Simetralno razvrstavanje objekata uslužnih sustava

Lukić, Ivica

Doctoral thesis / Disertacija

2013

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:218069

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-12-27**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE J. J. STROSSMAYERA

ELEKTROTEHNIČKI FAKULTET OSIJEK

Ivica Lukić

Simetralno razvrstavanje objekata uslužnih sustava

Doktorska disertacija

Osijek, 2013.

Doktorska disertacija je izrađena na Elektrotehničkom fakultetu Osijek, Sveučilišta J. J. Strossmayera u Osijeku.

Mentor: doc.dr.sc. Ninoslav Slavek

Disertacija ima 160 stranica.

Disertacija broj: 24

Povjerenstvo za ocjenu doktorske disertacije:

1. Prof. dr. sc. Goran Martinović, Elektrotehnički fakultet

Sveučilišta J. J. Strossmayera u Osijeku

2. Doc. dr. sc. Ninoslav Slavek, Elektrotehnički fakultet

Sveučilišta J. J. Strossmayera u Osijeku

3. Prof. dr. sc. Mario Žagar, Fakultet elektronike i računarstva Sveučilišta u Zagrebu

Povjerenstvo za obranu doktorske disertacije:

1. Prof. dr. sc. Goran Martinović, Elektrotehnički fakultet

Sveučilišta J. J. Strossmayera u Osijeku

2. Doc. dr. sc. Ninoslav Slavek, Elektrotehnički fakultet

Sveučilišta J. J. Strossmayera u Osijeku

3. Prof. dr. sc. Mario Žagar, Fakultet elektronike i računarstva Sveučilišta u Zagrebu

4. Prof. dr. sc. Željko Hocenski, Elektrotehnički fakultet

Sveučilišta J. J. Strossmayera u Osijeku

3. Doc. dr. sc. Krešimir Nenadićr, Elektrotehnički fakultet

Sveučilišta J. J. Strossmayera u Osijeku

Datum obrane disertacije: 10.01.2013.

Zahvale

Zahvaljujem se doc.dr.sc. Ninoslavu Slaveku, voditelju na poslijediplomskom doktorskom studiju i mentoru pri izradi doktorske disertacije, na odličnoj suradnji i podršci u obliku savjeta, konstruktivnih kritika i prijedloga. Osim pomoći na izradi doktorske disertacije zajedno smo sudjelovali u izradi znanstvenih i stručnih radova iz područja teme doktorske disertacije, kao i iz drugih područja tehničkih znanosti.

Velika zahvala ide roditeljima Slavku i Dragici, zbog pomoći i potpore koju su mi pružali tijekom čitavog obrazovanja i života i bez kojih ništa od ovog ne bi bilo moguće. Uz roditelje, posebna zahvala djedu Ivanu koji mi je uvijek pomagao.

Zahvaljujem se supruzi Lidiji koja mi je bila potpora u izradi doktorske disertacije i koja je preuzela druge obaveze na sebe i omogućila mi vrijeme za rad.

Zahvaljujem se prof.dr.sc. Franji Joviću na velikoj pomoći pri razvoju i izradi doktorske disertacije, koji je svojim savjetima i prijedlozima usmjeravao istraživanje u pravome smjeru.

Dekanu Elektrotehničkog fakulteta, prof.dr.sc. Radoslavu Galiću, također zahvaljujem na svim oblicima podrške i pomoći tijekom rada u nastavi i znanstvenim istraživanjima.

Posebna zahvala prijatelju Mirku koji je pomogao sa svojim savjetima i s kojim sam uvijek mogao raspraviti o problemima u istraživanju i sudjelovati u znanstvenom radu.

Zahvaljujem se svom nastavnom i nenastavnom osoblju Elektrotehničkog fakulteta na podršci i pomoći koju su mi pružili, ne samo pri izradi disertacije, nego i tijekom rada u nastavi i znanstvenim istraživanjima.

Također HVALA svima onima koji su na bilo koji način pomogli u izradi doktorske disertacije i ostalim aktivnostima na fakultetu, a nisu mogli biti spomenuti u zahvali.

Popis pojmova i skraćenica

- ED – Expected Distance – očekivana udaljenost
- MBR – Minimum Bounding rectangle – minimalno pravokutno područje nesigurnosti
- NED – Number of Expected Distance Calculations per object per Iteration – broj računanja očekivanih udaljenosti po jednom objektu u svakoj iteraciji
- PDF – Probability Density Function – funkcija gustoće vjerojatnosti
- SDSA – Segmentation of Data Set Area – podjela ukupnog područja skupa objekata
- SPP – Simetralna Podjela Prostora – postupak za razvrstavanje korištenjem simetralne podjele prostora
- TED – Total Expected Distance – ukupna očekivana udaljenost
- UD – Uncertain Data – podaci koji u sebi sadrže nesigurnost

Popis slika

Popis tablica

Sadržaj

1. Uvod

Razvrstavanje je proces koji skupinu objekata raspoređuje u grupe na takav način da su objekti unutar pojedine grupe sličniji jedni drugima po nekim svojstvima podataka koji ih opisuju, nego s objektima u svim drugim grupama promatramo li ista svojstva opisa podataka. Takve grupe međusobno sličnih objekata nazivaju se grozdovi (eng. clusters). Razvrstavanje je sastavni dio rudarenja podataka (eng. data mining) i statističke analize podataka o različitim objektima. Razvrstavanje ima primjenu u brojnim područjima kao što su umjetna inteligencija, lokacijske usluge, procesiranje slika, prepoznavanje uzoraka, informacijske tehnologije, biologija, marketing i brojne druge znanstvene grane, jer uključuje važan informacijski alat to jest selekciju. Zbog mnoštva primjena razvijeni su brojni algoritmi za razvrstavanje koji koriste drugačije modele stvaranja grozdova. Algoritmi se razlikuju u svojem pristupu razvrstavanju i načinima na koje tvore grozdove. Upotrebljavaju se različiti parametri selekcije, kao što su korištena funkcija udaljenosti između objekata, ukupan broj grozdova i prag gustoće (eng. density threshold). Parametri su ovisni o skupu objekata koje treba razvrstati i mijenjaju se za svaki novi skup objekata.

U ovoj disertaciji pod objektima smatramo dvodimenzionalne prostorne podatke i sličnost među objektima određuje se pomoću funkcije udaljenosti. Što je udaljenost među objektima manja oni su međusobno sličniji. Objekti se razvrstavaju tako da imaju najmanju udaljenost od središta grozda, iz čega slijedi da su objekti iz iste grupe blizu središtu grozda i blizu jedni drugima. Razvrstavanje je iterativni postupak koji se ponavlja sve dok rješenje ne konvergira, to jest dok se ne zadovolji ciljna funkcija. Ciljna funkcija može biti minimalni pomak središta grozdova u sljedećoj iteraciji. To je minimalna udaljenost za koju se mogu pomaknuti središta grozdova u odnosu na središta u prethodnoj iteraciji. Isto tako uvjet konvergiranja može biti da objekti prestaju prelaziti iz jednog grozda u drugi. Kad se zadovolji ciljna funkcija razvrstavanje je završeno. Često je potrebno mijenjati ciljnu funkciju ili broj grozdova dok se ne dobije zadovoljavajuće rješenje. Zbog velikog broja iteracija razvrstavanje je dugotrajan proces, a dodatan problem može predstavljati i nesigurnost objekata. Nesigurnost objekata može nastati zbog nepreciznosti mjerenja, šuma pri mjerenju ili zbog periodičnog nadopunjavanja podataka o objektima. Ova disertacija posvećena je razvrstavanju objekata koji sadrže nesigurnost u svojim prostornim podacima. U većini primjena prostorni podaci se periodički nadopunjuju pa se nakon nekog vremena nova pozicija objekta mora predvidjeti na osnovi starih podataka uz određenu nesigurnost. Objekti koji sadrže nesigurnost nisu

predstavljeni kao točka u prostoru, nego kao minimalno pravokutno područje nesigurnosti (eng. Minimum Bounding Rectangle - MBR) u kojem se mora nalaziti objekt. Vjerojatnost da se objekt nalazi izvan minimalnog područja nesigurnosti je nula. MBR se obično predstavlja pomoću funkcije gustoće vjerojatnosti (eng. Probability Density Function - PDF). Funkcija gustoće vjerojatnosti predstavljena je tako da se MBR podijeli u niz uzoraka. Svakome uzorku dodijeljena je vjerojatnost da se na njemu nalazi objekt, a zbroj vjerojatnosti svih uzoraka unutar MBR-a mora biti jednak 1.

Računanje očekivane udaljenosti (eng. Expected Distance - ED) između središta grozda i nesigurnog objekta puno je zahtjevnije nego jednostavno računanje udaljenosti kod objekata koji ne sadrže nesigurnost. Očekivana udaljenost računa se kao integral svih umnožaka vjerojatnosti da se objekt nalazi na nekome od uzoraka i udaljenosti tog uzorka od središta grozda. Zbog velikog broja uzoraka koji su potrebni za predstavljanje funkcije gustoće vjerojatnosti računanje očekivane udaljenosti može biti i po nekoliko tisuća puta duže nego računanje jednostavne udaljenosti. Posljedično, razvrstavanje nesigurnih objekata može biti i po nekoliko tisuća puta dugotrajnije nego razvrstavanje objekata koji ne sadrže nesigurnost. Slijedi da je pri razvrstavanju nesigurnih objekata računanje očekivane udaljenosti vremenski najzahtjevnija računalna operacija. Kako bi se izbjeglo računanje očekivane udaljenosti razvijeni su postupci za razvrstavanje nesigurnih objekata koji odbacuju grozdove kao kandidate za pojedini objekt bez računanja očekivane udaljenosti.

U ovoj disertaciji predstavljen je postupak razvrstavanja prostornih podataka, zasnovan na simetralnoj podjeli prostora te usporedbi i odbacivanju grozdova, koji znatno smanjuje broj računanja očekivanih udaljenosti [1]. Pomoću simetrale koja prolazi središtem dužine koja spaja dva grozda dvodimenzionalni prostor je podijeljen na dvije ravnine i na osnovi te podjele odbacuje se jedan od grozdova kao kandidat za promatrani nesigurni objekt. U svakoj ravnini nalazi se jedan od dva promatrana grozda. Postupak koristi svojstvo da ako se neka točka nalazi u jednoj od ravnina, onda je promatrana točka sigurno bliža grozdu koji se nalazi u istoj ravnini, nego grozdu u drugoj ravnini. Stoga grozd iz druge ravnine može biti odbačen kao kandidat za promatrani objekt bez računanja očekivane udaljenosti. Postupak koristi projekciju promatrane točke i projekcije grozdova na simetralu koja dijeli prostor na dvije ravnine. Uspoređuju se vrijednosti koordinata projekcija na simetralu i vrijednosti originalnih koordinata točke i grozdova. Ako su točka i grozd u istoj ravnini mogu se dogoditi dva različita slučaja. U prvom slučaju y koordinate projekcije biti će veće od originalnih koordinata, dok će u drugome slučaju one biti manje. Ako je jedan od dva uvjeta zadovoljen

točka i grozd su u istoj ravnini, a drugi grozd se odbacuje. Prednost postupka je u tome što se odbačeni grozd ne uspoređuje s preostalim grozdovima pa se smanjuje složenost algoritma. Daljnju usporedbu nastavlja samo onaj grozd koji se nalazi s iste strane simetrale kao promatrani objekt. Nakon što se obave usporedbe s preostalim grozdovima, većina grozdova će biti odbačena kao potencijalni kandidati za promatrani objekt i za njih se ne mora računati očekivana udaljenost. U disertaciji su provedeni pokusi pomoću kojih se pokazalo kako predloženi postupak učinkovitije odbacuje grozdove nego postojeći postupci. Pod učinkovitosti se misli da je postupak temeljen na simetralnoj podjeli prostora odbacio više grozdova od postojećih postupaka ili je sam postupak odbacivanja grozdova brži nego kod postojećih postupaka. Detaljno objašnjenje postupka napravljeno je u četvrtom poglavlju.

U ovoj disertaciji predstavljen je i postupak podjele područja skupa objekata određivanjem prostornih odnosa objekata s ciljem povećanja mogućnosti paralelne obrade [2]. Postupak dijeli područje skupa objekata, to jest skup objekata u dvodimenzionalnom prostoru, na konačan broj malih pravokutnih područja čiji je broj ovisan o broju i rasporedu objekata i grozdova u prostoru. Svako pravokutno područje razvrstava se odvojeno i promatraju se samo objekti unutar pravokutnog područja. Budući da se promatraju samo objekti u jednom području nije nužno promatrati sve grozdove, nego samo grozdove iz promatranog područja i susjednih područja. Takva područja nazvana su unutarnja pravokutna područja. Ukupno područje skupa objekata ispravno je podijeljeno na konačan broj malih pravokutnih područja ako vrijedi sljedeći uvjet. Grozdovi u unutarnjim područjima imaju takav raspored koji osigurava da niti jedan grozd iz vanjskih područja, koja se trenutno ne promatraju, ne može biti bliži bilo kojem objektu iz promatranog pravokutnog područja, nego što su to grozdovi iz unutarnjih područja. Ako je uvjet zadovoljen svi vanjski grozdovi su odbačeni za sve objekte unutar promatranog područja bez računanja očekivane udaljenosti.

Prednosti postupka podjele područja skupa objekata su smanjenje broja relacija između objekata i grozdova koje se moraju promatrati, jer se ne promatraju vanjski grozdovi. Postupak se može kombinirati s postojećim metodama za razvrstavanje, tako da se one primjene za razvrstavanje objekata u pojedinom pravokutnom području. Pravokutna područja međusobno su nezavisna što pruža mogućnost za paralelno razvrstavanje. Svako pravokutno područje može se procesirati kao paralelan proces, što znatno skraćuje vrijeme razvrstavanja. U petom poglavlju provedeni su pokusi kojima su pokazane prednosti paralelnog izvođenja procesa razvrstavanja na računalima s više jezgri.

Razvijeni postupci iskorišteni su za stvaranje modela i predviđanje stanja uslužnog sustava razvrstavanjem postojećih podataka o objektima uslužnog sustava. Rezultati razvrstavanja iskorišteni su za opisivanje trenutnog stanja uslužnog sustava i predviđanje budućeg stanja. Za čuvanje podataka o uslužnom sustavu napravljena je baza podataka koja sadrži zemljopisne koordinate svih kućnih brojeva u Osijeku. Ulice su podijeljene na dijelove i svaki dio ulice predstavlja minimalno područje nesigurnosti jednog nesigurnog objekta. Nesigurni objekt uzorkovan je tako da svaki kućni broj predstavlja jedan uzorak u funkciji gustoće vjerojatnosti. Stoga broj uzoraka kojim je predstavljena funkcija gustoće vjerojatnosti ovisi o tome koliko kućnih brojeva ima u jednom nesigurnom objektu. Vjerojatnost da se objekt nalazi na nekom uzorku predstavlja broj zadataka ili učestalost zadataka uslužnog sustava na tome uzorku. Kako bi se čuvali podaci o zadacima uslužnog sustava napravljena je baza podataka u kojoj se spremaju svi zadaci koje je uslužni sustav obavio na pojedinom uzorku. Iz baze podataka se za svaki uzorak može saznati broj i vrijeme obavljanja zadataka, iz kojih se mogu izračunati vjerojatnosti za pojedini uzorak Na osnovu postojećih podataka o svim uzorcima u šestom poglavlju napravljeno je predviđanje stanja uslužnog sustava pomoću kojeg se može saznati gdje će se u budućnosti ukazati potreba za uslužnim sustavom. Predviđanjem se mogu znatno uštedjeti sredstva za održavanje uslužnog sustava i vrijeme reakcije (vrijeme potrebno da se obavi zadatak).

Razvrstavanjem postojećih podataka o uslužnom sustavu određena su središta grozdova u kojima se nalaze međusobno bliski zadaci. Središta grozdova predstavljaju mjesta na kojima je najveća koncentracija zadataka za uslužni sustav. Uslužni sustav može iskoristiti predviđanje i smjestiti svoje djelatnike u središtima grozdova kako bi što učinkovitije obavljali zadatke uslužnog sustava i uspješno obavljali nove zadatke. U šestom poglavlju izloženi su rezultati sustavne analize prostorno-vremenskih značajki uslužnog sustava. Pod vremenskim značajkama podrazumijevaju se različiti vremenski okviri u kojima se znatno mijenjaju zahtjevi za uslužnim sustavom. Budući da su potrebe za uslužnim sustavom ovisne o različitim događajima u gradu, napravljena su predviđanja za pojedine izvanredne situacije zbog kojih se mijenjaju zahtjevi za uslužnim sustavom. Model je iskoristio postojeće podatke u sličnim situacijama i pomoću njih predvidio buduće zahtjeve za uslužnim sustavom.

Disertacija je organizirana prema sljedećem redoslijedu. U drugom poglavlju opisani su općeniti postupci za razvrstavanje. Opisane su mjere sličnosti između objekata i različiti modeli za razvrstavanje objekata. Za svaki model predstavljeni su najpoznatiji algoritmi za razvrstavanje i navedene njihove prednosti i nedostaci.

U trećem poglavlju predstavljeni su postupci za razvrstavanje nesigurnih objekata, koji se znatno razlikuju od postupaka za razvrstavanje objekata koji ne sadrže nesigurnost. Navedeni postupci najčešće su nastali modifikacijom postupaka za razvrstavanje objekata koji ne sadrže nesigurnost.

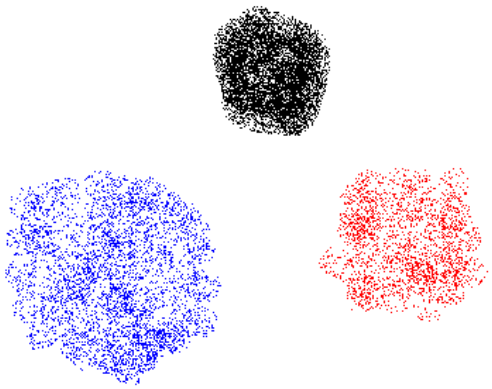
U četvrtom poglavlju opisani su postupci razvijeni u ovoj doktorskoj disertaciji i opisane njihove prednosti u odnosu na postojeće postupke. Objašnjeni su razlozi zbog kojih navedeni postupci imaju bolja svojstva od postojećih postupaka.

U petom poglavlju napravljeni su pokusi i analize koje uspoređuju nekoliko postupaka za razvrstavanje nesigurnih objekata. Pokusi su napravljeni za različite situacije u kojima se mijenja jedan od parametara dok ostali parametri zadržavaju osnovne vrijednosti. Također su napravljeni pokusi i analiza rezultata razvrstavanja pomoću paralelnih računalnih postupaka.

Na kraju, u šestom poglavlju predstavljen je model za predviđanje stanja uslužnog sustava. Opisano je razvrstavanje objekata uslužnog sustava i prednosti dobivene predviđanjem budućeg stanja uslužnog sustava. Napravljeni su pokusi za različite izvanredne situacije koje mijenjaju zahtjeve za uslužnim sustavom. One se nastoje predvidjeti razvrstavanjem postojećih podataka o uslužnom sustavu. Na samome kraju napisani su zaključak i sažetak doktorske disertacije.

2. Razvrstavanje objekata

Razvrstavanje (eng. Clustering) je proces koji skupinu objekata raspodjeljuje u homogene skupine prema određenim uvjetima. Takve skupine nazivaju se grozdovi i sastoje se od međusobno sličnih objekata. Razvrstavanje ima primjenu u brojnim područjima kao što su umjetna inteligencija, lokacijske usluge, procesiranje slika, prepoznavanje uzoraka, informacijske tehnologije, biologija, marketing i brojne druge znanstvene grane. Objekti koji se nalaze u istom grozdu slični su jedni drugima, a različiti su od objekata koji se nalaze u svim drugim grozdovima. Kako bi se opisala sličnost ili različitost među objektima upotrebljava se mjera sličnosti [3]. Mjera sličnosti ili koeficijent sličnosti se koristi kako bi se kvantitativno opisala sličnost između dva objekta. To znači, što je koeficijent sličnost veći dva objekta su sličnija, i obrnuto ako je koeficijent sličnosti manji objekti su manje slični. Ako ne postoji mjera sličnosti između različitih objekata tada nije moguće provesti ispravno razvrstavanje, jer ne postoje kvantitativni podaci prema kojima bi se ono napravilo. Postoje različite mjere sličnosti koje će biti obrađene u poglavlju 2.1. Svi objekti u jednom grozdu trebaju imati što veću mjeru sličnosti, imati malu međusobnu udaljenost, te se razlikovati od objekata u drugim grozdovima. Prema [4] postoje kompaktni i ulančani grozdovi. Kompaktni grozd je skup objekata koji imaju visoku mjeru sličnosti i grozd se može predstaviti sa središnjom točkom, to jest sa središtem grozda. Na slici 2.1 prikazano je nekoliko kompaktnih grozdova u dvodimenzionalnom prostoru. Sa slike je vidljivo da su grozdovi međusobno odvojeni i svaki od njih je predstavljen pomoću središnje točke.



