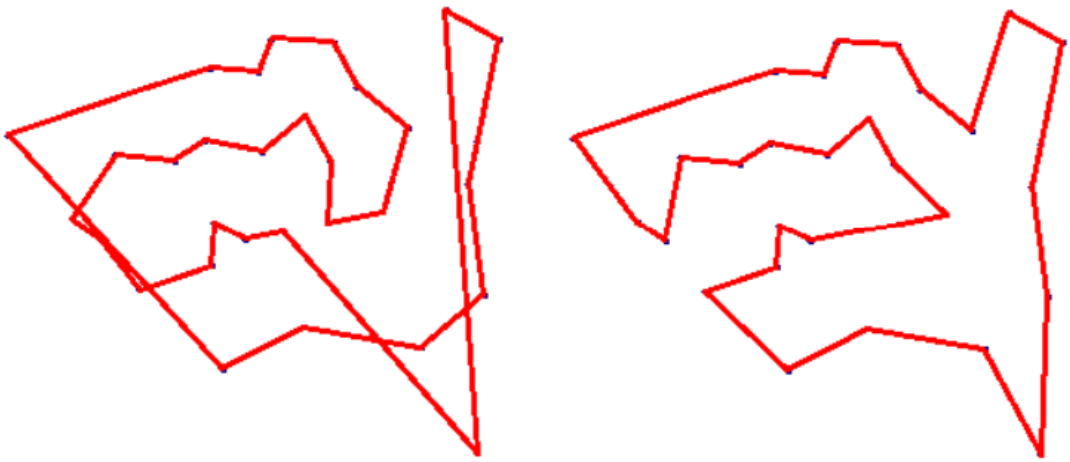
	Vozilo 1	Vozilo 2	Ukupno
Prijeđena udaljenost	4299,658	5242,410	9542,068
Broj posjećenih lokacija	92	84	176

Tablica 9: Prijeđena udaljenost za primjer sa 2 vozila, [Izradio autor]

Ukoliko usporedimo rezultate dobivene u tablicama 8. i 9., ukupna prijeđena udaljenost je manja u primjeru s dva vozila dok je broj posjećenih lokacija jednak.

6.2. Analiza rezultata dobivenih 2-Opt algoritmom

Nad inicijalnim rješenjem dobivenim algoritmom najbližeg susjeda, provodi se 2-Opt algoritam kako bi poboljšao rješenje algoritma najbližeg susjeda. Način rada 2-Opt algoritma objašnjen je na grafu 12. u poglavlju 5.3. Na slikama 24. 25. 26. i 27. prikazana su rješenja algoritma 2-Opt u usporedbi s algoritmom najbližeg susjeda.