Slika 2.1 Primjer triju grozdova s odvojenim podacima skupova

Ulančani grozd je skup objekata koji su međusobno slični i svaki objekt je povezan sa njemu susjednim objektima. Znači da mora postojati putanja u grozdu koja međusobno povezuje bilo koja dva objekta unutar grozda. Na slici 2.2 prikazana su dva ulančana grozda i vidljivo je da između bilo koja dva objekta unutar jednog grozda postoji putanja koja ih povezuje.

Slika 2.2 Primjer dvaju ulančanih grozdova s povezanim podacima skupova

Još jedna podjela algoritama za razvrstavanje je na jednoznačno razvrstavanje i neizrazito razvrstavanje (eng. fuzzy). U jednoznačnom razvrstavanju svaki objekt pripada jednom i samo jednom grozdu. Matematički rezultat takvog razvrstavanja može se predstaviti s *k x n* matricom:

$$O = \begin{pmatrix} o_{11} & o_{12} & \cdots & o_{1n} \\ o_{21} & o_{22} & \cdots & o_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ o_{k1} & o_{k2} & \cdots & o_{kn} \end{pmatrix} \tag{2.1}$$

gdje je *k* broj grozdova, *n* broj objekata, a vrijednosti o_{ji} zadovoljavaju sljedeće uvjete:

$$o_{ji} \in \{0,1\}, \;\; 1 \leq j \leq k, 1 < i < n \tag{2.2}$$

$$\sum_{j=1}^{k} o_{ji} = 1, \;\; 1 \leq i \leq n \tag{2.3}$$

$$\sum_{i=1}^{n} o_{ji} > 0, \;\; 1 \leq j \leq k \tag{2.4}$$

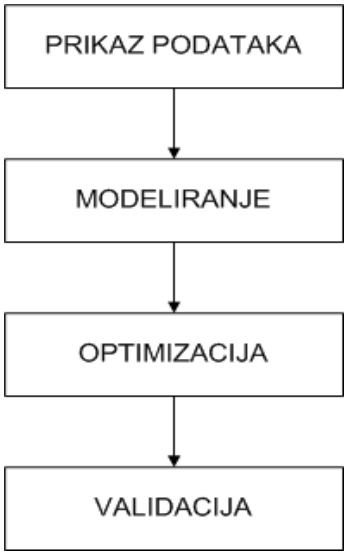
Iz jednadžbe (2.2) je vidljivo kako svaki objekt ili pripada ili ne pripada nekome od grozdova. Nula označava da objekt ne pripada grozdu, a jedinica označava da mu pripada. Jednadžba (2.3) označava da svaki objekt pripada samo jednom grozdu, dok jednadžba (2.4) osigurava da svaki grozd mora sadržavati barem jedan objekt, to jest nisu dozvoljeni prazni grozdovi koji ne sadržavaju objekte. U neizrazitom razvrstavanju objekt može pripadati jednom ili više grozdova s odgovarajućim vjerojatnostima pripadnosti pojedinome grozdu. Rezultat neizrazitog razvrstavanja se može predstaviti s k x n matricom kao u prošlom primjeru, ali sa sljedećim ograničenjima:

$$o_{ji} \in [0,1], \;\; 1 \leq j \leq k, 1 < i < n \tag{2.5}$$

$$\sum_{j=1}^{k} o_{ji} = 1, \;\; 1 \leq i \leq n \tag{2.6}$$

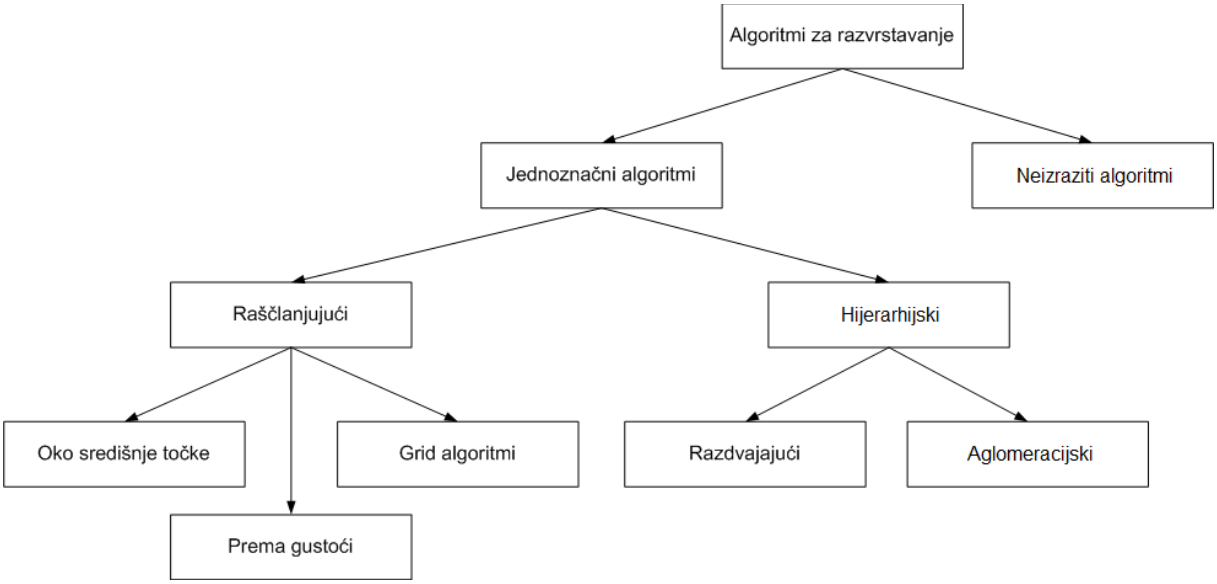
$$\sum_{i=1}^{n} o_{ji} > 0, \;\; 1 \leq j \leq k \tag{2.7}$$

Svako dobro dizajnirano razvrstavanje trebalo bi se sastojati od 4 faze [5]. Faze su prikaz podataka, modeliranje, optimizacija i validacija koje su ispravnim redoslijedom prikazane na slici 2.3. U prvoj fazi promatraju se moguće strukture koje se mogu pronaći među podacima i potrebno je pronaći najbolji način kako ih prikazati u poželjnom obliku. U fazi modeliranja se na osnovi prikaza podataka o objektima određuju pojmovi i kriteriji prema kojima će se raditi razvrstavanje. Odabire se i model razvrstavanja koji će se koristiti za stvaranje grozdova.



Slika 2.3 Četiri faze dobro dizajniranog procesa razvrstavanja

Modeli algoritama za razvrstavanje mogu se podijeliti u dvije općenite kategorije, a to su jednoznačni i neizraziti algoritmi. Jednoznačni algoritmi se nadalje dijele na raščlanjujuće i hijerarhijske algoritme. Raščlanjujući algoritmi mogu se podijeliti na brojne modele od kojih je najpoznatiji model razvrstavanja oko središnje točke (eng. Centroid model). Hijerarhijski algoritmi dijele se na razdvajajuće hijerarhijske algoritme i aglomeracijske hijerarhijske algoritme kao što je prikazano na slici 2.4.



Slika 2.4 Modeli i podjela algoritama za razvrstavanje

U razdvajajućim hijerarhijskim algoritmima kreće se od vrha prema dolje. Algoritam kreće s jednim velikim grozdom u kojem se nalaze svi objekti, i on se postupno razdvaja u manje grozdove. U aglomeracijskom hijerarhijskom algoritmu kreće se od dna prema vrhu. Na početku se u svakom grozdu nalazi samo jedan objekt i grozdovi se postupno spajaju u veće grozdove dok ne ostane jedan veliki grozd u kojem se nalaze svi objekti. Kod velikih skupova objekata hijerarhijski algoritmi su nepraktični, jer zahtijevaju $O(n^2)$ memorijskog prostora i $O(n^3)$ procesorskog vremena, gdje je n broj objekata. Za razliku od hijerarhijskih algoritama raščlanjujući algoritmi stvaraju grozdove na samo jednoj razini. Ti grozdovi se za razliku od hijerarhijskih algoritama, međusobno ne preklapaju u prostoru. Iako su provedena različita istraživanja koja bi rješavala općenite probleme razvrstavanja, većina praktičnih zadataka ima specifične probleme i zahtijevaju specifične metode razvrstavanja koje ovise o pojedinoj situaciji. Zato se može govoriti samo o općenitim algoritmima koji se dorađuju za svaku specifičnu primjenu u praksi. U praktičnim primjenama razvrstavanje se suočava i s

problemom da neki podaci ili nisu dostupni ili imaju grešku, to jest nesigurnost. Zbog toga se razvrstavanje nesigurnih objekata znatno razlikuje od razvrstavanja objekata koji nemaju nesigurnost, što je objašnjeno u trećem poglavlju. Ako za neki objekt nedostaju sva mjerenja tada se taj objekt mora ukloniti iz skupa objekata, a ako nedostaju samo neki podaci o objektu tada se može primijeniti jedan od dva sljedeća postupka:

1. Nepoznate vrijednosti nekog podatka o objektu se zamijene sa srednjom vrijednosti tog podatka svih objekata u grozdu.

2. S nepoznatim vrijednostima nekog podatka o objektu suočava se tijekom razvrstavanja objekata.

U fazi optimizacije obavljaju se prilagodbe korištenog modela, prikaza podataka o objektima i korištenih algoritama kako bi se dobili što bolji rezultati. U zadnjoj fazi validacije obavlja se provjera dobivenih rezultata i provjerava ispravnost razvrstavanja. Ako se u bilo kojoj od faza utvrdi pogreška vraća se u prethodnu fazu i ispravlja se pronađena pogreška.

2.1. Mjere sličnosti

Pri razvrstavanju objekata mora se koristiti mjera sličnosti pomoću koje će se odrediti pripadnost objekta nekome od grozdova. Koeficijent sličnosti označava jačinu veze između dva objekta. Što su dva objekta međusobno sličnija koeficijent sličnosti je veći. Ako su $\boldsymbol{x} = (x_1, x_2, \dots, x_d)$ i $\boldsymbol{y} = (y_1, y_2, \dots, y_d)$ dvije točke (objekta) u višedimenzionalnom prostoru tada je koeficijent sličnosti između točaka $\boldsymbol{x}$ i $\boldsymbol{y}$ funkcija vrijednosti njihovih atributa, to jest njihovih koordinata u prostoru.

$$s(\boldsymbol{x}, \boldsymbol{y}) = s(x_1, x_2, \dots, x_d, y_1, y_2, \dots, y_d) \tag{2.8}$$

Mjera sličnosti mora zadovoljavati tri sljedeća svojstva:

1. $0 \leq s(\boldsymbol{x}, \boldsymbol{y}) \leq 1$

2. $s(\boldsymbol{x}, \boldsymbol{y}) = 1 \iff \boldsymbol{x} = \boldsymbol{y}$

3. $s(\boldsymbol{x}, \boldsymbol{y}) = s(\boldsymbol{y}, \boldsymbol{x})$

gdje su $\boldsymbol{x}$ i $\boldsymbol{y}$ proizvoljne točke u višedimenzionalnom prostoru.

Vidi se da mjera sličnosti ne smije biti negativna i za dva ista objekta koeficijent sličnost jednak je jedan. Mjera sličnosti je većinom simetrična tako da vrijedi $s(\boldsymbol{x}, \boldsymbol{y})= s(\boldsymbol{y}, \boldsymbol{x})$, no postoje slučajevi asimetrične mjere sličnosti [6]. Kao mjera sličnost između dva objekta može se koristiti i udaljenost. Ako se kao mjera sličnosti koristi udaljenost između dvije točke tada udaljenost mora zadovoljavati sljedeće uvjete:

1. Udaljenost ne može biti negativna $d(\boldsymbol{x}, \boldsymbol{y}) \geq 0$
2. Refleksivnost $d(\boldsymbol{x}, \boldsymbol{y}) = 0 <=> \boldsymbol{x} = \boldsymbol{y}$
4. Komutacija $d(\boldsymbol{x}, \boldsymbol{y})$= d$(\boldsymbol{y}, \boldsymbol{x})$
3. Nejednakost trokuta $d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z}) + d(\boldsymbol{y}, \boldsymbol{z})$

   gdje su $\boldsymbol{x}$, $\boldsymbol{y}$ i $\boldsymbol{z}$ proizvoljne točke u višedimenzionalnom prostoru.

Ako je D skup objekta, tada prema [7] postoji 12 različitih struktura koje mogu opisati mjeru sličnosti S:

1. Mjera sličnosti S nad skupom objekata $D \times D$ je euklidska udaljenost.
2. Mjera sličnosti S nad skupom objekata $D \times D$ je metrička.
3. Mjera sličnosti S nad skupom objekata $D \times D$ je simetrična.
4. Mjera sličnosti S nad skupom objekata $D \times D$ je ispravno vrednovana.
5. Mjera sličnosti S nad skupom objekata $D \times D$ je potpuna, što znači da je svaki par objekata je uspoređen jedan s drugim.
6. Mjera sličnosti S nad skupom objekata $D \times D$ je djelomična, što znači da je samo dio parova objekata uspoređen.
7. Mjera sličnosti S je stablo nad skupom objekata D.
8. Mjera sličnosti S je potpuna „relativna sličnost“ nad D, to jest za svaki i u skupu D vrijedi $j \leq_i k$. To znači da j nije sličniji i nego što je to k.
9. Mjera sličnosti S je djelomična „relativna sličnost“ nad D.
10. Mjera sličnosti S je dihotomija nad D u kojoj je $D \times D$ podijeljen u skup sličnih parova i u skup različitih parova.
11. Mjera sličnosti S je trihotomija nad D u kojoj je $D \times D$ podijeljen u skup sličnih parova, skup različitih parova i skup preostalih parova.
12. Mjera sličnosti S je podskup od D koji čine samo slični objekti.

2.1.1. Matrice blizine

Matrica blizine u sebi sadrži podatke o međusobnoj bliskosti skupa objekata. U općem slučaju matrica blizine je simetrična matrica u kojoj se indeks sličnosti odnosi na mjeru sličnosti, a može se odnositi i na mjeru različitosti. Matrica blizine sadrži podatke o bliskosti između svih objekata i opisuje koliko je bilo koji objekt sličan s bilo kojim drugim objektom u skupu objekata. Primjer matrice blizine je matrica udaljenosti. Za prostorni skup objekata $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$ u kojem je svaki objekt $\boldsymbol{x_i}$ opisan vektorom $\boldsymbol{x_i} = (x_{i1}, x_{i2}, \dots, x_{id})$ matrica udaljenosti $M_{udalj}(D)$ opisana je sljedećom jednadžbom:

$$M_{udalj}(D) = \begin{pmatrix} 0 & d_{12} & \dots & d_{1n} \\ d_{21} & 0 & \dots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \dots & 0 \end{pmatrix} \tag{2.9}$$

gdje je $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j)$, a d predstavlja neku od funkcija udaljenosti.

U sljedećim poglavljima predstavljene su najčešće korištene funkcije udaljenosti. Matrica sličnosti za isti skup prostornih objekata D opisana je sljedećom jednadžbom:

$$M_{sl}(D) = \begin{pmatrix} 1 & s_{12} & \dots & s_{1n} \\ s_{21} & 1 & \dots & s_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n1} & s_{n2} & \dots & 1 \end{pmatrix} \tag{2.10}$$

gdje je $s_{ij} = s(\boldsymbol{x}_i, \boldsymbol{x}_j)$, a s predstavlja neku od mjera sličnosti.

Matrica udaljenosti $M_{udalj}(D)$ i matrica sličnosti $M_{sl}(D)$ nad skupom objekata D su primjeri matrica blizine. Ako su funkcija udaljenosti i mjera sličnosti simetrične, tada će i dvije matrice blizine također biti simetrične. Osim matricama blizine mjera sličnosti može se iskazati i grafovima blizine. Graf blizine je težinski graf u kojem čvorove čine razvrstani objekti, a grane predstavljaju koeficijente sličnosti kao što to čine i unosi u matricama blizine. Koeficijenti sličnosti su isti u oba slučaja, ali su predstavljeni na različit način. Ako je graf blizine usmjeren onda se može predstaviti s asimetričnom matricom blizine, a ako graf nije usmjeren može se predstaviti simetričnom matricom blizine.

2.1.2. Matrice raspršenosti

Za skup objekata $D = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_n}\}$ u kojem je svaki objekt $\boldsymbol{x_i}$ opisan višedimenzionalnim vektorom $\boldsymbol{x_i} = (x_{i1}, x_{i2}, \dots, x_{id})$ matrica raspršenosti opisana je sljedećom jednadžbom [8] :

$$\mathrm{M_{ras}(D)} = \sum_{i=1}^{n} (\mathbf{x_i} - \bar{\mathbf{x}})^{\mathrm{T}} (\mathbf{x_i} - \bar{\mathbf{x}}) \tag{2.11}$$

gdje je $\bar{\mathbf{x}}$ aritmetička sredina svih vrijednosti od $\boldsymbol{x}$ opisana jednadžbom:

$$\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x_i} \tag{2.12}$$

Može se reći da matrica raspršenosti $\mathrm{M_{ras}(D)}$ predstavlja sumu kvadrata, a trag matrice predstavlja statističku raspršenost skupa objekata D i opisan je sljedećom jednadžbom:

$$\mathrm{Tr(M_{ras}(D))} = \sum_{i=1}^{n} (\mathbf{x_i} - \bar{\mathbf{x}}) \, (\mathbf{x_i} - \bar{\mathbf{x}})^{\mathrm{T}} \tag{2.13}$$

Za grozd $C = \{C_1, C_2, \dots, C_k\}$ koji je podskup skupa objekata D matrica raspršenosti $\mathrm{M_{uras}(C)}$ unutar grozda naziva se i unutrašnja matrica raspršenosti grozda C. Tada je matrica raspršenosti predstavljena sljedećom jednadžbom:

$$\mathrm{M_{uras}(C)} = \sum_{i=1}^{k} \sum_{\mathbf{x} \epsilon \mathrm{C_i}} (\mathbf{x} - \mathbf{z_i})^{\mathrm{T}} (\mathbf{x} - \mathbf{z_i}) \tag{2.14}$$

gdje je $\mathbf{z_i}$ srednja vrijednost grozda C_i i opisana je sljedećom jednadžbom:

$$\mathbf{z_i} = \frac{1}{|C_i|} \sum_{\mathbf{x} \epsilon \mathrm{C_i}} \mathbf{x} \tag{2.15}$$

Isto tako matrica raspršenosti između dva grozda može se opisati sljedećom jednadžbom:

$$\mathrm{M_{mras}(C)} = \mathrm{M_{ras}(C)} - \mathrm{M_{uras}(C)} \tag{2.16}$$

2.1.3. Matrica kovarijanci

Za skup objekata $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$ u kojem je svaki objekt $\boldsymbol{x_i}$ opisan višedimenzionalnim vektorom $\boldsymbol{x_i} = (x_{i1}, x_{i2}, \dots, x_{id})$, kovarijanca između dva takva atributa x_p i x_q definirana je kao omjer sume produkata njihovih standardnih devijacija od srednjeg broja objekata [9] i opisana je sljedećom jednadžbom:

$$\mathrm{c_{pq}} = \frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{x_{ip}} - \overline{\mathrm{x_p}}\right)\left(\mathrm{x_{iq}} - \overline{\mathrm{x_q}}\right) \tag{2.17}$$

gdje je x_{ij} j-ti atribut objekata $\mathbf{x}_\mathrm{i}$ i $\overline{\mathrm{x}_\mathrm{j}}$ je srednja vrijednosti svih podataka na j-tom atributu. Srednja vrijednost svih podataka na j-tom atributu opisana je sljedećom jednadžbom:

$$\overline{\mathrm{x}_\mathrm{j}} = \frac{1}{n}\sum_{i=1}^{n}\mathrm{x_{ij}}\ , \qquad \mathrm{j} = 1,2,\dots,\mathrm{d} \tag{2.18}$$

Matrica kovarijanci je matrica veličine $d \times d$ u kojoj svaki unos (p, q) predstavlja kovarijancu između atributa x_p i x_q.

$$\sum = \begin{pmatrix} c_{11} & c_{12} & \dots & c_{1d} \\ c_{21} & c_{22} & \dots & c_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & c_{dd} \end{pmatrix} \tag{2.19}$$

Gornja jednadžba može se opisati i na drugi način:

$$\sum = \frac{1}{n}\boldsymbol{X}^T\boldsymbol{X} \tag{2.20}$$

gdje $\boldsymbol{X}^T$ predstavlja transponiranu matricu od $\boldsymbol{X}$ veličine $n \times d$ opisanu jednadžbom:

$$\boldsymbol{X} = (x_{ij} - \bar{x}_j)_{n\times d} = \begin{pmatrix} \boldsymbol{x}_1 - \overline{\mathrm{x}_1}\mathbf{e}_\mathrm{d} \\ \boldsymbol{x}_2 - \overline{\mathrm{x}_2}\mathbf{e}_\mathrm{d} \\ \vdots \\ \boldsymbol{x}_d - \overline{\mathrm{x}_\mathrm{d}}\mathbf{e}_\mathrm{d} \end{pmatrix} \tag{2.21}$$

gdje je $\mathbf{e}_\mathrm{d}$ jedinični vektor sa d dimenzija opisan jednadžbom:

$$\mathbf{e}_\mathrm{d} = (1,1,\dots 1) \tag{2.22}$$

2.2. Mjere za udaljenost brojčanih podataka

Odabir mjere za udaljenost u pojedinoj primjeni ovisi o brojnim čimbenicima kojima je opisan skup objekata. Bitno je odabrati mjeru za udaljenost koja će najbolje odgovarati u pojedinom slučaju. Najbolja mjera najčešće se odabire kao rezultat iskustva, vještine i znanja s prethodnih projekata. U idućim poglavljima opisane su mjere za udaljenost koje se najčešće koriste pri razvrstavanju objekata.

2.2.1. Euklidska udaljenost

Euklidska udaljenost je najčešće korištena mjera za udaljenost brojčanih podataka. Za dva objekta $\boldsymbol{x}$ i $\boldsymbol{y}$ u prostoru s d dimenzija euklidska udaljenost opisana je sljedećom jednadžbom:

$$\mathrm{d}_{\mathrm{euk}}(\boldsymbol{x},\boldsymbol{y}) = \left[\sum_{j=1}^{d}\left(\mathrm{x_j}-\mathrm{y_j}\right)^2\right]^{\frac{1}{2}} = [(\boldsymbol{x}-\boldsymbol{y})(\boldsymbol{x}-\boldsymbol{y})^T]^{\frac{1}{2}} \tag{2.23}$$

gdje su $\mathrm{x_j}$ i $\mathrm{y_j}$ vrijednosti j-tog atributa objekata $\boldsymbol{x}$ i $\boldsymbol{y}$. Osim euklidske udaljenosti može se koristiti i kvadratna euklidska udaljenost koja je opisana sljedećom jednadžbom:

$$\mathrm{d}_{\mathrm{kveuk}}(\boldsymbol{x},\boldsymbol{y}) = \mathrm{d}_{\mathrm{euk}}(\boldsymbol{x},\boldsymbol{y})^2 = \sum_{j=1}^{d}\left(\mathrm{x_j}-\mathrm{y_j}\right)^2 = (\boldsymbol{x}-\boldsymbol{y})(\boldsymbol{x}-\boldsymbol{y})^T \tag{2.24}$$

2.2.2. Manhattan udaljenost

Manhattan udaljenost također se naziva i udaljenost gradskih blokova (eng. City block distance), jer se upotrebljava za računanje udaljenosti između gradskih blokova. Opisana je kao suma udaljenosti svih atributa. Za dva objekta $\boldsymbol{x}$ i $\boldsymbol{y}$ u prostoru s d dimenzija Manhattan udaljenost opisana je sljedećom jednadžbom:

$$\mathrm{d}_{\mathrm{man}}(\boldsymbol{x},\boldsymbol{y}) = \sum_{j=1}^{d}\left|\mathrm{x_j}-\mathrm{y_j}\right| \tag{2.25}$$

Ako za neki od objekata $\boldsymbol{x}$ ili $\boldsymbol{y}$ nedostaje vrijednost za pojedini atribut, Manhattan udaljenost se u tome slučaju opisuje sljedećom jednadžbom [10] :

$$\mathrm{d}_{\mathrm{manw}}(\boldsymbol{x},\boldsymbol{y}) = \sum_{j=1}^{d} \frac{\mathrm{w_j}\left|\mathrm{x_j} - \mathrm{y_j}\right|}{\sum_{\mathrm{j=1}}^{\mathrm{d}} \mathrm{w_j}} \tag{2.26}$$

gdje je $\mathrm{w_j} = 1$ ako oba objekta $\boldsymbol{x}$ i $\boldsymbol{y}$ imaju poznate vrijednosti j-tog atributa, i $\mathrm{w_j} = 0$ ako jedna od vrijednosti nije poznata.

Postoji i djelomična verzija Manhattan udaljenosti u kojoj se pri računanju udaljenosti koristi samo dio od ukupnog broja dimenzija svih atributa. Opisana je jednadžbom [11]:

$$\mathrm{d}_{\mathrm{djelman}}(\boldsymbol{x},\boldsymbol{y}) = \sum_{j=P}^{d} \frac{\left|\mathrm{x_j} - \mathrm{y_j}\right|}{|\mathrm{P}|} \tag{2.27}$$

gdje je P podskup ukupnog skupa objekata D, i P ne smije biti prazan skup.

2.2.3. Maksimalna udaljenost

Maksimalna udaljenost definirana je kao maksimalna vrijednost udaljenosti svih atributa. Za dva objekta $\boldsymbol{x}$ i $\boldsymbol{y}$ u prostoru s d dimenzija maksimalna udaljenost opisana je sljedećom jednadžbom:

$$\mathrm{d}_{\mathrm{max}}(\boldsymbol{x},\boldsymbol{y}) = max_{1 \le j \le d}\left|\mathrm{x_j} - \mathrm{y_j}\right| \tag{2.28}$$

2.2.4. Minkowskijeva udaljenost

Euklidska udaljenost, Manhattan udaljenost, i maksimalna udaljenost tri su specijalna slučaja Minkowskijeve udaljenosti. Minkowskijeva udaljenost opisana je sljedećom jednadžbom [12]:

$$\mathrm{d_{Min}}(\boldsymbol{x},\boldsymbol{y}) = \left(\sum_{j=1}^{d}\left|\mathrm{x_j} - \mathrm{y_j}\right|^{\mathrm{r}}\right)^{\frac{1}{r}}, \quad r \geq 1 \tag{2.29}$$

gdje r predstavlja red Minkowskijeve udaljenosti.

Za $\mathrm{r} = 2$ predstavlja euklidsku udaljenost, za $\mathrm{r} = 1$ Manhattan udaljenost i za $\mathrm{r} = \infty$ maksimalnu udaljenost. Ako se u skupu objekata nalaze kompaktni ili izolirani grozdovi Minkowskijeva udaljenost daje dobre rezultate [13]. U suprotnom atributi s većim vrijednostima dominiraju nad ostalima, a kako bi se to izbjeglo atribute je potrebno normalizirati ili koristiti težinske sheme [14].

2.2.5. Mahalanobisova udaljenost

Mahalanobisova udaljenost može ublažiti distorziju udaljenosti prouzrokovanu linearnim kombinacijama atributa [13]. Mahalanobisova udaljenost opisana je sljedećom jednadžbom:

$$\mathrm{d_{Mah}}(\boldsymbol{x},\boldsymbol{y}) = \sqrt{(\boldsymbol{x}-\boldsymbol{y})\sum^{-1}(\boldsymbol{x}-\boldsymbol{y})^{T}} \tag{2.30}$$

gdje je $\sum$ matrica kovarijanci opisana u poglavlju 2.1.3.

Jednadžba (2.30) upotrebljava težinsku shemu nad objektima. Još jedno važno svojstvo Mahalanobisove udaljenosti je invarijantnost na nejedinične transformacije. Ako je C nejedinična $d \times d$ matrica primijenjena nad skupom objekata D prema sljedećoj jednadžbi:

$$\mathbf{y}_\mathrm{i} = \mathrm{C}\mathbf{x}_\mathrm{i}, \quad \mathrm{i} = 1,2,\ldots,\mathrm{n} \tag{2.31}$$

tada se nova matrica kovarijanci računa prema sljedećoj jednadžbi:

$$\frac{1}{\mathrm{n}}\mathrm{Y^T}\mathbf{Y} = \frac{1}{\mathrm{n}}(\mathbf{X}\mathrm{C^T})^{\mathrm{T}}(\mathbf{X}\mathrm{C^T}) \tag{2.32}$$

gdje je **X** opisan u jednadžbi (2.21), a **Y** se slično definira za transformirani skup objekata.

Mahalanobisova udaljenost između točaka $\mathbf{y}_\mathrm{i}$ i $\mathbf{y}_\mathrm{j}$ opisana je sljedećom jednadžbom:

$$\begin{aligned} \mathrm{d}_{\mathrm{Mah}}(\boldsymbol{y_i}, \boldsymbol{y}_j) &= \sqrt{(\boldsymbol{x_i} - \boldsymbol{x_j})\mathrm{C}^{\mathrm{T}}\left((\mathbf{X}\mathrm{C}^{\mathrm{T}})^{\mathrm{T}}(\mathbf{X}\mathrm{C}^{\mathrm{T}})\right)^{-1} C\left(\boldsymbol{x_i} - \boldsymbol{x_j}\right)^T} \\ &= \sqrt{(\boldsymbol{x_i} - \boldsymbol{x_j})\left(\frac{1}{\mathrm{n}}\mathbf{X}^{\mathrm{T}}\mathbf{Y}\right)^{-1}\left(\boldsymbol{x_i} - \boldsymbol{x_j}\right)^T} = \mathrm{d}_{\mathrm{Mah}}(\boldsymbol{x_i}, \boldsymbol{x_j}) \end{aligned} \tag{2.33}$$

Jednadžba (2.33) pokazuje da je Mahalanobisova udaljenost invarijantna na nejedinične transformacije. U [15] je predložena generalizirana Mahalanobisova udaljenost koja uključuje težine pojedinih atributa. Neka je λ_j težina dodijeljena j-tom atributu i neka je Λ dijagonalna matrica s težinama opisana sljedećom jednadžbom:

$$\Lambda = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_\mathrm{d} \end{pmatrix} \tag{2.34}$$

Generalizirana Mahalanobisova udaljenost može se opisati sljedećom jednadžbom:

$$\mathrm{d}_{\mathrm{gMah}}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(\boldsymbol{x} - \boldsymbol{y})\Lambda\sum{}^{-1}\Lambda(\boldsymbol{x} - \boldsymbol{y})^T} \tag{2.35}$$

Nedostaci Mahalanobisove udaljenosti su veliki troškovi proračuna, jer se matrica kovarijanci računa nad čitavim skupom objekata D.

2.2.6. Ostale udaljenosti

Jedna od mana euklidske udaljenosti su slučajevi kada nisu poznate vrijednosti nekih atributa. Euklidska udaljenost između takvih objekata može biti manja nego između objekata koji imaju poznate vrijednosti istog atributa. U takvim slučajevima euklidsku udaljenost potrebno je izmijeniti pa se dobije nova udaljenost koja se naziva prosječna udaljenost. Za dva objekta $\boldsymbol{x}$ i $\boldsymbol{y}$ u prostoru s d dimenzija prosječna udaljenost opisana je sljedećom jednadžbom:

$$\mathrm{d}_{\mathrm{pros}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{d}\left(\sum_{j=1}^{d}\left|\mathrm{x_j} - \mathrm{y_j}\right|^2\right)^{\frac{1}{2}} \tag{2.36}$$

Druga modifikacija euklidske udaljenosti je tetivna udaljenost koja je opisana kao duljina tetive koja spaja dvije normalizirane točke unutar sfere s radijusom jednakim jedan [16]. Za dva objekata $\boldsymbol{x}$ i $\boldsymbol{y}$ u prostoru s d dimenzija tetivna udaljenost opisana je sljedećom jednadžbom:

$$\mathrm{d}_{\mathrm{tet}}(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{d}\left(2 - 2\frac{\sum_{j=1}^{d} \mathrm{x_j y_j}}{\|\boldsymbol{x}\|_2 \|\boldsymbol{y}\|_2}\right)^{\frac{1}{2}} \tag{2.37}$$

gdje $\|\boldsymbol{x}\|_2$ predstavlja L_2 normu opisanu sljedećom jednadžbom:

$$\|\boldsymbol{x}\|_2 = \sqrt{\sum_{j=1}^{d} \mathrm{x_j}^2} \tag{2.38}$$

Tetivna udaljenost rješava problem sa skaliranjem mjera udaljenosti i koristi se za rješavanje nedostataka euklidske udaljenosti. Također postoji i modifikacija tetivne udaljenosti koja se zove geodetska udaljenost [17].

2.3. Mjere sličnosti između grozdova

Ako se koristi hijerarhijski aglomeracijski algoritam za razvrstavanje tada se u svakoj iteraciji dvije slične grupe grozdova spajaju kako bi se stvorio novi grozd. Navedeni proces se nastavlja dok se ne dobije željeni broj grozdova. Nasuprot tome, u razdvajajućem hijerarhijskom razvrstavanju radi se obrnuto. Kreće se sa svim objektima u jednom grozdu koji se postupno dijeli u manje grozdove. U oba slučaja potrebno je izračunati udaljenost između objekata i grozdova te udaljenost između dva grozda. U sljedećim poglavljima opisane su mjere sličnosti između dvaju grozdova $C_1 = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_p\}$ i $C_2 = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \dots, \boldsymbol{z}_q\}$ gdje su p i q broj objekata u pojedinom grozdu. Opisani su načini na koji se može prikazati sličnost između dva grozda. Jedan od načina na koji se mogu utvrditi razlike među grozdovima je mjerenje udaljenosti između središta grozdova. Ako su C_1 i C_2 dva grozda koji sadrže brojčane podatke o objektima tada se udaljenost između središta grozdova može izračunati prema sljedećoj jednadžbi:

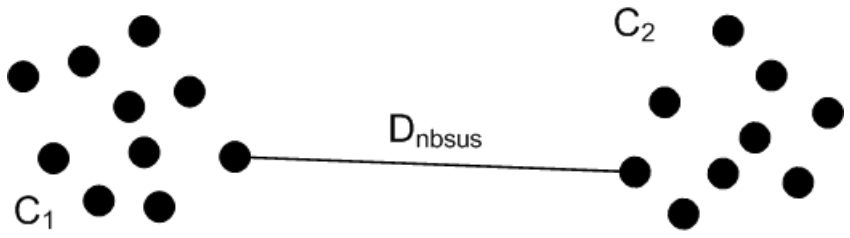
$$D_{sred}(C_1, C_2) = d(\mu(C_1), \mu(C_2)) \tag{2.39}$$

gdje su $\mu(C_1)$ i $\mu(C_2))$ središta grozdova izračunata prema sljedećoj jednadžbi:

$$\mu(C_j), = \frac{1}{|C_j|} \sum_{\mathbf{x} \in C_j} \mathbf{x}, \quad \text{j} = 1,2. \tag{2.40}$$

Za mjeru sličnosti između dva grozda može se koristiti i udaljenost između najbližih susjeda D_{nbsus}. Za neku funkciju udaljenosti d između dva grozda C_1 i C_2 ona se može izračunati prema sljedećoj jednadžbi [18] :

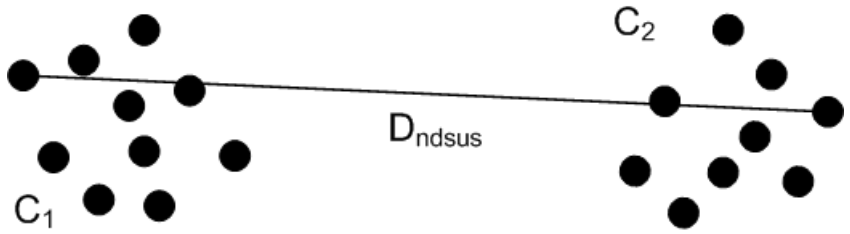
$$D_{nbsus}(C_1, C_2) = min_{1 \le i \le p, 1 \le j \le q} d(\boldsymbol{y}_i, \boldsymbol{z}_j) \tag{2.41}$$



Slika 2.5 Udaljenost između najbližih susjeda za grozdove C_1 i C_2

Udaljenost između najbližih susjeda pronalazi najmanju udaljenost između nekog objekta u grozdu C_1 i objekta u grozdu C_2, to jest daje najmanju udaljenost između dva grozda. Slično tome može se opisati udaljenost između najdaljih susjeda D_{ndsus}. Za neku funkciju udaljenosti d između dva grozda C_1 i C_2 udaljenost između najdaljih susjeda može se izračunati prema sljedećoj jednadžbi [19] :

$$D_{ndsus}(C_1, C_2) = max_{1\leq i\leq p, 1\leq j\leq q} d(\boldsymbol{y}_i, \boldsymbol{z}_j) \tag{2.42}$$



Slika 2.6 Udaljenost između najdaljih susjeda za grozdove C_1 i C_2

Udaljenost između najdaljih susjeda pronalazi najveću udaljenost između nekog objekta u grozdu C_1 i objekta u grozdu C_2, to jest daje najveću udaljenost između dva grozda.

Još jedan način mjerenja sličnosti između grozdova je prosječna udaljenosti D_{prosus}. Prosječna udaljenost između susjeda za danu funkciju udaljenosti d između dva grozda C_1 i C_2 može se izračunati prema sljedećoj jednadžbi [19] :

$$D_{prosus}(C_1, C_2) = \frac{1}{pq}\sum_{i=1}^{p}\sum_{j=1}^{q} d(\boldsymbol{y}_i, \boldsymbol{z}_j) \tag{2.43}$$

Prosječna udaljenost računa prosječnu udaljenost između svih objekta u grozdu C_1 i svih objekata u grozdu C_2, to jest daje prosječnu udaljenost između dva grozda.

2.4. Aglomeracijski hijerarhijski algoritmi

Aglomeracijski hijerarhijski algoritmi za razvrstavanje mogu se podijeliti na grafičke i geometrijske algoritme. U grafičke algoritme pripadaju metoda jednostruke veze, metoda potpune veze, metoda srednje vrijednosti grupe i težinska metoda srednje vrijednosti grupe, dok u geometrijske algoritme pripada Ward-ova metoda, metoda središnje točke i metoda srednje vrijednosti.

U grafičkim algoritmima grozdovi se predstavljaju pomoću grafova ili međusobno povezanih točaka, dok se u geometrijskim algoritmima grozdovi predstavljaju pomoću središnje točke. Memorijski zahtjevi hijerarhijskih algoritama mogu se smanjiti ako se grozd predstavi pomoću središnje točke. Za matricu sličnosti ili različitosti treba $O(n^2)$ memorijskog prostora, gdje je n broj objekata. Središnja točka novog grozda može biti izvedena iz središnjih točaka grozdova koji ga tvore kao što je prikazano u tablici 2.1.

Tablica 2.1 Računanje središta novog grozda nastalog spajanjem dvaju grozdova

Naziv metode	$\mu(C_i \cup C_j)$	Različitost između C_i i C_j
Metoda srednje vrijednosti	$\frac{\mu(C_i) + \mu(C_j)}{2}$	$\left\|\mu(C_i) - \mu(C_j)\right\|^2$
Metoda središnje točke	$\frac{\lvert C_i\rvert\mu(C_i) + \lvert C_j\rvert\mu(C_j)}{\lvert C_i\rvert + \lvert C_j\rvert}$	$\left\|\mu(C_i) - \mu(C_j)\right\|^2$

2.4.1. Metoda jednostruke veze

Metoda jednostruke veze predstavljena je u [20] i pripada u jednostavnije hijerarhijske algoritme za razvrstavanje. Metoda je poznata i pod drugim imenima kao što su metoda najbližeg susjeda, minimum metoda i metoda povezanosti [21]. Metoda jednostruke veze je invarijantna na monotone transformacije, na primjer logaritamske transformacije. Metoda koristi udaljenost između najbližih susjeda za mjerenje sličnosti između dvaju grozdova. Ako su C_i, C_j i C_k tri grozda tada se udaljenost između C_k i $C_i \cup C_j$ može izračunati prema Lance-Williams-ovoj jednadžbi [22]:

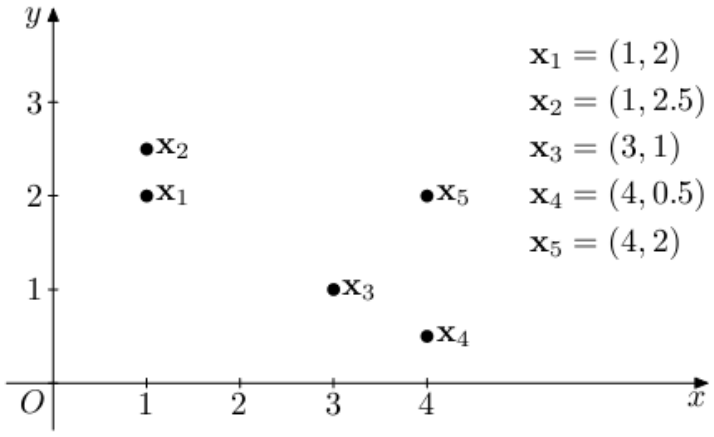
$$\begin{aligned} D\big(C_k, C_i \cup C_j\big) &= \frac{1}{2}D(C_k, C_i) + \frac{1}{2}D\big(C_k, C_j\big) - \frac{1}{2}\Big(D(C_k, C_i) - D\big(C_k, C_j\big)\Big) \\ &= min\big\{D(C_k, C_i), D\big(C_k, C_j\big)\big\} \end{aligned} \tag{2.44}$$

Iz jednadžbe (2.44) lako je zaključiti kako vrijedi sljedeće:

$$D(C_k, C_j) = min_{\boldsymbol{x}\in C_k, \boldsymbol{y}\in C_j} d(\boldsymbol{x}, \boldsymbol{y}) \tag{2.45}$$

gdje su C_k i C_j dva ne prazna i različita grozda koja se ne preklapaju, a $d(\boldsymbol{x}, \boldsymbol{y})$ je funkcija udaljenosti koja se koristi za računanje matrica različitosti.