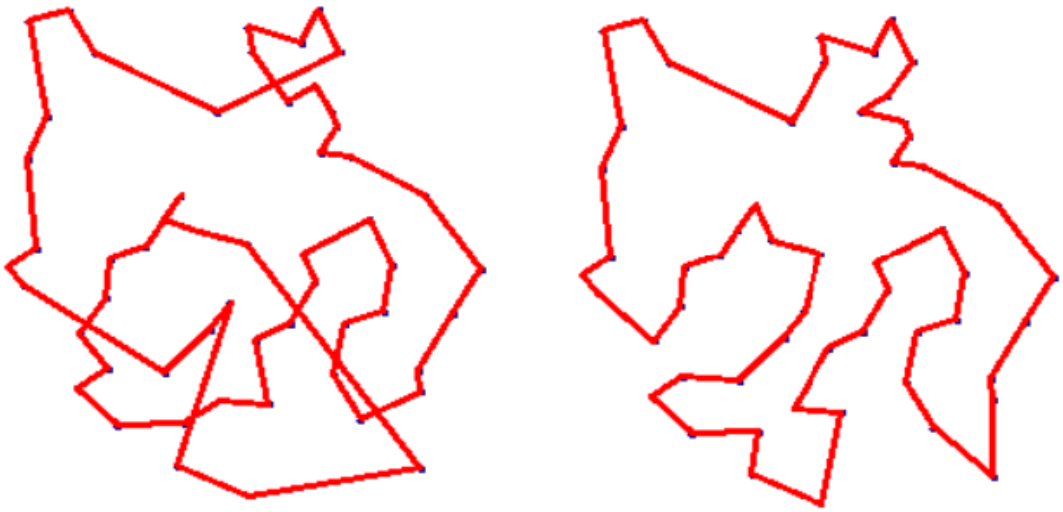


Vozilo 1 - NNH Vozilo 1 – 2-Opt

Slika 24: Poboljšano rješenje za vozilo 1 korištenjem 2-Opt algoritma, [Izradio autor]

Na slici 24. jasno se vidi da je 2-Opt algoritam poboljšao rješenje dobiveno algoritmom najbližeg susjeda, odnosno udaljenost koje vozilo treba proći je manja i iznosi 1666,928.

Slika 25. prikazuje poboljšano rješenje za vozilo 2, dok slike 26. i 27. prikazuju poboljšana rješenja za vozila 3 i 4.



Vozilo 2 - NNH Vozilo 2 – 2-Opt

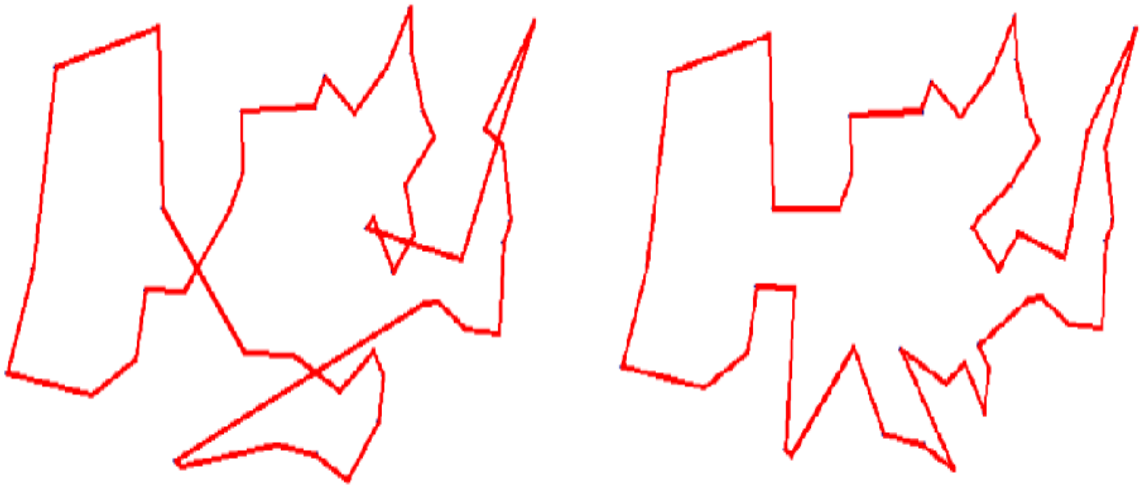
Slika 25: Poboljšano rješenje za vozilo 2 korištenjem 2-Opt algoritma, [Izradio autor]

Korištenjem 2-Opt algoritma, udaljenost koje vozilo 2 mora proći je manja u odnosu na rješenje dobiveno algoritmom najbližeg susjeda i iznosi 2099,639.

Vozilo 3 - NNH Vozilo 3 – 2-Opt

Slika 26: Poboljšano rješenje za vozilo 3 korištenjem 2-Opt algoritma, [Izradio autor]

Korištenjem 2-Opt algoritma, udaljenost koje vozilo 3 mora proći je manja u odnosu na rješenje dobiveno algoritmom najbližeg susjeda i iznosi 2494,830.



Vozilo 4 - NNH Vozilo 4 – 2-Opt

Slika 27: Poboljšano rješenje za vozilo 4 korištenjem 2-Opt algoritma, [Izradio autor]

Korištenjem 2-Opt algoritma, udaljenost koje vozilo 3 mora proći je manja u odnosu na rješenje dobiveno algoritmom najbližeg susjeda i iznosi 2065,525.