Prema [21] algoritmi jednostruke veze dijele se na pet različitih tipova, a jedan od tipova su algoritmi povezanosti. Algoritmi povezanosti temeljeni su na teoriji grafova, a objekti se predstavljaju kao vrhovi grafa. Dva vrha (i, j) su povezani s granama samo u slučaju ako je udaljenost između dva vrha $d_{ij} \leq \Delta$, gdje grozdovi na razini Δ odgovaraju povezanim pod-grafovima. Algoritmi povezanosti su zahtjevni za računanje, jer je njihova složenost $O(n^5)$, gdje je n broj objekata [23]. Za ilustraciju naveden je primjer skupa objekata iz [12], koji je prikazan na slici 2.7. Matrica udaljenosti izračunata je upotrebom euklidske udaljenosti i prikazana je u tablici 2.2.



Slika 2.7 Skup prostornih podataka s pet različitih točaka u dvodimenzionalnom prostoru

Tablica 2.2 Matrica udaljenosti u prvoj iteraciji

	x_1	x_2	x_3	x_4	x_5
x_1	0	0.5	2.24	3.35	3
x_2	0.5	0	2.5	3.61	3.04
x_3	2.24	2.5	0	1.12	1.41
x_4	3.35	3.61	1.12	0	1.5
x_5	3	3.04	1.41	1.5	0

U metodi jednostruke veze u prvoj iteraciji grozdovi $\boldsymbol{x}_1$ i $\boldsymbol{x}_2$ se spajaju i tvore novi grozd, budući da oni imaju najmanju međusobnu udaljenost u matrici udaljenosti. U drugoj iteraciji računaju se udaljenosti između novog grozda i preostalih grozdova:

$$D(\{\boldsymbol{x}_1, \boldsymbol{x}_2\}, \boldsymbol{x}_3) = min\{d(\boldsymbol{x}_1, \boldsymbol{x}_3), d(\boldsymbol{x}_2, \boldsymbol{x}_3)\} = 2.24$$

$$D(\{\boldsymbol{x}_1, \boldsymbol{x}_2\}, \boldsymbol{x}_4) = min\{d(\boldsymbol{x}_1, \boldsymbol{x}_4), d(\boldsymbol{x}_2, \boldsymbol{x}_4)\} = 3.35$$

$$D(\{\boldsymbol{x}_1, \boldsymbol{x}_2\}, \boldsymbol{x}_5) = min\{d(\boldsymbol{x}_1, \boldsymbol{x}_5), d(\boldsymbol{x}_2, \boldsymbol{x}_5)\} = 3$$

Nakon izračuna dobije se matrica udaljenosti u drugoj iteraciji:

Tablica 2.3 Matrica udaljenosti u drugoj iteraciji

	$\{x_1, x_2\}$	x_3	x_4	x_5
$\{x_1, x_2\}$	0	2.24	3.35	3
x_3	2.24	0	1.12	1.41
x_4	3.35	1.12	0	1.5
x_5	3	1.41	1.5	0

U drugoj iteraciji spajaju se $\boldsymbol{x}_3$ i $\boldsymbol{x}_4$, jer oni imaju najmanju međusobnu udaljenost. U trećoj iteraciji se ponovno računaju udaljenosti između novog grozda i preostalih grozdova:

$$D(\{\boldsymbol{x}_3, \boldsymbol{x}_4\}, \{\boldsymbol{x}_1, \boldsymbol{x}_2\}) = min\{d(\boldsymbol{x}_1, \boldsymbol{x}_3), d(\boldsymbol{x}_2, \boldsymbol{x}_3), d(\boldsymbol{x}_1, \boldsymbol{x}_4), d(\boldsymbol{x}_2, \boldsymbol{x}_4)\}$$

$$= min\{D(\{\boldsymbol{x}_1, \boldsymbol{x}_2\}, \boldsymbol{x}_3), D(\{\boldsymbol{x}_1, \boldsymbol{x}_2\}, \boldsymbol{x}_4)\} = 2.24$$

$$D(\{\boldsymbol{x}_3, \boldsymbol{x}_4\}, \boldsymbol{x}_5) = min\{d(\boldsymbol{x}_3, \boldsymbol{x}_5), d(\boldsymbol{x}_4, \boldsymbol{x}_5)\} = 1.41$$

Tablica 2.4 Matrica udaljenosti u trećoj iteraciji

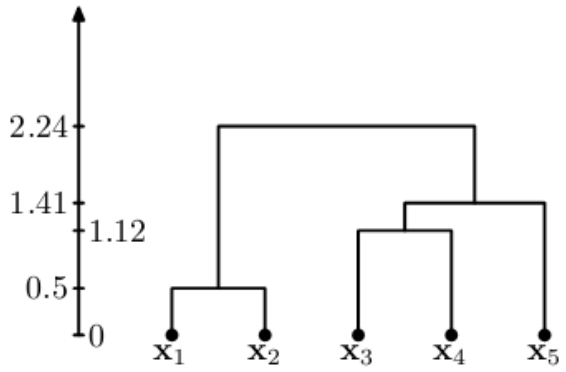
	$\{x_1, x_2\}$	$\{x_3, x_4\}$	x_5
$\{x_1, x_2\}$	0	2.24	3
$\{x_3, x_4\}$	2.24	0	1.41
x_5	3	1.41	0

Nakon izračuna dobije se nova matrica udaljenosti prikazana u tablici 2.4. U trećoj iteraciji spajaju se grozdovi $\{\boldsymbol{x}_3, \boldsymbol{x}_4\}$ i $\boldsymbol{x}_5$, jer sada oni imaju najmanju udaljenosti u matrici udaljenosti, te se dobije nova matrica udaljenosti:

Tablica 2.5 Matrica udaljenosti u četvrtoj iteraciji

	$\{x_1, x_2\}$	$\{x_3, x_4, x_5\}$
$\{x_1, x_2\}$	0	2.24
$\{x_3, x_4, x_5\}$	2.24	0

Na kraju se u četvrtoj iteraciji preostala dva grozda spajaju u jedan grozd i algoritam je završen. Rezultat razvrstavanja primjenom metode jednostruke veze na pet točaka u dvodimenzionalnom prostoru i udaljenostima među grozdovima prikazan je na slici 2.8.



Slika 2.8 Graf nastao primjenom metode jednostruke veze na pet točaka u prostoru

2.4.2. Metoda potpune veze

Metoda potpune veze za računanje matrice udaljenosti koristi udaljenost između najdaljih susjeda. Isto kao i metoda jednostruke veze ona je invarijantna na monotone transformacije. Ako su C_i, C_j i C_k tri grozda tada se udaljenost između C_k i $C_i \cup C_j$ može izračunati prema Lance-Williams-ovoj jednadžbi [22]:

$$\begin{aligned} D\big(C_k, C_i \cup C_j\big) &= \frac{1}{2}D(C_k, C_i) + \frac{1}{2}D\big(C_k, C_j\big) + \frac{1}{2}\Big(D(C_k, C_i) - D\big(C_k, C_j\big)\Big) \\ &= max\big\{D(C_k, C_i), D\big(C_k, C_j\big)\big\} \end{aligned} \tag{2.46}$$

Iz jednadžbe (2.46) lako je zaključiti da vrijedi:

$$D\big(C_k, C_j\big) = max_{\boldsymbol{x}\in C_k, \boldsymbol{y}\in C_j} d(\boldsymbol{x}, \boldsymbol{y}) \tag{2.47}$$

gdje su C_k i C_j dva ne prazna i različita grozda koja se ne preklapaju, a $d(\boldsymbol{x}, \boldsymbol{y})$ je funkcija udaljenosti prema kojoj se računa matrica udaljenosti. Koristeći isti primjer s pet točaka kao i kod metode jednostruke veze dobije se isti graf, ali se razlikuje po dobivenim visinama, to jest udaljenostima između grozdova.

2.4.3. Metoda srednje vrijednosti grupe

Metoda srednje vrijednosti grupe [3] još se naziva i UPGMA (eng. Unweighted pair group method using arithmetic averages). U ovoj metodi udaljenost između dva grozda računa se kao srednja vrijednost udaljenosti između svih mogućih parova objekata koji se nalaze u oba grozda. Ako su C_i, C_j i C_k tri grozda tada se udaljenost između C_k i $C_i \cup C_j$ može izračunati prema Lance-Williams-ovoj jednadžbi [22]:

$$D\big(C_k, C_i \cup C_j\big) = \frac{|C_i|}{|C_i| + |C_j|} D(C_k, C_i) + \frac{|C_j|}{|C_i| + |C_j|} D\big(C_k, C_j\big) \tag{2.48}$$

Iz jednadžbe (2.48) lako je zaključiti da vrijedi:

$$D\big(C_k, C_j\big) = \frac{1}{|C_k||C_j|} \sum_{\boldsymbol{x} \in C_k, \boldsymbol{y} \in C_j} d(\boldsymbol{x}, \boldsymbol{y}) \tag{2.49}$$

gdje su C_k i C_j dva ne prazna i različita grozda koji se ne preklapaju, a $d(\boldsymbol{x}, \boldsymbol{y})$ je funkcija udaljenosti prema kojoj se računa matrica udaljenosti.

Primjenom metode srednje vrijednosti grupe nad istim skupom objekata kao i u prethodnim metodama dobije se isti graf, ali se razlikuje po dobivenim visinama, to jest udaljenostima između grozdova. Postoji i varijacija metode srednje vrijednosti grupe i zove se težinska metoda srednje vrijednosti grupe, koja se još naziva i WPGMA (eng. Weighted pair group method using arithmetic averages). Ako su C_i, C_j i C_k tri grozda tada se udaljenost između C_k i $C_i \cup C_j$ može izračunati prema Lance-Williams-ovoj jednadžbi [22]:

$$D\big(C_k, C_i \cup C_j\big) = \frac{1}{2} D(C_k, C_i) + \frac{1}{2} D\big(C_k, C_j\big) \tag{2.50}$$

2.4.4. Metoda središnje točke

Metoda središnje točke [3] naziva se i UPGMUC (eng. Unweighted pair group method using centroids). Ako su C_i, C_j i C_k tri grozda tada se udaljenost između C_k i $C_i \cup C_j$ može izračunati prema Lance-Williams-ovoj jednadžbi [22]:

$$D(C_k, C_i \cup C_j) = \frac{|C_i|}{|C_i| + |C_j|} D(C_k, C_i) + \frac{|C_j|}{|C_i| + |C_j|} D(C_k, C_j) - \frac{|C_i||C_j|}{(|C_i| + |C_j|)^2} D(C_i, C_j) \tag{2.51}$$

Iz jednadžbe (2.52) lako je zaključiti da vrijedi:

$$D(C_k, C_j) = \frac{1}{|C_k||C_j|} \sum_{\boldsymbol{x} \in C_k, \boldsymbol{y} \in C_j} d(\boldsymbol{x}, \boldsymbol{y}) - \frac{1}{2|C_k|^2} \sum_{\boldsymbol{x}, \boldsymbol{y} \in C_k} d(\boldsymbol{x}, \boldsymbol{y}) - \frac{1}{2|C_j|^2} \sum_{\boldsymbol{x}, \boldsymbol{y} \in C_j} d(\boldsymbol{x}, \boldsymbol{y}) \tag{2.52}$$

gdje su C_k i C_j dva ne prazna i različita grozda koja se ne preklapaju, a $d(\boldsymbol{x}, \boldsymbol{y})$ je funkcija udaljenosti prema kojoj se računa matrica udaljenosti.

2.4.5. Metoda srednje vrijednosti

Metoda srednje vrijednosti [3] naziva i WPGMUC (eng. Weighted pair group method using centroids). Ako je kod metode središnje točke jedan grozd puno veći od drugog, tada će središte novog grozda koji je nastao spajanjem biti jako blizu središta većeg grozda, jer on ima puno veći utjecaj. U metodi srednje vrijednosti središte novog grozda je neovisno o veličini grozdova koji tvore novi grozd. Nedostatak metode srednje vrijednosti je u tome što nije pogodna za računanje korelacije zbog nemogućnosti interpretacije u geometrijskom smislu. Ako su C_i, C_j i C_k tri grozda tada se udaljenost između C_k i $C_i \cup C_j$ može izračunati prema sljedećoj jednadžbi:

$$D(C_k, C_i \cup C_j) = \frac{1}{2} D(C_k, C_i) + \frac{1}{2} D(C_k, C_j) - \frac{1}{4} D(C_i, C_j) \tag{2.53}$$

2.5. Razdvajajući hijerarhijski algoritmi

Razdvajajući hijerarhijski algoritmi rade na suprotnom principu nego aglomeracijski algoritmi. Na početku se svi objekti nalaze u jednome grozdu i postupno se razdvajaju na više grozdova. U svakoj iteraciji broj grozdova se uveća za jedan, tako da se postojeći grozd podijeli na dva nova grozda prema određenom kriteriju. Razdvajajući hijerarhijski algoritmi dijele se na dva tipa algoritama [24]. Prvi tip algoritama razdvaja objekte prema vrijednostima samo jednog atributa koji opisuje objekt, dok drugi tip algoritama razdvaja objekte prema vrijednostima svih atributa koji opisuju objekt. Nije moguće nabrojati sve moguće načine na koji se jedan grozd može podijeliti na dva dijela, jer postoji $2^{|C|-1}-1$ različitih načina na koji se grozd C može podijeliti na dva grozda A i B [25]. Još jedan problem razdvajajućih algoritama je kako postići monotonost, to jest da se grozdovi mogu dijeliti samo prema jednom atributu. U svakoj iteraciji razdvaja se samo jedan grozd i pitanje je koji grozd se treba razdvojiti u idućoj iteraciji? Razine moraju biti tako definirane da grozdovi u nižim razinama ne budu veći nego neki grozdovi u višim razinama. Zbog toga su razvijeni algoritmi koji ne moraju razmatrati sve moguće načine na koji se grozd može podijeliti u dva dijela te su također i monotoni. U idućim poglavljima opisani su najpoznatiji algoritmi.

2.5.1. Metoda DIANA

Metoda DIANA (eng. DIvisive ANAlysis) je razdvajajuća hijerarhijska metoda koja je predstavljena u [26]. Ona se može primijeniti na sve skupove objekata nad kojima je moguće primijeniti i neku od aglomeracijskih metoda. To znači ako se neki skup objekata može razvrstati pomoću aglomeracijskih metoda tada se može razvrstati i pomoću metode DIANA. Metoda započinje nizom uzastopnih podjela i u svakoj iteraciji razdvaja se najveći grozd. To je grozd koji ima najveći promjer, pa se može zaključiti da u njemu postoji najveća različitost između objekata. Razdvajanje se ponavlja $n-1$ puta, to jest sve dok se početni grozd ne razdvoji na pojedinačne objekte. Za neki grozd C njegov promjer se može izračunati prema sljedećoj jednadžbi:

$$Prom(C) = max_{\boldsymbol{x},\boldsymbol{y}\in C} d(\boldsymbol{x},\boldsymbol{y}) \tag{2.54}$$

Dobivene vrijednosti promjera kasnije se koriste za predstavljanje visina u grafu. U svakoj iteraciji grozd C razdvaja se na dva nova grozda A i B za koje vrijedi $A \cap B = \Phi$ (prazan skup) te $A \cup B = C$. U početku je postavljeno da je $A = C$ i $B = \Phi$, zatim se iterativno neki objekti prebacuju iz grozda A u grozd B. U prvoj iteraciji objekt $\boldsymbol{y}_1$ će se prebaciti iz grozda A u grozd B ako daje maksimum sljedeće funkcije:

$$D(\boldsymbol{x}, A\backslash\{\boldsymbol{x}\}) = \frac{1}{|A|-1} \sum_{\boldsymbol{y}\in A, \boldsymbol{y}\neq \boldsymbol{x}} d(\boldsymbol{x}, \boldsymbol{y}) \tag{2.55}$$

Nakon toga upisuju se nova stanja u grozdove A i B gdje se objekt $\boldsymbol{y}_1$ izbacuje iz grozda A i dodaje u grozd B. U sljedećoj iteraciji traže se novi objekti koji će se prebaciti iz grozda A u grozd B ako daju maksimum sljedeće funkcije:

$$D(\boldsymbol{x}, A\backslash\{\boldsymbol{x}\}) - D(\boldsymbol{x}, B) = \frac{1}{|A|-1} \sum_{\boldsymbol{y}\in A, \boldsymbol{y}\neq \boldsymbol{x}} d(\boldsymbol{x}, \boldsymbol{y}) - \frac{1}{|B|} \sum_{\boldsymbol{z}\in B} d(\boldsymbol{x}, \boldsymbol{z}) \tag{2.56}$$

Iteracije se nastavljaju sve dok je maksimalna vrijednost gornje funkcije (2.56) pozitivna, a zaustavlja se ako je vrijednost negativna ili jednaka nuli, što znači da je dijeljenje grozda C završeno. Nedostatak ove metode je što kao promjer koristi najveću razliku između dva objekta u grozdu pa se mogu pojaviti mala odstupanja pri razvrstavanju.

2.5.2. Metoda DISMEA

Metoda DISMEA predstavljena je u [27] i koristi algoritam k-means za dijeljenje jednog grozda na dva dijela. Algoritam kreće sa cijelim skupom objekata, i u svakoj iteraciji se razdvaja grozd koji ima najveću sumu kvadratnih udaljenosti. Razdvajanje se nastavlja sve dok se svaki grozd na kraju ne sastoji od jednog objekta. Početni grozd C koji se sastoji od n objekata, u prvoj iteraciji se dijeli na dva grozda C_1 i C_2 za koje vrijedi $C_1 \neq \Phi$, $C_2 \neq \Phi$, $C_1 \cap C_2 = \Phi$ i $C_1 \cup C_2 = \mathrm{C}$. Dijeljenje se obavlja tako da daje maksimum sljedeće funkcije:

$$F(C_1, C_2; C) = \sum_{i=1}^{2} \sum_{\boldsymbol{x}\in C_i} \|\boldsymbol{x} - \mu(C_i)\|^2 \tag{2.57}$$

gdje je $\mu(C_i)$ središte grozda C_i izračunato prema sljedećoj jednadžbi:

$$\mu(C_i) = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x} \tag{2.58}$$

U svakoj sljedećoj iteraciji razdvaja se onaj grozd koji ima najveću sumu kvadratnih udaljenosti izračunatu prema sljedećoj jednadžbi:

$$E(C) = \sum_{\boldsymbol{x} \in C} \|\boldsymbol{x} - \mu(C)\|^2 \tag{2.59}$$

Jedan od načina kako pronaći najbolju podjelu je proći kroz sve kombinacije i provjeriti iznos sume kvadratnih udaljenosti. Ali to je nepraktično zbog velikog broja kombinacija, stoga DISMEA koristi k-means algoritam za podjelu grozda na dva dijela.

2.6. Algoritmi oko središnje točke

Grozdovi koji nastaju razvrstavanjem oko središnje točke su konveksnog oblika i predstavljaju se pomoću središnje točke. Ovaj tip algoritama nije pogodan za razvrstavanje grozdova koji su nepravilnog oblika. Svim algoritmima ovog tipa potrebno je navesti početne parametre kao što su objekti, broj grozdova i ciljna funkcija. Ciljna funkcija određuje točnost razvrstavanja i algoritam se ponavlja sve dok se ne zadovolji ciljna funkcija. Ciljna funkcija je najčešće predstavljena minimalnim pomakom središta grozdova u odnosu na središta iz prethodne iteracije. Algoritmi oko središnje točke imaju veću učinkovitost nego hijerarhijski algoritmi pa se koriste na velikim bazama podataka i velikim dimenzijama. U idućim poglavljima objašnjeni su najčešće korišteni algoritmi, te njihove prednosti i nedostaci.

2.6.1. Algoritam k-means

Algoritam k-means opisan je u [28] i razvijen je u svrhu razvrstavanja objekata s brojčanim podacima. Grozd je predstavljen pomoću središnje točke i kod ovog algoritma broj grozdova k je unaprijed poznat i ne mijenja se tijekom svih iteracija. Također mora biti

poznata funkcija pogreške to jest ciljna funkcija koja određuje kraj razvrstavanja i prekid iteracija. Algoritam započinje odabirom početnih središta grozdova i dodjelom objekata najbližem grozdu. Nakon toga računaju se nova središta grozdova kao srednja vrijednost svih objekata pridruženih grozdu. U idućoj iteraciji ponovno se obavlja dodjeljivanje objekata novim središtima grozdova. Proces se ponavlja sve dok se ne zadovolji ciljna funkcija (funkcija pogreške) ili dok objekti ne prestanu mijenjati pripadnost pojedinom grozdu. Funkcija pogreške računa se prema sljedećoj jednadžbi:

$$E = \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in C_i} d(\boldsymbol{x}, \mu(C_i)) \tag{2.60}$$

gdje je k broj grozdova, $\mu(C_i)$ središte grozda C_i, a $d(\boldsymbol{x}, \mu(C_i))$ predstavlja udaljenost između objekta $\boldsymbol{x}$ i središta grozda.

Kao funkcija udaljenosti najčešće se koristi euklidska udaljenost, a mogu se koristiti i druge udaljenosti koje su opisane u prethodnim poglavljima. Pseudo kod za algoritam k-means može se pronaći u [29]. Algoritam se može podijeliti na inicijalizacijsku i iterativnu fazu. U inicijalizacijskoj fazi algoritam slučajnim odabirom raspodjeljuje objekte u k grozdova. U iterativnoj fazi računaju se udaljenosti između objekata i grozdova, i objekti se dodjeljuju najbližem grozdu. Bitno je napomenuti da se računaju udaljenosti od svakog objekta do svakog grozda. Cilj je minimizirati ciljnu funkciju prema određenim uvjetima. Za skup objekata $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$ ciljna funkcija može se opisati sljedećom jednadžbom:

$$P(W, Q) = \sum_{l=1}^{k} \sum_{i=1}^{n} w_{il} d_{euk}(\boldsymbol{x}_i, \boldsymbol{q}_l) \tag{2.61}$$

gdje je $Q = \{\boldsymbol{q}_l, l = 1,2,\dots,k\}$ skup grozdova, a W je matrica veličine $n \times k$ koja zadovoljava sljedeće uvjete:

1. $w_{il} \in \{0,1\} \;\; i = 1,2,\dots,n, \;\; l = 1,2,\dots,k$
2. $\sum_{l=1}^{k} w_{il} = 1 \;\; i = 1,2,\dots,n$

Algoritam k-means može se rješavati kao i optimizacijski problem [30], gdje se minimizira jednadžba (2.61) prema uvjetima jedan i dva. Ona se rješava tako da se rastavi na dva podproblema [31]:

1. Pod problem P1 - postaviti $Q = \hat{Q}$ i riješiti reducirani problem $P(W, \hat{Q})$.
2. Pod problem P2 – postaviti $W = \widehat{W}$ i riješiti reducirani problem $P(\widehat{W}, Q)$.

Za rješavanje pod problema postavljena su dva teorema.

Teorem 1: Postavlja se fiksna vrijednost za $\hat{Q} = \{\hat{\boldsymbol{l}}_l\ ,\ l = 1,2,\ldots,k\}$. Tada je funkcija $P(W, \hat{Q})$ minimizirana ako vrijedi:

$$w_{il} = \begin{cases} 1 & ako\ je\ d_{euk}(\boldsymbol{x}_i, \hat{\boldsymbol{q}}_l) = min_{l \le t \le k} d_{euk}(\boldsymbol{x}_i, \hat{\boldsymbol{q}}_t) \\ 0 & za\ sve\ ostalo \end{cases}$$

Teorem 2: Postavlja se fiksna vrijednost za $\widehat{W}$. Tada je funkcija $P(\widehat{W}, Q)$ minimizirana ako vrijedi:

$$\boldsymbol{q}_{lj} = \frac{\sum_{i=1}^{n} \widehat{w}_{il} \boldsymbol{x}_{ij}}{\sum_{i=1}^{n} \widehat{w}_{il}}$$

gdje je $l = 1,2,\ldots,k,\ j = 1,2,\ldots,d,\ i = 1,2,\ldots,n$

Vremenska složenost algoritma je $O(nkd)$ po jednoj iteraciji, gdje je d broj dimenzija, k je broj grozdova i n je broj objekata u skupu objekata. Budući da se slijed $P(\cdot;\cdot)$ smanjuje on će nakon konačnog broja iteracija konvergirati u lokalnu minimalnu točku. Algoritam k-means ima sljedeća svojstva:

1. Nije učinkovit kod velikog skupa objekata, jer je njegova složenost linearno proporcionalna veličini skupa objekata.
2. Često se prekida izvođenje kada se postigne lokalni optimum.
3. Grozdovi su konveksnog oblika.
4. Koristi se za brojčane podatke.
5. Performanse su ovisne o početnom odabiru središta grozdova.

Algoritam k-means ovisan je o odabiru središta grozdova pa su razvijene metode koje predlažu dobre izbore za početna središta grozdova [32]. Koriste se četiri inicijalizacijske

metode: metoda slučajnog odabira, Forgyev pristup, Macqueenov pristup i Kaufmanov pristup. Navedene i druge inicijalizacijske metode opisane su u [33]. Algoritam predstavljen u ovom poglavlju naziva se standardni k-means algoritam. Postoje različite varijacije koje će biti predstavljene u idućim poglavljima, a detaljnije se može pogledati u [34].

2.6.2. Neprekidni k-means algoritam

Neprekidni k-means [35] je brži od standardnog k-means algoritma, jer on početna središta grozdova odabire uniformnim odabirom iz čitavog skupa objekata, dok standardni k-means središta grozdova uzima proizvoljno. Nadalje neprekidni k-means u svakoj iteraciji uzima uzorke objekata, dok standardni k-means uzima sve objekte u slijedu. Slučajno uzorkovanje objekata predstavlja izvorni Macqueenov koncept koji je korišten za razvrstavanje u neprekidnom prostoru. Prema Macqueenu funkcija pogreške E_i za svako područje R_i je dana sljedećom jednadžbom:

$$E_i = \int \rho(\boldsymbol{x}) \|\boldsymbol{x} - \boldsymbol{z_i}\|^2 d\boldsymbol{x} \tag{2.62}$$

gdje je $\rho(\boldsymbol{x})$ funkcija gustoće vjerojatnosti.

To je neprekidna funkcija nad područjem R_i sa središtem u $\boldsymbol{z_i}$. Zbrajanjem svih funkcija pogrešaka E_i dobije se ukupna pogreška područja skupa objekata. Slučajnim uzorkovanjem skupa objekata može se estimirati funkcija gustoće vjerojatnosti i na taj način dobiti funkciju pogreške bez korištenja svih objekata iz skupa objekata. Zbog korištenja manjeg broja objekata neprekidni k-means algoritam je brži od standardnog algoritma.

2.6.3. Usporedni k-means algoritam

Usporedni k-means algoritam [36] koristi jednostavan pristup kako bi se izbjegle mnoge usporedbe i ubrzao k-means algoritam. Ako je $\boldsymbol{x}$ objekt iz skupa objekata D, a μ_i i μ_j dvije srednje vrijednosti (središta) tada pomoću nejednakosti trokuta možemo izračunati:

$$d(\boldsymbol{x}, \mu_i) + d(\boldsymbol{x}, \mu_j) \geq d(\mu_i, \mu_j) \tag{2.63}$$

Ako je zadovoljeno $d(\boldsymbol{x}, \mu_i) \geq d(\mu_i, \mu_j) - d(\boldsymbol{x}, \mu_j)$ i ako je zadovoljen sljedeći uvjet $d(\mu_i, \mu_j) \geq 2d(\boldsymbol{x}, \mu_i)$ može se zaključiti da vrijedi $d(\boldsymbol{x}, \mu_j) \geq d(\boldsymbol{x}, \mu_i)$. U tome slučaju izbjegnuto je računanje udaljenosti $d(\boldsymbol{x}, \mu_j)$. Budući da broj grozdova obično nije velik udaljenosti između svih parova srednjih vrijednosti lako se izračunaju prije svake iteracije. Znači da se prije usporedbe objekta $\boldsymbol{x}$ sa središtem μ_j provede provjera iz jednadžbe (2.63) koristeći najbliže središte podatku $\boldsymbol{x}$. Broj usporedbi koje napravi gornji algoritam ograničen je sa sljedećom vrijednosti $k^2 + nkd$, gdje je d broj dimenzija, k je broj grozdova i n je broj objekata.

2.6.4. Sortni k-means algoritam

Sortni k-means algoritam [36] je dodatak usporednom k-means algoritmu. Ovaj algoritam sortira središta grozdova po udaljenosti od svakog središta posebno, kako bi se postiglo veće ubrzanje. Ako je $D_{ij} = d(\mu_i, \mu_j)$ za $i.j = 1,2, \dots m_{i1}$, gdje je μ_i središte i-tog grozda tada se matrica M sastoji od $k \times k$ dimenzija u kojoj reci $(m_{i1}, m_{i2}, \dots, m_{ik})$ predstavljaju permutacije od 1 do k za koje vrijedi $d(\mu_i, \mu_{mi1}) \leq d(\mu_i, \mu_{mi2}) \leq d(\mu_i, \mu_{mik})$, budući da su udaljenosti sortirane. Vremenska složenost algoritma je $O(ndy + k^2 d + k^2 logk)$, gdje je d broj dimenzija, k je broj grozdova, n je broj objekata i γ je prosječan broj središta, računajući za sve objekte $\boldsymbol{x}$, koji nisu duplo udaljeniji od objekta $\boldsymbol{x}$ nego što je to središte koje je objektu $\boldsymbol{x}$ bilo dodijeljeno u prethodnoj iteraciji.

2.6.5. Algoritam x-means

Za razliku od algoritma k-means gdje je broj grozdova k poznat od samog početka i predaje se kao ulazni parametar algoritma, algoritam x-means [37] koristi Bayesov informacijski kriterij BIC za određivanje najboljeg broja grozdova. BIC kriterij za neki skup objekata $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$ koristi zamjenske modele $M_j = \{C_1, C_2, \dots, C_k\}$. Svaki model odgovara različitom broju grozdova k. Za svaki model računaju se vjerojatnosti $P(M_j|D)$ koje govore koliko je model dobar. BIC to jest Schwartzov kriterij je opisan jednadžbom:

$$BIC(M_j) = \hat{l}_j(D) - \frac{p_j}{2} logn \tag{2.64}$$

gdje je $\hat{l}_j(D)$ vjerojatnost skupa objekata D prema j-tom modelu u točki najveće vjerojatnosti, p_j je broj parametara u M_j. Kao najbolji uzima se model s najvećom vjerojatnosti. Za Gaussovu razdiobu procjena vrijednosti varijance dana je sljedećom jednadžbom:

$$\hat{\sigma}^2 = \frac{1}{n-k}\sum_{i=1}^{n}(\boldsymbol{x}_i - \mu_{(i)})^2 \tag{2.65}$$

gdje je $\mu_{(i)}$ središte objekta $\boldsymbol{x}_i$, a (i) označava indeks središta koje je najbliže objektu $\boldsymbol{x}_i$. Vjerojatnost za pojedini objekt može se izračunati pomoću sljedeće jednadžbe:

$$\hat{P}(\boldsymbol{x}_i) = \frac{|C_{(i)}|}{n}\frac{1}{\sqrt{2\pi}\hat{\sigma}^d}\exp\left(\frac{1}{2\hat{\sigma}^2}\left\|\boldsymbol{x}_i - \mu_{(i)}\right\|^2\right) \tag{2.66}$$

Slijedi da je vjerojatnost svih objekata:

$$\begin{aligned} l(D) &= log\prod_{i=1}^{n}P(\boldsymbol{x}_i) \\ &= \sum_{i=1}^{n}\left(log\frac{1}{\sqrt{2\pi}\hat{\sigma}^d} - \frac{1}{2\hat{\sigma}^2}\left\|\boldsymbol{x}_i - \mu_{(i)}\right\|^2 + log\frac{|C_{(i)}|}{n}\right) \end{aligned} \tag{2.67}$$

gdje je broj parametara $p_j = (d+1)k$.

BIC kriterij korišten je globalno za određivanje najboljeg modela, a lokalno za razdvajanje grozdova. Izvođenje algoritma može se opisati na sljedeći način. Prvo se zadaje raspon broja grozdova k, to jest najmanji i najveći broj grozdova, a zatim algoritam kreće s najmanjim brojem grozdova i dodaje nove grozdove sve dok se ne dosegne najveći broj grozdova. Novi grozd dodaje se tako da se jedan grozd razdvoji na dva nova grozda prema BIC kriteriju. Na kraju procesa uzima se skup grozdova s najboljim svojstvima.

2.6.6. Harmonični k-means algoritam

Harmonični k-means algoritam [38] nastao je iz algoritma k-means. Glavno svojstvo mu je da na njega ne utječe početni izbor središta grozdova. Funkcija pogreške za algoritam k-means može se izračunati prema sljedećoj formuli:

$$E = \sum_{i=1}^{n} \min\{d(\boldsymbol{x}, \mu_j), \quad j = 1,2, \dots k\} \tag{2.68}$$

gdje je μ_j središte j-tog grozda.

Iz jednadžbe (2.68) slijedi da se funkcija pogreške za harmonični k-means algoritam može izračunati ako zamijenimo funkciju minimuma s funkcijom harmonijskog prosjeka. Koristeći kvadratnu euklidsku udaljenost dobije se:

$$E = \sum_{i=1}^{n} \mathrm{HA}\left(\{d_{kveuk}(\boldsymbol{x}, \mu_j), \quad j = 1,2, \dots k\}\right) = \sum_{i=1}^{n} \frac{k}{\sum_{j=1}^{k} \frac{1}{(\boldsymbol{x} - \mu_j)^T (\boldsymbol{x} - \mu_j)}} \tag{2.69}$$

gdje je $d_{kveuk}\left(\boldsymbol{x}, \mu_j\right)$ kvadratna euklidska udaljenost, i HA() je harmonijski prosjek opisan sljedećom jednadžbom:

$$\mathrm{HA}(\{a_i : i = 1,2, \dots m\}) = \frac{m}{\sum_{i=1}^{m} a_i^{-1}} \tag{2.70}$$

Ako se funkcija pogreške djelomično derivira s obzirom na središta $\mu_l, \quad l = 1,2, \dots k$ i postavljajući ih na nulu, dobije se rekurzivna jednadžba za harmonični k-means algoritam:

$$\frac{\partial E}{\partial \mu_l} = -k \sum_{i=1}^{n} \frac{2(\boldsymbol{x} - \mu_l)}{d_{il}^4 \left(\sum_{j=1}^{k} d_{ij}^{-2}\right)^2} \tag{2.71}$$

gdje je $d_{ij} = d_{euk}(\boldsymbol{x}, \mu_j)$.

Rješavanjem jednadžbe (2.72) dobiju se nova središta grozdova:

$$\mu_l^* = \frac{\sum_{i=1}^{n} d_{il}^4 \left(\sum_{j=1}^{k} d_{ij}^{-2}\right)^{-2} \boldsymbol{x}_i}{\sum_{i=1}^{n} d_{il}^4 \left(\sum_{j=1}^{k} d_{ij}^{-2}\right)^{-2}} \qquad l = 1,2,\dots,k \tag{2.72}$$

Ako se u jednadžbu (2.72) uvrste početna središta grozdova dobiju se nova središta, i funkcija se rekurzivno ponavlja dok se središta ne stabiliziraju.

2.6.7. Algoritam k-vjerojatnosti

Algoritam k-vjerojatnosti [10] može se koristiti za razvrstavanje mješovitih tipova objekata. On koristi opći koeficijent udaljenosti između dva objekta. Udaljenost d_{pj} između objekta i grozda p opisana je sljedećom jednadžbom:

$$d_{ip}^2 = \sum_k \frac{w_{ipk}\left(x_{ik} - \mu_{pk}\right)^2}{\sum_k w_{ipk}} \tag{2.73}$$

gdje je x_{ik} vrijednost k-te varijable za objekt i, μ_{pk} je srednja vrijednost k-te varijable za grozd p, w_{ipk} je težina koja ima vrijednost 1 ili 0 ovisno o tome postoji li usporedba između objekta i grozda za k-tu varijablu. Vrijednosti su $w_{ipk} = 1$ ako postoji usporedba i $w_{ipk} = 0$ ako ona ne postoji. Srednja vrijednost μ_{pk} je vektor φ_{pks} vjerojatnosti za svako stanje s od k-te varijable unutar grozda p. Ciljna funkcija algoritma opisana je sljedećom jednadžbom:

$$E = \sum_p E_p \tag{2.74}$$

gdje je E_p zbroj euklidskih kvadratnih udaljenosti opisan sljedećom jednadžbom:

$$E_p = \sum_{i \in p} n_i \sum_k \frac{w_{ipk}\left(x_{ik} - \mu_{pk}\right)^2}{\sum_k w_k} \tag{2.75}$$

gdje je n_i diferencijalna težina za objekt i koja je obično jednaka 1, w_k je diferencijalna težina za k-ti atribut, gdje je $w_k = 0$ ako x_{ik} ili μ_{pk} nemaju vrijednost za k-ti atribut.

Cilj je minimizirati sumu kvadrata euklidskih udaljenost. Algoritam kreće s inicijalnom podjelom skupa objekata na k grozdova i u sljedećim iteracijama raspodjeljuje objekte minimizirajući ukupnu vrijednost kvadrata euklidskih udaljenosti. Objekt se premješta iz grozda p u grozd q ako vrijedi:

$$E_p + E_q > E_{p-1} + E_{q+1} \tag{2.76}$$

2.6.8. Algoritam k-prototipa

Algoritam k-prototipa [31] se koristi za razvrstavanje mješovitih tipova objekata. U ovom algoritmu prototip predstavlja središte grozda. Ako su za dva mješovita objekta prvih p atributa brojčani, a ostalih $m - p$ atributa kategorički tada se mjera različitosti može opisati sljedećom jednadžbom:

$$d(X,Y) = \sum_{j=1}^{p}(x_j + y_j)^2 + \gamma \sum_{j=p+1}^{m} \delta(x_j, y_j) \tag{2.77}$$

gdje je γ balansna težina koja se koristi za izbjegavanje favoriziranja bilo kojeg tipa atributa.

Euklidska udaljenost je korištena za mjerenje udaljenosti između brojčanih podataka, a jednostavna usporedba razlike za mjerenje kategoričkih podataka [39]. Cilj algoritma je minimiziranje sljedeće funkcije troška:

$$P(W,Q) = \sum_{l=1}^{k}(P_l^r + \gamma P_l^c) \tag{2.78}$$

gdje je:

$$P_l^r = \sum_{l=1}^{n} w_{i,l} \sum_{j=1}^{p}(x_{i,j} - q_{l,j})^2 \tag{2.79}$$

$$P_l^c = \sum_{l=1}^{n} w_{i,l} \sum_{j=p+1}^{m} \delta(x_{i,j}, q_{l,j}) \tag{2.80}$$

Algoritam k-prototipa je isti kao algoritam k-vjerojatnosti osim u fazi premještanja objekata. Složenost algoritma k-prototipa je $O((t+1)kn)$, gdje je n broj objekata, k broj grozdova, a t je broj iteracija u procesu premještanja objekata u druge grozdove.

2.7. Ostali modeli algoritama za razvrstavanje

U ovom poglavlju opisane su različiti modeli algoritama za razvrstavanje. Jedan od modela su algoritmi razvrstavanja prema gustoći (eng. Density-based). Ovaj model algoritama ima mogućnost pronalaska grozdova koji imaju nepravilne oblike. Grozdovi su područja s velikom gustoćom objekata, a između njih nalaze se područja male gustoće. Ovakav tip algoritama samo jednom pregledava čitavo područje skupa objekata i nema iteracija. Nadalje, broj grozdova nije poznat unaprijed, jer se broj grozdova određuje pregledom područja skupa objekata i određuje ovisno o gustoći objekata u području.