Razlike prijeđenih udaljenosti između 2-Opt algoritma i algoritma najbližeg susjeda za svako vozilo prikazane su u tablici 10.

	Vozilo 1	Vozilo 2	Vozilo 3	Vozilo 4	Ukupno
NNH	2093,405	2499,172	3031,654	2264,197	9888,428
2-Opt	1666,928	2099,639	2494,830	2065,525	8326,922
Broj posjećenih lokacija	33	55	41	47	176
Razlika	426,072	366,533	536,824	198,672	1561,506
Razlika [%]	20,35	14,66	17,70	8,77	15,79

Tablica 10: Prijeđena udaljenost za primjer s 4 vozila, [Izradio autor]

Dobivene razlike na primjeru iz tablice 10. možemo promatrati kao razlike prijeđenih udaljenosti, odnosno, na primjeru vozila 1, udaljenost koje vozilo mora proći je 20,35% manja kada se koristi 2-Opt algoritam u odnosu na algoritam najbližeg susjeda. Vozilo 4, korištenjem 2-Opt algoritma ostvarilo je najmanju uštedu u odnosu na ostala vozila te ona iznosi 8,77%.

7. ZAKLJUČAK

Kako bi se istražila i odredila kvaliteta mobilne mreže na određenom geografskom području potrebno je provesti različita testiranja na različitim lokacijama. Tvrtke koje rade takva istraživanja moraju imati osigurano vozilo, ili više njih, s kojim će obići sve lokacije na kojima je potrebno istražiti kvalitetu mobilne mreže. Nerijetko se radi o jako velikom broju lokacija kako bi se dala generalna slika kvalitete mobilne mreže nekog geografskog područja. Problemi s kojima se takve tvrtke susreću su velika financijska sredstva koja je potrebno izdvojiti za gorivo kao i veliki utrošak vremena kako bi se obišle sve lokacije ukoliko se rute ne planiraju optimalno. Korištenjem algoritama za rješavanje problema trgovačkog putnika takvi se problemi mogu riješiti.

Kada bi se za rješavanje problema s velikim brojem lokacija koristile egzaktne metode koje predstavljaju ujedno i najjednostavniji način rješavanja problema trgovačkog putnika, takvo bi rješavanje vremenski trajalo jako dugo, unatoč sve većoj procesnoj moći računala u današnjem svijetu. Isprobavanje svih permutacija dovodi do toga da takvi načini pretraživanja prostora predstavljaju faktorijelnu složenost problema. Iz tog razloga, u ovom radu koristili su se heuristički algoritmi za rješavanje problema trgovačkog putnika.

Dobiveno rješenje primjenom algoritma najbližeg susjeda, predstavlja pohlepno rješenje i rezultati dobiveni tim algoritmom u većini slučajeva ne predstavljaju dovoljno dobro rješenje. Iz tog razloga za algoritam najbližeg susjeda ne možemo reći da je pronašao optimalno rješenje. Kako bi se popravilo rješenje dobiveno algoritmom najbližeg susjeda, korišten je 2-Opt algoritam koji je ukupnu prijeđenu udaljenost za sva vozila minimizirao za 15,79%.

Osim algoritama za rješavanje problema trgovačkog putnika, objašnjeno je kako se korištenjem algoritma za grupiranje može postići zadovoljavajuća raspodjela lokacija po vozilu na primjeru poduzeća P3 Communications.

Dakle, primjenom algoritama za rješavanje problema trgovačkog putnika u kombinaciji s algoritmima za grupiranje doprinosi se optimizaciji troškova poduzeća, bilo da se radi o minimiziranju ukupne udaljenosti čime se smanjuju troškovi prijevoza ili smanjenju vremena potrebnog da se obiđu sve lokacije.