Primjer takvog algoritma je DBSCAN (eng. Density-Based Spatial Clustering of Applications with Noise) koji može detektirati grozdove nepravilnog oblika [40]. Algoritam koristi koncept zvan ϵ-susjedstvo jednog objekta. Ako se neki podatak predstavi s $\boldsymbol{x}$ njegovo se ϵ -susjedstvo označava s $N_\epsilon(\boldsymbol{x})$ i opisano je sljedećom jednadžbom:

$$N_\epsilon(\boldsymbol{x}) = \{\boldsymbol{y} \in D : d(\boldsymbol{x}, \boldsymbol{y}) \leq \epsilon\} \tag{2.81}$$

gdje je D skup objekata i $d(\boldsymbol{x}, \boldsymbol{y})$ je jedna od funkcija udaljenosti. Za objekt $\boldsymbol{x}$ može se reći da je izravno dostupan preko gustoće od točke $\boldsymbol{y}$ s obzirom na ϵ i N_{min} ako vrijedi:

1. $\boldsymbol{x} \in N_\epsilon(\boldsymbol{y})$

2. $|N_\epsilon(\boldsymbol{y})| \geq N_{min}$ gdje je $|\mathrm{N}_\epsilon(\mathbf{y})|$ broj podataka u $\mathrm{N}_\epsilon(\mathbf{y})$

Izravni pristup preko gustoće je simetričan za dva objekta koji se nalaze u istom grozdu, ali općenito nije simetričan za objekt unutar grozda i objekt na granici grozda. Objekt $\boldsymbol{x}$ je dostupan preko gustoće od objekta $\boldsymbol{y}$ ako postoji skup objekata $\boldsymbol{x} = \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_i, = \boldsymbol{y}$ takav da je $\boldsymbol{x}_l$ izravno dostupan preko gustoće objekta $\boldsymbol{x}_{l+1}$ za svaki $l = 1,2, \ldots, i-1$. Za razliku od dostupnosti povezanost je simetrična i za objekt unutar grozda i objekt na granici grozda.

Dva objekta $\boldsymbol{x}$ i $\boldsymbol{y}$ su povezana preko gustoće ako postoji objekt $\boldsymbol{z}$ od kojeg su oba objekta $\boldsymbol{x}$ i $\boldsymbol{y}$ dostupni preko gustoće. Algoritam DBSCAN kreće od jednog objekta $\boldsymbol{x}$ i pridružuje mu sve objekte koji su mu dostupni preko gustoće. Ako je $\boldsymbol{x}$ objekt unutar grozda on se uspješno stvara, no ako je objekt s ruba grozda tada nema objekata koji su dostupni preko gustoće i algoritam prelazi na idući objekt koji nije razvrstan. DBSCAN ima mogućnost spojiti dva grozda u jedan ako se nalaze blizu jedan drugome. Algoritam kao ulaz zahtijeva parametre ϵ i N_{min} koji su isti za sve grozdove, stoga nije lagano izabrati dobre vrijednosti za navedene parametre. Međutim, iskustveno se ti parametri određuju prema „najtanjem“ grozdu u skupu objekata i nazivaju se graf k sortiranih udaljenosti [40].

Drugi algoritam ovog tipa je BRIDGE. To je hibridni algoritam koji je nastao kombinacijom k-means i DBSCAN algoritma [41]. Može razvrstavati velike skupine objekata, a istodobno poboljšava k-means algoritam uklanjanjem šuma, jer je algoritam DBSCAN otporan na šum. U BRIDGE algoritmu najprije se koristi k-means za podjelu skupa objekata na k grozdova, a zatim se primjenjuje DBSCAN nad svakim dijelom posebno za pronalazak grozdova po gustoći. Na kraju se za poboljšanje k-means grozdova i uklanjanje šuma koristi DBSCAN. U najpoznatije algoritme razvrstavanja prema gustoći također pripadaju DBCLASD (eng. Distribution Based Clustering of Large Spatial Databases), DENCLUE (eng. DENsity based CLUstEring) o kojima se detaljnije može pročitati u [12].

Također postoje i modeli algoritama koji su temeljeni na grafovima. Ovi algoritmi najprije naprave graf, a nakon toga primjenjuje se neki od algoritama za razvrstavanje na pojedine dijelove grafa. Najpoznatiji algoritmi ovog tipa su kameleon algoritam i ROCK algoritam.

Sljedeći model algoritama su algoritmi koji koriste grid. Oni znatno smanjuju vremensku složenost pogotovo kod velikih skupova objekata. Njihova prednost je što ne promatraju pojedine objekte nego prostor koji okružuje više objekata. Općenito se sastoje od sljedećih koraka [42]:

1. Stvaranje grid strukture, to jest podjela područja skupa objekata u ćelije.
2. Računanje gustoće za svaku ćeliju.
3. Sortiranje ćelija prema gustoći.
4. Pronalazak središta grozda.
5. Obuhvaćanje susjednih ćelija.

Najpoznatiji predstavnik grid algoritama je STING (eng. STatistical INformation Grid-based clustering method). Objekti su podijeljeni u pravokutne ćelije koje su hijerarhijski predstavljene. Ako je korijen na razini 1, tada su njegovi potomci na razini 2 i tako dalje. Ćelija na razini i nastaje unijom svojih ćelija potomaka na razini $i+1$. U STING algoritmu svaka ćelija ima četiri potomka i svaki potomak predstavlja jedan od četiri kvadranta roditeljske ćelije. Algoritam nije pogodan za veliki broj dimenzija jer je vremenska složenost algoritma $O(k)$, gdje je k broj ćelija na zadnjoj razini. Svaka ćelija ima četiri potomka pa slijedi da je broj ćelija na drugoj razini 2^d, gdje je d broj dimenzija. Ostali predstavnici ovog tipa algoritama su OPTIGrid, GRIDCLUS i GDILC, o kojima se detaljnije može vidjeti u [12].

Svi do sada navedeni algoritmi za razvrstavanje pripadali su jednoznačnim algoritmima. Za razliku od jednoznačnih algoritama za razvrstavanje gdje svaki objekt pripada samo jednom grozdu, u neizrazitim algoritmima objekt može pripadati u više grozdova. Neizraziti algoritmi svaki objekt povezuju sa svim grozdovima pomoću članske funkcije. Koncept neizrazitih skupova prvi put je uveden u [43], i od tada se neizrazito razvrstavanje koristi u različitim područjima. Za skup D koji se sastoji od n objekata od koji je svaki opisan s d atributa, i neka je c cijeli broj između jedan i n. Tada je neizrazita c-podjela definirana s matricom $U = (u_{li})$ veličine $c \times n$ koja zadovoljava sljedeće uvjete:

1. $u_{li} \in [0,1], \quad 1 \leq l \leq c, 1 \leq i \leq n$
2. $\sum_{l=1}^{c} u_{li} = 1, \;\; 1 \leq i \leq n$
3. $\sum_{i=1}^{n} u_{li} > 0, \;\; 1 \leq l \leq c$

gdje u_{li} označava stupanj pripadnosti objekta i grozdu l.

Za svaku neizrazitu c-ćeliju postoji odgovarajuća jednoznačna c -ćelija. Ako je u_{li} član bilo koje neizrazite c-ćelije tada je odgovarajuća jednoznačna c-ćelija od u_{li} opisana sljedećom jednadžbom:

$$u_{li} = \begin{cases} 1 \;\; ako\ je\ l = \arg max_{1 \leq j \leq c} u_{li} \\ 0 \quad za\ sve\ ostale\ slučajeve \end{cases} \tag{2.82}$$

Svaki neizraziti skup opisan je članskom funkcijom koja svakom objektu dodjeljuje stupanj pripadnosti u rasponu od 0 do 1. Kada je A običan skup objekata funkcija $f_A(x)$ može

poprimiti samo vrijednosti 0 ili 1, ovisno ne pripada ili pripada u A. Dok u neizrazitom skupu vrijednost $f_A(x)$ označava stupanj pripadnosti x u A. Za potrebe neizrazitog razvrstavanja modificirani su brojni algoritmi iz jednoznačnog razvrstavanja. Neizraziti *k*-means algoritam je modifikacija *k*-means algoritma prilagođena za neizrazito razvrstavanje [44]. Isto tako postoje neizrazite modifikacije i ostalih algoritma. Oni koriste isti princip kao kod jednoznačnog razvrstavanja koji je prilagođen za neizraziti skup objekata. Može se reći da postoje neizrazite modifikacije većine jednoznačnih algoritama za razvrstavanje.

3. Razvrstavanje objekata koji sadrže nesigurnost

Razvrstavanje objekata koji sadrže nesigurnost vrlo je zanimljivo i zahtjevno područje za istraživanje pa postoje brojna istraživanja u tome području. U ovoj disertaciji detaljno je opisano razvrstavanje objekata koji sadrže nesigurnost u svojim prostornim podacima. Nesigurnost objekata može proizići iz mnoštva različitih razloga kao što su nepreciznost u mjerenju, pogreške u uzorkovanju podataka ili zbog nemogućnosti redovitog osvježavanja prostornih podataka s novim vrijednostima pa se oni moraju aproksimirati. U većini slučajeva nije moguće u svakome trenutku znati točnu poziciju objekta, jer postoji određeno vremensko kašnjenje od zadnjeg izvještaja s točnom pozicijom ili samo mjerenje pozicije nije precizno. Također, objekti nisu statični i gibaju se u prostoru pa je njihova trenutna pozicija drugačija od pozicije u zadnjem izvještaju. Nova pozicija mora biti procijenjena s određenom nesigurnosti koristeći pretpostavke o smjeru i brzini pokretnog objekta. Zbog nesigurnosti pozicija objekta u ovom slučaju više nije točka u prostoru nego je predstavljena s funkcijom gustoće vjerojatnosti (eng. Probability density function-PDF). U ovoj disertaciji promatrani su objekti u 2D prostoru pa je i njihova nesigurnost dvodimenzionalna. Broj nesigurnih objekata u različitim primjenama može biti jako velik i podatke o njihovoj nesigurnosti potrebno je spremati u bazama podataka. Takve baze podataka zovu se nesigurne baze podataka, jer čuvaju informacije o objektima koji sadrže nesigurnost. Zbog velikog broja objekata njihovo razvrstavanje vremenski je vrlo zahtjevno pa su potrebni dobri algoritmi za razvrstavanje i velika računalna moć. Kao i kod algoritama sa sigurnim objektima, objekti su podijeljeni u grozdove prema sličnosti u svojim prostornim podacima. Slijedi da su objekti u jednom grozdu prostorno bliski i ta se činjenica može iskoristiti na mnoštvo načina. Sva djelovanja između objekata su lokalna, svi objekti teže središtu grozda, očuvanje energije je bolje, komunikacija je brža, smanjuje se promet, vrijeme reakcije je brže i tako dalje.

Kao što je ranije rečeno pozicija objekta nije predstavljena jednom točkom u prostoru nego minimalnim područjem nesigurnosti, koje se predstavlja pomoću PDF-a [45]. Uobičajeni algoritmi za razvrstavanje mogu razvrstavati samo one objekte koji su predstavljeni pomoću točke i nemaju mogućnost razvrstavati nesigurne objekte, jer su oni predstavljeni pomoću minimalnog područja nesigurnosti. Jedna od mogućnosti je nesigurne objekte predstaviti pomoću središnje točke i primijeniti uobičajeni algoritam za razvrstavanje. No u [46] je dokazano da razvrstavanje nesigurnih objekata koristeći minimalno pravokutno

područje nesigurnosti daje bolje rezultate, nego predstavljanje pomoću središnje točke i razvrstavanje uobičajenim algoritmima za razvrstavanje.

Pozicija objekta može biti nesigurna na dva različita načina. U prvom slučaju nije sigurno postoji li objekt na onome mjestu na kojem bi trebao biti, i to se zove egzistencijalna nesigurnost [47]. Takvi objekti se u bazama podataka spremaju sa svojim prostornim podacima i vjerojatnosti koja opisuje kolika je vjerojatnost postojanja takvog objekta [48]. U drugom slučaju sigurno je postojanje objekta na predviđenom mjestu, ali je točna vrijednost prostornih podataka nesigurna. Objekt se modelira pomoću minimalnog pravokutnog područja nesigurnosti koje ga okružuje (eng. Minimum bounding rectangle - MBR), i u njemu se nalaze sve moguće vrijednosti prostornih podataka za promatrani objekt [46][49]. Minimalno područje nesigurnosti dijeli se na uzorke i svakome uzorku dodjeljuje se vjerojatnost da se objekt nalazi na tome uzorku. Zbroj svih vjerojatnosti uzoraka mora biti jednak 1, i vjerojatnost da se objekt nalazi izvan minimalnog područja nesigurnosti jednaka je 0. U ovoj disertaciji proučava se drugi slučaj u kojem je sigurno postojanje objekta, ali njegovi prostorni podaci sadrže nesigurnost. Rezultati razvrstavanja mogu se koristiti za određivanje najvjerojatnijih vrijednosti podataka nekog objekta. U ovome slučaju radi se o određivanju najvjerojatnije pozicije koju ima nesigurni objekt. Na primjer, kod algoritama za razvrstavanje prema gustoći nastoje se odrediti područja s najvećom gustoćom nesigurnih objekata [50][51][52].

Cilj razvrstavanja je minimiziranje ciljne funkcije, kao što je na primjer minimiziranje ukupne kvadratne udaljenosti objekata do središta grozda [28]. Osim kvadratne euklidske udaljenosti mogu se koristiti i druge mjere za udaljenost kao što je Manhattan udaljenost, euklidska udaljenost ili Minkowskijeva udaljenost [53]. U većini algoritama za razvrstavanje nesigurnih objekata funkcija gustoće vjerojatnosti je predstavljena skupom uzoraka koji sadrže vjerojatnosti za svaku vrijednost unutar MBR-a. Zbog preciznosti potreban je veliki broj uzoraka za predstavljanja PDF-a svakog objekta što znatno povećava vrijeme izvođenja procesa razvrstavanja. Potrebno je izračunati očekivanu udaljenost (eng. Expected distance - ED) između svakog nesigurnog objekata i svih grozdova. Za računanje očekivane udaljenosti jednog nesigurnog objekta mora se računati udaljenost svakog uzorka kojim je opisana funkcija gustoće vjerojatnosti do središta grozda. Zbog toga je vrijeme računanja očekivane udaljenosti i po nekoliko stotina puta dugotrajnije nego računanje euklidske udaljenosti [54]. Na primjer ako se za predstavljanje PDF-a koristi 196 uzoraka tada se računa 196 euklidskih

udaljenosti između jednog objekta i jednog grozda, što znači da je računanje očekivane udaljenosti 196 puta vremenski zahtjevnije nego jednostavno računanje euklidske udaljenosti.

U drugom poglavlju opisani različiti algoritmi za razvrstavanje objekata koji ne sadrže nesigurnost. Navedeni algoritmi nemaju mogućnost razvrstavati nesigurne objekte pa je nastala potreba za izmjenom tih algoritama kako bi se mogli primijeniti na nesigurne objekte i uspješno ih razvrstati. Najjednostavniji algoritam za razvrstavanje nesigurnih objekata je uk-means [46]. On je nastao izmjenom algoritma k-means koji se koristi za razvrstavanje sigurnih objekata. Isto kao i k-means on ima mogućnost stvaranja grozdova oko središnje točke na takav način da ukupna očekivana udaljenost objekata do grozdova bude minimizirana. Nedostatak algoritma uk-means je u tome što se moraju računati očekivane udaljenosti od svakog objekata do svih grozdova. Računanje očekivanih udaljenosti je najzahtjevniji dio procesa razvrstavanja pa je uk-means algoritam neučinkovit za razvrstavanje nesigurnih objekata. Kod velikog broja objekata (40000) i uzoraka (1000) proces razvrstavanja može trajati i preko 100 sati. Nastala je potreba za novim algoritmima u kojima se nastoji izbjeći računanje očekivanih udaljenosti. Pomoću procesa odbacivanja ili čišćenja (eng. Pruning) pronalaze se oni grozdovi kojima objekt sigurno ne može biti dodijeljen te se oni odbacuju bez računanja očekivane udaljenosti. Takvi algoritmi pripadaju u skupinu algoritama za čišćenje.

Jedan od tih algoritama koristi Voronojeve dijagrame [55] za odbacivanje grozdova bez računanja očekivane udaljenosti. Algoritam pomoću *k* središta grozdova dijeli prostor skupa objekata na *k* Voronojevih ćelija i provjerava nalazi li se MBR jednog objekta unutar neke Voronojeve ćelije [56][57]. Ako se MBR nesigurnog objekta nalazi unutar bilo koje Voronojeve ćelije, ostali grozdovi su odbačeni bez računanja očekivane udaljenost. Objekt se pridružuje grozdu čije je središte unutar promatrane Voronojeve ćelije. Time je vrijeme izvođenja procesa razvrstavanja skraćeno, jer je izbjegnuto računanje većeg dijela očekivanih udaljenosti. Drugi pristup razvrstavanju nastao je upotrebom MinMax algoritma [58]. MinMax algoritam za svaki objekt pronalazi minimalnu i maksimalnu udaljenost objekta do središta grozda. U sljedećem koraku pronalazi se najmanja maksimalna udaljenost između objekta i svih grozdova. Ako za neki grozd vrijedi da je njegova minimalna udaljenost veća od najmanje maksimalne udaljenosti tada se taj grozd može odbaciti kao kandidat za promatrani objekt bez računanja očekivane udaljenosti. U sljedećim poglavljima navedeni algoritmi su detaljno objašnjeni i opisane su njihove prednosti i nedostaci.

3.1. Algoritam uk-means

Algoritam uk-means nastao je izmjenom algoritma k-means koji se koristio za razvrstavanje objekata koji ne sadrže nesigurnost. K-means predstavljen je u [28] i može podijeliti n objekata u k različitih grozdova. Na početku algoritma odabiru se početna središta grozdova i objekt se dodaje onom grozdu čije središte mu je najbliže. Nakon što se svi objekti dodijele grozdovima, nova središta grozdova računaju se kao srednja vrijednost svih objekata dodijeljenih grozdu. Proces se ponavlja sve dok se ne zadovolji ciljna funkcija ili dok objekti ne prestanu prelaziti iz grozda u grozd. Prilagodbom algoritma k-means za nesigurne pokretne objekte [49] nastao je uk-means (eng. uncertain k-means) [46]. Algoritam uk-means razvijen je za potrebe praćenja pokretnih objekata koji imaju nesigurnost u prostornim podacima. Potrebe mogu biti nadgledanje koji objekti se nalaze unutar jednog kilometra od određene pozicije, koji kamioni se nalaze blizu kamionu koji je pokvaren i treba pomoć, nadzor zračnog prometa u jednom području i tako dalje. Takve primjene nazvane su MOD aplikacije (eng. Moving objects detection). Drugi primjeri su poduzeća za prijevoz, pomorski ili zračni transport. Razvojem sustava za upravljanje bazama podataka (SUBP), ili engleski DBMS (eng. Database management system), omogućeno je spremanje velikih količina informacija o objektima. Ako se bazi postavi upit za pozicijom određenog objekta baza kao odgovor vraća pretpostavljenu lokaciju objekta s koordinatama (x, y) i mogućom nesigurnosti. Nesigurnost je predstavljena minimalnim područjem nesigurnosti u kojem se objekt mora nalaziti. Što se baza češće nadopunjava s novim vrijednostima nesigurnost je manja, ali je veće opterećenje komunikacijskog prometa. Stoga se radi kompromis između nesigurnosti i komunikacije. Nesigurnost objekta prikazuje se pomoću funkcije gustoće vjerojatnosti zbog povećanja sličnosti dobivenih rezultata stvarnim vrijednostima podataka o objektima [59]. Za skup nesigurnih objekata $O = \{o_1, o_2, \dots, o_n\}$ u prostoru R^m s m dimenzija udaljenost između dva objekta uvijek je veća od nule:

$$d(o_i, o_j) \geq 0 \tag{3.1}$$

Funkcija gustoće vjerojatnosti $f_i(x)$ nesigurnog objekta u svakoj točki $x \in R^m$ mora biti veća od nule $f_i(x) > 0 \ \ \forall x \in R^m$ pa za sve točke unutar MBR-a vrijedi sljedeća jednadžba:

$$\int_{x \in R^m} f_i(x)dx = 1 \tag{3.2}$$

Funkcija gustoće vjerojatnosti procjenjuje se tako da se MBR podijeli na odgovarajući broj uzoraka. Svaki uzorak ima vjerojatnost da se objekt nalazi na tome uzorku i svaka od tih vjerojatnosti se sprema u bazu. Za računanje očekivane udaljenosti ED mora se izračunati udaljenost od svakog uzorka do središta grozda. Udaljenosti se množe s funkcijom gustoće vjerojatnosti i zbrajaju u integralu. Za bolju preciznost potreban je što veći broj uzoraka za predstavljanje PDF-a. Očekivana udaljenost između objekta o_i i bilo koje točke y izračunata je prema sljedećoj jednadžbi:

$$ED(o_i, y) = \int_{x \in A_i} d(x, y) f_i(x) dx \tag{3.3}$$

gdje A_i predstavlja minimalno područje nesigurnosti.

Za sve točke koje se nalaze izvan MBR-a funkcija gustoće vjerojatnosti jednaka je nuli $f_i(x) = 0$ pa točke izvan MBR-a nisu uključene u računanje integrala prema jednadžbi (3.3). Cilj razvrstavanja je za neki skup grozdova $C = \{c_1, c_2, \dots, c_k\}$ pronaći sve relacije $h: \{1, \dots, n\} \rightarrow \{1, \dots, k\}$ između objekata i grozdova, takve da ukupna očekivana udaljenost TED između objekata i središta grozdova bude minimizirana:

$$TED = \sum_{i=1}^{n} ED\left(o_i, c_{h(i)}\right) \tag{3.4}$$

Algoritam uk-means opisan je sljedećim pseudo kodom:

```
Odabrati k različitih grozdova
Ponavljati
  Za svaki objekt o_i iz skupa objekata
    Za svaki grozd c_j iz skupa grozdova
      Izračunati očekivanu udaljenost ED(o_i,c_j)
      Pridružiti objekt grozdu sa najmanjom ED
  Izračunati nova središta grozdova
Sve dok središta grozdova ne konvergiraju
```

Slika 3.1 Pseudo kod algoritma uk-means

Algoritam uk-means se može prilagoditi kako bi minimizirao sumu očekivanih kvadratnih pogreški $E(SSE)$. Ako se nesigurni objekt predstavi s x_i i njegov PDF s $f(x_i)$ tada se $E(SSE)$ može izračunati prema sljedećoj jednadžbi:

$$E\left(\sum_{j=1}^{k}\sum_{i\in c_j}\|c_j - x_i\|\right) = \sum_{j=1}^{k}\sum_{i\in c_j}\int \|c_j - x_i\|^2 f(x_i)dx_i \tag{3.5}$$

gdje je $\|c_j - x_i\|$ funkcija udaljenosti između grozda i objekta, a središte grozda se računa prema sljedećoj jednadžbi:

$$c_j = E\left(\frac{1}{|C_j|}\sum_{i\in C_j} x_i\right) = \frac{1}{|C_j|}\sum_{i\in C_j} x_i\, f(x_i)dx_i \tag{3.6}$$

Jednadžba (3.6) može se primijeniti za različite oblike područja nesigurnosti i različite funkcije gustoće vjerojatnosti. Prema [49] i [60] postoje dva različita tipa nesigurnosti kod pokretnih objekata. Prvi tip je kod pravocrtnog gibanja, to jest linijska nesigurnost, a drugi tip je nesigurnost kod slobodnog gibanja. Kod pravocrtnog gibanja objekt se giba u jednom smjeru određenom brzinom koja mora biti manja od najveće brzine V_{max}. Stoga je stvarna pozicija objekta od trenutka t_0 do trenutka t uniformno raspodijeljena unutar kruga radijusa $V_{max} \times (t - t_0)$. Ako je središte grozda predstavljeno s $c = (p, q)$, a objekt x je predstavljen linijskom nesigurnosti koja ima uniformnu distribuciju. Isto tako ako krajnje točke linijske nesigurnosti predstavimo s (a, b) i (c, d) parametri te dužine opisani su $(a + t(c - a), b + t(d - b))$, gdje je t u rasponu [0,1]. Slijedi da je duljina te dužine jednaka:

$$D = \sqrt{(c-a)^2 + (d-b)^2} \tag{3.7}$$

Suma očekivanih kvadratnih pogreški *E(SSE)* za linijsku nesigurnost može se izračunati kao:

$$E(\|c - x\|^2) = \int_0^1 f(t)(D^2t^2 + Bt + C)dt \tag{3.8}$$

gdje je $B = 2\big[(c-a)(a-p) + (d-b)(b-q)\big]$, $C = (p-a)^2 + (q-b)^2$.

Ako se radi o uniformnoj razdiobi tada vrijedi da je $f(t) = 1$, i nakon uvrštavanja u jednadžbu (3.8) nakon izračuna dobije se sljedeća vrijednost:

$$E(\|c - x\|^2) = \frac{D^2}{3} + \frac{B}{2} + C \tag{3.9}$$

Kod slobodnog gibanja, ako je središte grozda predstavljeno s $c = (p, q)$ i objekt x je predstavljen s kružnom nesigurnosti koja ima uniformnu razdiobu. Nadalje, ako kružna nesigurnost ima središte (h, k) i radijus R, a PDF je predstavljen s $f(r, \theta)$ slijedi da je:

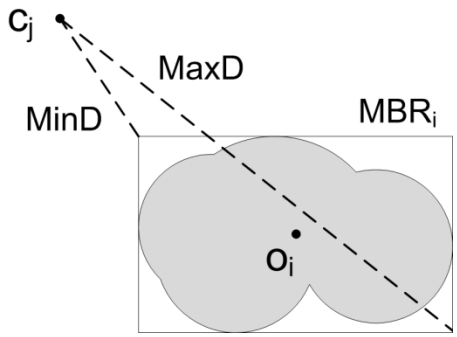
$$E(\|c - x\|^2) = \int_0^R \int_0^{2\pi} f(r, \theta)(A\, cos\theta + B\, sin\theta + C) r d\theta dr \tag{3.10}$$

gdje je $A = 2r(h - p)$, $B = 2r(k - q)$, $C = r^2 + (h - p)^2 + (k - q)^2$.

Osim uniformne razdiobe može se koristiti Gaussova razdioba ili tehnike za uzorkovanje koje predstavljaju PDF pomoću uzoraka. Za sve razdiobe, na isti način može se izračunati suma očekivanih kvadratnih pogreški.

3.2. Algoritam MinMax

U prošlom poglavlju je navedeno da je računanje očekivane udaljenosti ED vremenski zahtjevno zbog velikog broja uzoraka koji su potrebni za predstavljanje funkcije gustoće vjerojatnosti. Upotrebom uk-means algoritma moraju se računati očekivane udaljenosti od svakog objekata do svih grozdova. Ako je broj objekata jednak n, broj grozdova jednak k i broj iteracija jednak t, tada je broj računanja očekivanih udaljenosti u čitavom procesu razvrstavanja nkt. To je puno računanja koje oduzima vrijeme i kako bi se smanjio broj računanja očekivanih udaljenosti uvode se algoritmi za čišćenje (eng. Pruning). U skupinu algoritama za čišćenje pripada i algoritam MinMax [59]. U algoritmu MinMax se za izbjegavanje računanja nepotrebnih očekivanih udaljenosti koristi minimalno područje nesigurnosti (eng. Minimum Bounding Rectangle-MBR). Unutar MBR-a moraju se nalaziti sve moguće vrijednosti objekta kao što je prikazano na slici 2.2. Izvan MBR-a vjerojatnost postojanja objekta jednaka je nuli.



Slika 3.2 Minimalno pravokutno područje nesigurnosti objekta o_i

Pomoću MBR-a moguće je umjesto očekivane udaljenosti ED koristiti euklidsku udaljenost koja je jednostavnija za računanje. Na taj način neki grozdovi su odbačeni kao mogući kandidati za pojedini objekt, bez računanja očekivane udaljenosti. U algoritmu MinMax je za svaki objekt potrebno definirati minimalnu udaljenost do središta nekog grozda. Za objekt o_i pronalazi se minimalna udaljenost do središta nekog grozda prema sljedećoj jednadžbi:

$$MinD(o_i, c_j) = min_{x \in MBR_i} d(x, c_j) \tag{3.11}$$

Pronalazi se i maksimalna udaljenost do središta nekog grozda prema sljedećoj jednadžbi:

$$MaxD(o_i, c_j) = max_{x \in MBR_i} d(x, c_j) \tag{3.12}$$

Nakon toga potrebno je za objekt o_i pronaći minimalnu udaljenost među maksimalnim udaljenostima do središta grozdova:

$$MinMaxD(o_i, c_j) = min_{c_j \in C}\{MaxD(o_i, c_j)\} \tag{3.13}$$

Iz svega navedenog je očito, da je minimalna udaljenost od objekta do središta grozda manja od očekivane udaljenost ED, te da je maksimalna udaljenost od objekta do središta grozda veća od očekivane udaljenosti ED kao što je prikazano sljedećom jednadžbom:

$$MinD\big(o_i, c_j\big) \leq ED(o_i, c_j) \leq MaxD(o_i, c_j) \tag{3.14}$$

Prema jednadžbi (3.14), ako za dva različita grozda c_p i c_q vrijedi da je minimalna udaljenost od nesigurnog objekta o_i do grozda c_p veća od maksimalne udaljenosti objekta o_i do grozda c_q kao na sljedećoj jednadžbi:

$$MinD(o_i, c_p) \geq MaxD(o_i, c_j) \tag{3.15}$$

Grozd c_p se bez računanja očekivane udaljenosti može odbaciti kao kandidat za objekt o_i. Postupak se ponavlja i svi grozdovi koji imaju minimalnu udaljenost do objekta o_i veću od najmanje maksimalne udaljenosti $MinMaxD$ do objekta o_i su odbačeni kao kandidati. Postupak je ponovljen za sve objekte iz skupa nesigurnih objekata i izbjegava se računanje većine očekivanih udaljenosti. Algoritam MinMax opisan je sljedećim pseudo kodom:

Za svaki grozd c_j i objekt o_i
  Izračunati $MinD(o_i,c_j)$ i $MaxD(o_i,c_j)$
  Izračunati $MinMaxD(o_i)$
Za svaki grozd c_j iz skupa grozdova
  Ako je $MinD(o_i,c_j)>MinMaxD(o_i)$
    Ukloniti grozd c_j kao kandidata za objekt o_i
Za sve preostale grozdove izračunati ED

Slika 3.3 Pseudo kod algoritma MinMax

Za sve preostale grozdove, koji nisu odbačeni MinMax algoritmom, potrebno je izračunati njihovu očekivanu udaljenost do objekta o_i i objekt se dodjeljuje onom grozdu koji ima najmanju očekivanu udaljenost. Osim navedenog poboljšanja u odnosu na uk-means, algoritam MinMax nudi još jedno poboljšanje. Svaki put kad se izračuna očekivane udaljenosti $ED(o_i, c_j)$ upisuje se vrijednost najmanje očekivane udaljenosti $Min\left(ED(o_i, c_j)\right)$ i ako je minimalna udaljenost nekog grozda manja od najmanje očekivane udaljenosti on se odbacuje prema sljedećoj jednadžbi:

$$MinD(o_i, c_p) \geq Min\left(ED(o_i, c_j)\right) \tag{3.16}$$

Na taj način dodatno je smanjen broj računanja očekivanih udaljenosti. Algoritam MinMax je učinkovit kod odbacivanja onih grozdova koji su daleko od objektu o_i najbližeg grozda. Druga bitna stavka kod algoritma MinMax je veličina MBR-a. Ako je MBR malen

većina grozdova će biti odbačena bez računanja očekivane udaljenosti, no ako je MBR velik tada će najmanja maksimalna udaljenost biti velika i proces odbacivanja grozdova neće biti toliko uspješan. Može se zaključiti da je uspješnost algoritma MinMax ovisna o veličini područja nesigurnosti, i ako je područje nesigurnosti malo algoritam će imati kraća vremena razvrstavanja, dok će u suprotnom slučaju imati dulja vremena razvrstavanja.

Kako bi se smanjilo minimalno područje nesigurnosti moguće je napraviti određena poboljšanja. U [61] predložena je upotreba nejednakosti trokuta, koje se koristi za određivanje najbolje gornje i donje granice udaljenosti između dvije točke. Poboljšanjem granica dodatno se smanjuje broj računanja očekivanih udaljenosti. Za potrebe nejednakosti trokuta mora se uzeti sidrišna točka y. Pomoću nejednakosti trokuta može se odrediti gornja i donja granica očekivane udaljenosti ED između objekta o_i i grozda c_j prema sljedećoj jednadžbi:

$$ED(o_i, c_j) \leq ED(o_i, y) + d(y, c_j) \tag{3.17}$$

gdje je $d(y, c_j)$ udaljenost od središta grozda c_j do točke y.

Ako se očekivana udaljenost između objekta o_i i točke y izračuna prije samog razvrstavanja, jer se ona ne mijenja tijekom iteracija, slijedi da se gornja granica očekivane udaljenosti $ED(o_i, c_j)$ može izračunati koristeći jednostavno računanje udaljenosti između točke y i središta grozda c_j. Ovakav način računanja zove se metoda procjene gornje granice pomoću predračuna očekivanih udaljenosti (eng. Upper Bound Estimation on Precomputed Expected Distances) i označava se s U_{pre}. Sada je vrlo lako dodati procijenjenu gornju granicu u algoritam MinMax. Procijenjena gornja granica uspoređuje se s maksimalnom udaljenosti i kao maksimalna udaljenost uzima se manja od njih. To se može opisati sljedećim pseudo kodom:

$$If(U_{preij} \leq MaxD(o_i, c_j) \;\; \{MaxD(o_i, c_j) = U_{preij}\} \tag{3.18}$$

Budući da se procijenjena granica koristi samo u onim slučajevima kad daje bolje vrijednosti, to jest manju gornju granicu, ne može se dogoditi da se dobije veće područje nesigurnosti upotrebom procijenjene granice. Osim toga moguće je koristiti više sidrišnih točaka i za svaku točku dobiti procijenjenu gornju granicu. Na kraju se odabire procijenjena granica koja ima najmanju vrijednost. U praktičnoj primjeni treba vidjeti koliko sidrišnih

točaka se može uzeti, a da se i dalje dobije ušteda na vremenu izvođenja. Služeći se sličnim principom moguće je procijeniti i donju granicu prema sljedećoj jednadžbi:

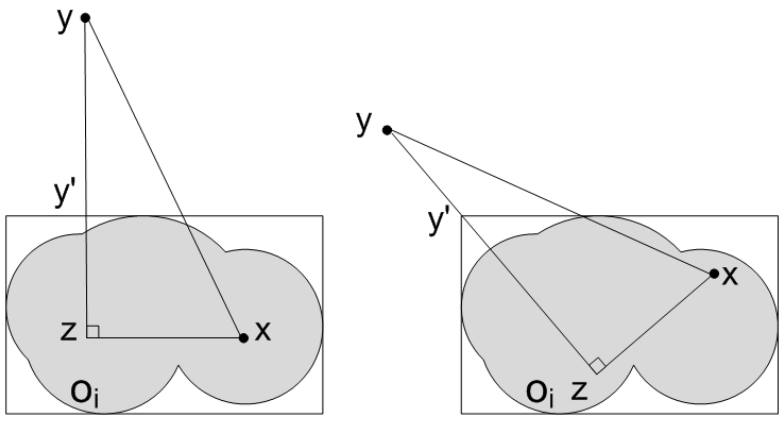
$$ED(o_i, c_j) \geq \left| d(y, c_j) - ED(o_i, y) \right| \tag{3.19}$$

Donja granica očekivane udaljenosti $ED(o_i, c_j)$ može se izračunati koristeći jednostavno računanje udaljenosti između točke y i središta grozda c_j. Procijenjena donja granica L_{pre} uspoređuje se s minimalnom udaljenosti i kao minimalna udaljenost uzima se veća od njih. To se može opisati sljedećim pseudo kodom:

$$If(L_{preij} \geq MinD(o_i, c_j) \;\; \{MinD(o_i, c_j) = L_{preij}\} \tag{3.20}$$

Budući da je sada minimalna udaljenost potencijalno veća nego prije, veća je i vjerojatnost za uspješno odbacivanje grozdova. Upotrebom procijenjene gornje i donje granice dobiveno je potencijalno smanjenje gornje granice i povećanje donje granice. Slijedi da će biti potrebno računati manji broj očekivanih udaljenosti. No sam proces procjene granica ima svoju cijenu, to jest povećanje vremena izvođenja. Ako se koristi r sidrišnih točaka u skupu nesigurnih objekata koji ima n članova tada je potrebno izračunati nr predračunskih očekivanih udaljenosti. Slijedi da će se metoda procjene isplatiti samo u onome slučaju kad je broj grozdova koji će biti odbačeni u svim iteracijama pomoću ove metode veći od nr. Jako je bitno odabrati ispravan broj sidrišnih točaka r kako bi procjena bila isplativa i kako bi se dobilo skraćenje vremena izvođenja.

U [61] je dokazano kako za smanjenje gornje granice sidrišne točke trebaju biti unutar minimalnog područja nesigurnosti. Ako je točka y izvan MBR-a objekta o_i, a točka y' je točka na rubu MBR-a koja je najbliža točki y, mogu se dogoditi dva različita slučaja kao što je prikazano na slici 3.4.



Slika 3.4 Primjer jedne sidrišne točke koja se nalazi izvan MBR-a

U prvome slučaju na slici 3.4, pravac koji spaja točke y i y' okomit je na rubove MBR-a, dok se u drugom slučaju točka y' nalazi u kutu MBR-a. Ako se uzme točka x koja se nalazi unutar MBR-a i koja se nalazi na pravcu koji spaja točke y i y' to jest $x = z$, kao što je prikazano na slici 3.4, tada vrijedi sljedeća jednakost:

$$d(x, y) \geq d(x, y') \tag{3.21}$$

U suprotnome ako se točka x ne nalazi na pravcu koji spaja točke y i y' vrijedi sljedeća jednakost:

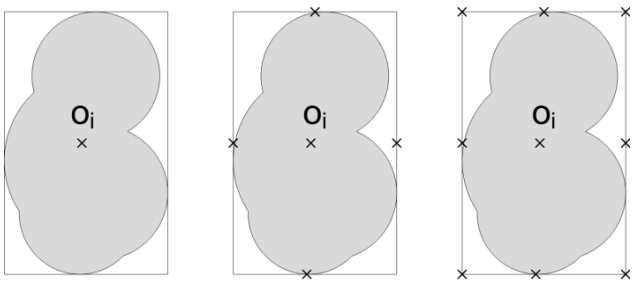
$$d(x, y) = \frac{d(x, z)}{cos\angle yxz} \geq \frac{d(x, z)}{cos\angle y'xz} = d(x, y') \tag{3.22}$$

Iz jednadžbe (3.22) slijedi:

$$ED(o_i, y) = \int f_i(x) d(x, y) dx \geq \int f_i(x) d(x, y') dx = ED(o_i, y') \tag{3.23}$$

Znači da točka y koja minimizira očekivanu udaljenost $ED(o_i, y)$ mora biti unutar MBR-a objekta o_i. Ne postoji jednostavan način pomoću kojeg se može izračunati koja točka minimizira očekivanu udaljenost pa se u praksi uzimaju različite točke unutar MBR-a objekta o_i. Najbolje je odabrati točke na različitim stranama MBR-a. Ovisno o broju sidrišnih točaka postoje različiti modeli odabira točaka kao što je prikazano na slici 3.5. Ako se uzima jedna

točka tada je najbolje odabrati središte MBR-a, u modelu s pet točaka uzima se središte MBR-a i središta svih stranica pravokutnika koji određuje MBR, što osigurava da je barem jedna od točaka bliže reprezentativnoj točki objekta nego neka točka izvan MBR-a.



Slika 3.5 Primjer modela s jednom, pet i devet sidrišnih točaka

Može se pokazati da odabir sidrišnih točaka za procjenu gornje granice daje bolje vrijednosti i za donju procijenjenu granicu [61]. Primjena navedenih metoda za procjenu granica posebno je pogodna u onim primjenama u kojima dio objekata često ne mijenja svoju poziciju. Za takve objekte nije potreban stalan predračun očekivanih udaljenosti što dodatno smanjuje vrijeme izvođenja.

Osim smanjenja razlike između donje i gornje granice očekivane udaljenosti nejednakost trokuta nudi još jedno poboljšanje. Moguće je što više smanjiti udaljenost $d(y, c_j)$ između točke y i središta grozda c_j. Stoga se preporuča u idućoj iteraciji kao sidrišnu točku y uzeti središte grozda iz prethodne iteracije, jer već postoji izračunata očekivana udaljenost $ED(o_i, y)$ koja je izračunata u prethodnoj iteraciji [61]. Ako je grozd c_j u idućoj iteraciji označen s c_j' može se pretpostaviti da je udaljenost $d(c_j, c_j')$ između središta grozda u dvije iteracije mala, pogotovo nakon većeg broja iteracija kad se središte grozda pomiče za neznatne udaljenosti. Primjenom novog poboljšanja za objekt o_i i grozd c_j nastaju dva različita slučaja. Prvi slučaj je kada $MinD(o_i, c_j) \geq MinMaxD(o_i)$ i tada se grozd c_j može odbaciti bez računanja očekivane udaljenosti. U idućoj iteraciji radi se usporedba novih udaljenosti $MinD(o_i, c_j') \geq MinMaxD(o_i)$. Budući da se središte grozda nije znatno promijenilo postoji vrlo velika vjerojatnost da će i grozd c_j' biti odbačen.