LITERATURA

[1] URL: http://www.smartphonehrvatska.com/2015/07/10/hrvatski-telekom-proglasen-najboljom-mrezom-u-hrvatskoj/ (pristupljeno: srpanj 2017.)

[2] URL: http://www.p3-networkanalytics.com/wp-content/uploads/2016/10/20160713_CERTIFICATE_Tele2_Croatia.pdf

[3] Antons, J.: Neural Correlates of Quality Perception for Compley Speech Signals, Technische Universitat, Berlin, 2014.

[4] URL: http://www.p3-networkanalytics.com/wp-content/uploads/2016/11/2015_Telekom_Croatia_Audit_Report.pdf (pristupljeno: srpanj 2017.)

[5] URL: http://www.p3-networkanalytics.com/wp-content/uploads/2016/10/20160713_CERTIFICATE_Tele2_Croatia.pdf (pristupljeno: srpanj 2017.)

[6] Applegate, D., Bixby, R., Chvatal, V., Cook, W., The Traveling Salesman Problem, Princeton University Press, New Jersey, 2006.

[7] https://www.wikitechy.com/technology/java-programming-the-knights-tour-problem/#

[8] https://alumni.usal.es/blog-g-de-grafo-encontrar-el-camino-mas-corto-es-un-problema-muy-largo/

[9] http://www.math.uwaterloo.ca

[10] Matai, R., Prakash, S., Mittal, M., Traveling Salesman Problem:

An Overview of Applications, Formulations, and Solution Approaches, In-

Tech, 2010.

[11] Carić, T., Optimizacija prometnih procesa, nastavni tekst, Fakultet prometnih znanost, Sveučilište u Zagrebu, 2014.

[12] Carić, T., Fosin, J., »Problem trgovačkog putnika P2,« Studeni 2014. Dostupno na: http://estudent.fpz.hr/Predmeti/O/Optimizacija_prometnih_procesa/Materijali/predavanje_2.pdf [Pokušaj pristupa: Kolovoz 2017.].

[13] Jelić, A., Analiza primjene algoritama problema trgovačkog putnika kod komisioniranja unutar visokoregalnog skladišta s vrlo uskim prolazima, Diplomski rad, Fakultet Strojarstva i Brodogradnje, Sveučilište u Zagrebu, 2015.

[14] Carić, T., Fosin, J., Problem trgovačkog putnika P4,« Studeni 2014. Dostupno na:http://estudent.fpz.hr/Predmeti/O/Optimizacija_prometnih_procesa/Materijali/predavanje_4.pdf [Pokušaj pristupa: Kolovoz 2017.].

[15] Salajić, I., Nikolić, J., Žoljom, M., TSP – Problem trgovačkog putnika na potpunom grafu – primjenom GA algoritma i SA, Prirodoslovno matematički fakultet, Sveučilište u Zagrebu, 2008.

[16] Jajuga, K., Sokołovski, A., Classification, Clustering, and Data Analysis, Springer, 2002.

[17] Roux, M. A comparative study of divisive hierarchical clustering algorithms, 2015.

[18] Jain, A., Murty, M., Flynn, P. “Data Clustering: A Review”, ACM Computing Surveys, Vol 31, No. 3, pp 264-323, 1999.

[19] http://www.learnbymarketing.com/methods/k-means-clustering/

[20] http://www.devx.com/dotnet/Article/33574/0/page/3

POPIS KRATICA

KPI (Key Performance Indicator) - Ključni pokazatelj uspješnosti

POLQA (Perceptual Objective Listening Quality Assessment) - Objektivna procjena kvalitete slušanja

MOS-LQ (Mean opinion score - Listening Quality) - Rezultat srednjeg mišljenja za kvalitetu slušanja

TSP (Traveling Salesman Problem) - Problem trgovačkog putnika

NP (non-polynomial) – Ne polinomno vrijeme

ACO (Ant Colony Optimization) – Optimizacije mravljom kolonijom

POPIS SLIKA

POPIS TABLICA

POPIS GRAFOVA