Drugi slučaj je kada $MinD(o_i, c_j) \leq MinMaxD(o_i)$ i tada se grozd c_j ne može odbaciti bez računanja očekivane udaljenosti $ED(o_i, c_j)$. U idućoj iteraciji potrebne je izračunati

gornju granicu očekivane udaljenosti $ED(o_i, c_j')$ koja se računa uzimanjem središta grozda iz prethodne iteracije kao sidrišne točke y. Jednadžba za nejednakost trokuta sada izgleda:

$$ED(o_i, c_j') \leq ED(o_i, c_j) + d(c_j, c_j') \tag{3.24}$$

U prethodnoj iteraciji izračunata je očekivana udaljenost $ED(o_i, c_j)$ pa se gornja granica očekivane udaljenosti $ED(o_i, c_j')$ može izračunati jednostavnim računanjem udaljenosti $d(c_j, c_j')$. Ova metoda se zove pomicanje središta grozdova (eng. Cluster shift) i gornja granica očekivane udaljenosti označava se s U_{cs}. Donja granica L_{cs} može se izračunati prema sljedećoj jednadžbi gdje je središte grozda iz prethodne iteracije opet uzeto kao sidrišna točka:

$$ED(o_i, c_j') \geq |ED(o_i, c_j) - d(c_j, c_j')| \tag{3.25}$$

Prednost metode pomicanja središta grozdova je u tome što nije potreban predračun očekivanih udaljenosti, jer su one već izračunate u prethodnoj iteraciji. Navedene granice očekivanih udaljenosti U_{pre}, L_{pre}, U_{cs} i L_{cs} mogu se međusobno kombinirati u različitim situacijama. Preporučuje se u prvim iteracijama koristiti U_{pre} i L_{pre}, jer se tada središta grozdova pomiču za znatnije vrijednosti, budući da objekti još migriraju među grozdovima. U kasnijim iteracijama središta grozdova se neznatno pomiču pa se preporučuje koristiti granice U_{cs} i L_{cs}. Rezultati pokusa koji pokazuju učinkovitost pojedinih granica i broj sidrišnih točaka predstavljeni su u [58].

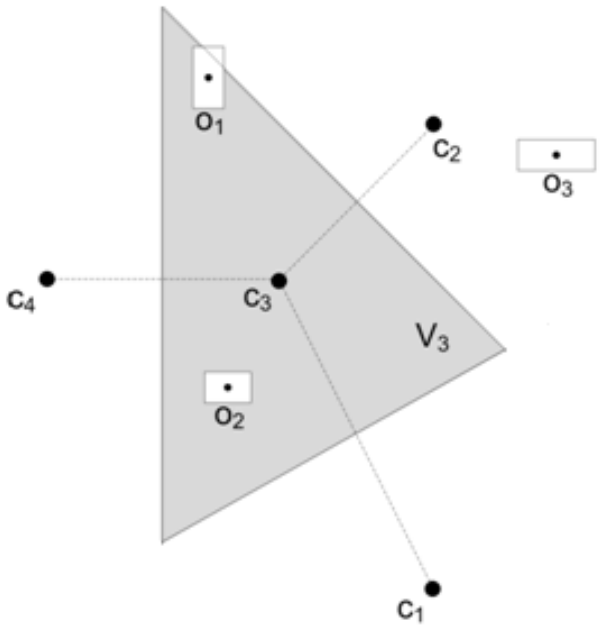
3.3. Algoritam koji koristi Voronojeve dijagrame

U prošlom poglavlju opisan je algoritam MinMax koji je značajno povećao uspješnost razvrstavanja u odnosu na uk-means algoritam. Algoritam MinMax koristio je minimalno područje nesigurnosti za određivanje granica očekivane udaljenosti, na osnovu kojih se moglo izbjeći računanja većine očekivanih udaljenosti. Nedostatak algoritma MinMax je u tome što on ne uzima u obzir geometrijsku strukturu prostora R^m. Nasuprot tome, algoritam koji koristi Voronojeve dijagrame uzima u obzir geometrijsku strukturu prostora R^m i na taj način

odbacuje veći broj grozdova [56]. On pomoću skupa grozdova $C = \{c_1, c_2, \dots, c_k\}$ dijeli geometrijski prostor R^m na k Voronojevih dijagrama (ćelija) koje imaju sljedeće svojstvo:

$$d(x, c_j) \leq d(x, c_q) \quad \forall x \in V(c_j), \;\; c_j \neq c_q \tag{3.26}$$

Izgled jedne Voronojeve ćelije prikazan je na slici 3.6.



Slika 3.6 Izgled jedne Voronojeve ćelije V_3

Granica između dvije susjedne ćelije $V(c_p)$ i $V(c_q)$ je simetrala (eng. Bisector) koja se označava s $B_{p|q}$. To je pravac koji je okomit i prolazi polovicom dužine koja međusobno spaja središta grozdova c_p i c_q. Pomoću svojstva prikazanog jednadžbom (3.26) iterativno se za svaki objekt o_i provjerava leži li njegov MBR u potpunosti unutar neke od Voronojevih ćelija. Ako je taj uvjet zadovoljen objekt o_i se pridružuje grozdu unutar čije ćelije se nalazi. To se može zaključiti, jer vrijedi sljedeće svojstvo:

$$Ed(o_i, c_p) < ED(o_i, c_q) \quad \forall c_q \in C \backslash \{c_p\} \tag{3.27}$$

Ako se objekt o_i pridruži nekom grozdu c_p unutar čije ćelije se nalazi njegov MBR, nema potrebe za računanjem očekivane udaljenosti ED, i svi ostali grozdovi se odbacuju kao potencijalni kandidati za objekt o_i. Sa slike 3.6 je vidljivo da se MBR objekta o_2 u potpunosti nalazi unutar ćelije V_3 pa je objekt o_2 pridružen grozdu c_3 bez računanja očekivane udaljenosti, a svi ostali grozdovi su odbačeni kao kandidati. Nasuprot tome, sa slike 3.6 je vidljivo da se MBR objekta o_1 djelomično nalazi unutar ćelije V_3 pa se objekt o_1 ne može pridružiti grozdu c_3. Isti slučaj je i za objekt o_3 čiji se MBR u potpunosti nalazi izvan ćelije V_3. Za sve objekte koji nisu pridruženi jednom od grozdova potrebno je izračunati očekivane udaljenosti. Algoritam pomoću Voronojevih dijagrama opisan je sljedećim pseudo kodom:

Izračunati Voronojeve diagrame za $C=\{c_1, \dots, c_m\}$
Za svaki c_j iz skupa grozdova i objekte o_i
  ako je MBR_i unutar $V(c_j)$
    objekt o_1 se dodjeljue grozdu c_j
Za sve nedodijeljene objekte i grozdove c_p, c_q
  ako je MBR_i na istoj strani $B_{p/q}$ kao grozd c_p
    ukloni c_q kao kandidata
Za sve preostale grozdove izračunati ED

Slika 3.7 Pseudo kod algoritma koji koristi Voronojeve dijagrame.

Osim odbacivanja grozdova pomoću Voronojevih dijagrama ova metoda može odbacivati grozdove pomoću simetrala. Ranije je rečeno da se Voronojevi dijagrami konstruiraju pomoću simetrala, koje su dostupne uz malo dodatnog vremena izvođenja. Za svaki objekt o_i se provjerava leži li njegov MBR u potpunosti u ravnini $H_{p/q}$. Ako je taj uvjet zadovoljen grozd c_q se odbacuje kao kandidat za objekt o_i. Međusobnom usporedbom preostalih grozdova, još se nekoliko grozdova može odbaciti kao potencijalni kandidati za objekt o_i i time je dodatno smanjen broj računanja očekivanih udaljenosti. Sljedeći teorem dokazuje da algoritam pomoću Voronojevih dijagrama odbacuje veći broj grozdova nego algoritam MinMax [56].

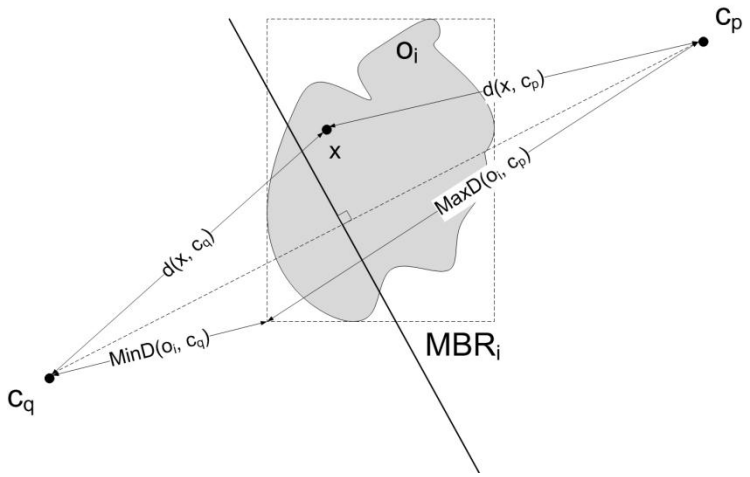
Teorem – Za neki objekt $o_i \in O$ i grozd $c_p \in C, p = 1, \dots, k$, vrijedi da ako algoritam pomoću Voronojevih dijagrama ne odbaci grozd c_p kao kandidata za objekt o_i tada ga neće odbaciti niti MinMax algoritam. Kao dokaz koristi se grozd c_q koji ima najmanju maksimalnu udaljenost MinMaxD do objekta o_i prema sljedećoj jednadžbi:

$$MaxD(o_i, c_q) = MinMaxD(o_i) \tag{3.28}$$

tada se mogu dogoditi dva različita slučaja. U prvom slučaju grozdovi c_p i c_q su jednaki. Tada vrijedi sljedeće:

$$\begin{matrix} & \leq & \\ MaxD(o_i, c_p) & = & MaxD(o_i, c_p) \quad p = q \\ & = & \end{matrix} \qquad \begin{matrix} MaxD(o_i, c_p) \\ \\ MinMaxD(o_i) \end{matrix} \tag{3.29}$$

MinMax algoritam odbacuje grozd c_p kao kandidata za objekt o_i samo u onim slučajevima kada je $MaxD(o_i, c_p) > MinMaxD(o_i)$, zbog toga on neće odbaciti grozd c_p u ovome slučaju iz čega slijedi da je teorem zadovoljen. U drugome slučaju grozdovi c_p i c_q su različiti. U tome slučaju mogu postojati dva nova pod slučaja. U prvome pod slučaju MBR objekta o_i se u potpunosti nalazi u ravnini $H_{q/p}$. U tome slučaju grozd će biti odbačen, što ne mora biti slučaj za algoritam MinMax, pa je teorem i dalje zadovoljen. U drugome pod slučaju MBR objekta o_i se preklapa s ravninom $H_{q/p}$. Ako se pogleda točka x, kao što je prikazano na slici 3.8, i za koju vrijedi $MBR_i \cap (H_{p/q} \cup B_{p|q})$.

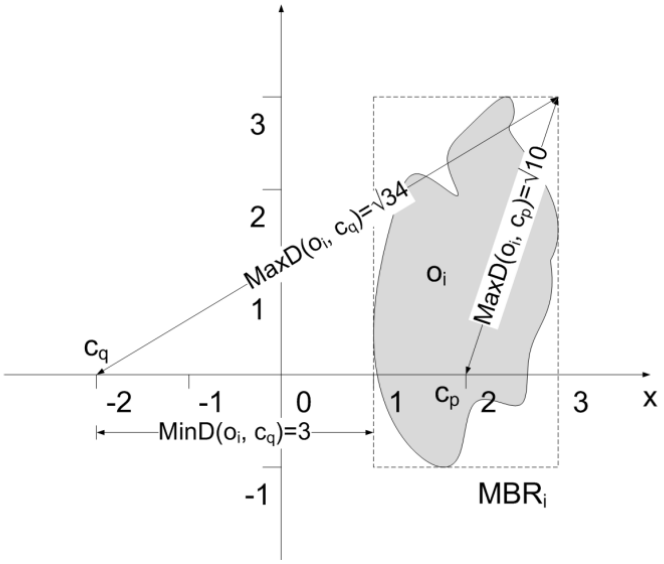


Slika 3.8 Primjer odbacivanja grozda u slučaju kad MBR objekta siječe simetralu

Dobije se sljedeće:

$$MaxD(o_i, c_p) \begin{matrix} \leq \\ \leq \\ \leq \\ = \end{matrix} \begin{matrix} d(x, c_q) \\ d(x, c_q) \\ MaxD(o_i, c_q) \\ MinMaxD(o_i) \end{matrix} \quad \begin{matrix} jer\ je\ x \in MBR_i \\ jer\ jex \in H_{p/q} \cup B_{p|q} \\ po\ definiciji \\ po\ definiciji \end{matrix} \tag{3.30}$$

I u ovome slučaju uvjet algoritma MinMax nije zadovoljen i grozd c_p nije odbačen, pa teorem i dalje vrijedi. Na slici 3.9 prikazan je slučaj u kojem algoritam pomoću Voronojevih dijagrama odbacuje grozd, dok ga algoritam MinMax ne odbacuje.



Slika 3.9 Primjer uspješnog odbacivanja grozda pomoću algoritma koji koristi Voronojeve dijagrame i neuspješnog odbacivanja pomoću algoritma MinMax

Na slici 3.9 prikazani su grozdovi c_p i c_q i simetrala $B_{p|q}$ između dva grozda koja je jednaka koordinatnoj osi y. Ovakvo rješenje je odabrano radi lakše razumljivosti primjera. Na istoj slici prikazan je i objekt o_i čiji se MBR u potpunosti nalazi u ravnini $H_{p/q}$. Zbog toga je grozd c_q odbačen kao kandidat za objekt o_i. Ako se očitaju koordinate sa slike dobije se da je:

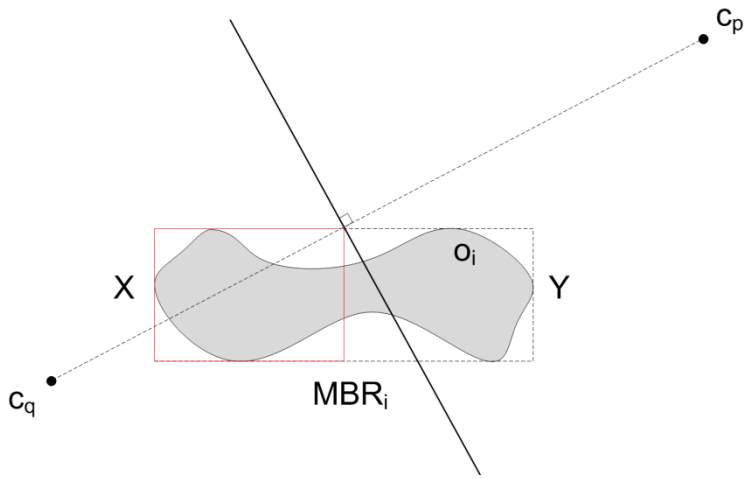
$$MinD(o_i, c_q) = 3$$

$$MaxD(o_i, c_q) = \sqrt{34}$$

$$MaxD(o_i, c_p) = \sqrt{10}$$

Slijedi da je $MinMaxD(o_i) > MinD(o_i, c_q)$ pa algoritam MinMax ne može odbaciti grozd c_q, jer nije zadovoljen uvjet za odbacivanje grozda. Pomoću svega navedenog dokazano je da algoritam koji koristi Voronojeve dijagrame ima veći broj odbačenih grozdova nego algoritam MinMax, što je bio i cilj gornjeg teorema. Za sve grozdove koji nisu odbačeni mora se računati očekivana udaljenost ED. Da bi se smanjilo potrebno vrijeme računanja može se koristiti djelomično računanje očekivane udaljenosti, u kojem nije potrebno računati čitavi integral od $ED(o_i, c_q)$. Ako se promatraju dva grozda c_p i c_q te objekt o_i, i ako MBR objekta o_i siječe simetralu $B_{p|q}$ tada se MBR dijeli na dva dijela X i Y za koje vrijedi sljedeće:

$$X \cup Y = MBR_i\,,\;\; X \cap Y = 0\,, X \in V(c_p) \tag{3.31}$$



Slika 3.10 Podjela minimalnog područja nesigurnosti na dva dijela

Podjela minimalnog područja nesigurnosti MBR prikazana je na slici 3.10. Sada se računanje očekivane udaljenosti može rastaviti na dva manja integrala:

$$ED(o_i, c_p) = \int_{x \in MBR_i} d(x, c_p)\, f_i(x)$$
$$= \int_{x \in X} d(x, c_p)\, f_i(x) + \int_{x \in Y} d(x, c_p)\, f_i(x) = ED_X(o_i, c_p) + ED_Y(o_i, c_p) \tag{3.32}$$

Jednadžba (3.32) se na isti način primjenjuje na grozd c_p. Budući da vrijedi $x \in V(c_p)$ slijedi da je $ED_X(o_i, c_p) < ED_X(o_i, c_q)$.

Nakon toga treba izračunati vrijednosti integrala $ED_Y(o_i, c_p)$ i $ED_Y(o_i, c_q)$ i ako vrijedi sljedeća nejednakost:

$$ED_Y(o_i, c_p) < ED_Y(o_i, c_q) \tag{3.33}$$

tad vrijedi sljedeće:

$$ED(o_i, c_p) < ED(o_i, c_q) \tag{3.34}$$

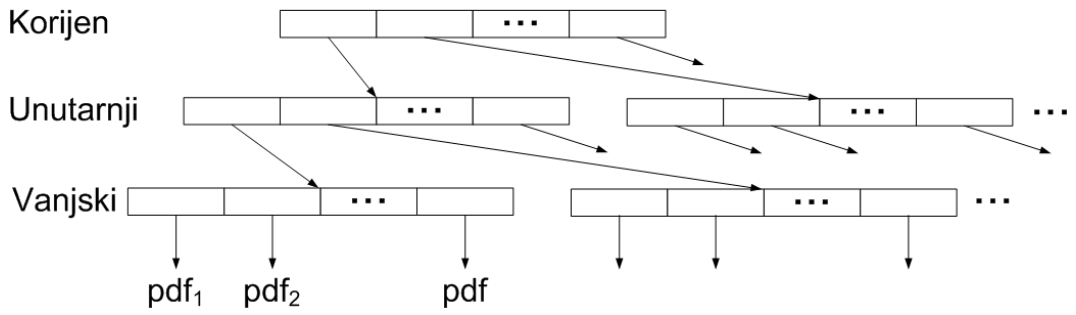
I grozd c_q može biti odbačen kao kandidat za objekt o_i. Ako gornji uvjet nije bio zadovoljen grozd c_q se ne može odbaciti kao kandidat za objekt o_i i mora se računati očekivana udaljenost $ED(o_i, c_q)$, no prednost je u tome što se samo mora izračunati $ED_X(o_i, c_q)$ i zbrojiti već ranije izračunatu $ED_Y(o_i, c_q)$ kako bi se dobila očekivana udaljenost $ED(o_i, c_q)$. Korištenjem djelomičnog računanja očekivane udaljenosti odbacuje se veći broj grozdova uz nešto dodatnog računanja. Bitno je naglasiti da se ova metoda može kombinirati i s metodom pomicanja središta grozdova (eng. Cluster shift) koja je predstavljena u poglavlju o algoritmu MinMax kako bi se dobili još bolji rezultati. Pokusi su pokazali da se kombinacijom ovih dvaju metoda dobiju bolji rezultati uz nekoliko dodatnih matematičkih operacija.

3.4. Razvrstavanje pomoću R-stabla

Razvrstavanje pomoću R-stabla nastoji grupirati nesigurne objekte u slične grupe i umjesto da se računske operacije provode nad objektima one se provode nad grupama [57]. Na taj način se uvjeti za odbacivanje grozdova koji su korišteni u algoritmima za razvrstavanje mogu ispitivati nad čitavom grupom umjesto nad jednim objektom. Prednost je u tome što se taj uvjet ispituje samo jednom, a prije se ispitivao nad svakim objektom posebno, te ako je uvjet zadovoljen grozd se odbacuje za sve objekte koji se nalaze u odbačenoj grupi. U najboljem slučaju može se dogoditi da se jedan grozd dodijeli čitavoj grupi i onda nema potrebe za računanjem očekivane udaljenosti za sve objekte unutar grupe.

Razvrstavanje pomoću R-stabla spada u hijerarhijske algoritme, jer se dvije grupe mogu spojiti u jednu i tako stvaraju hijerarhiju grupa. Tehnike za odbacivanje grozdova mogu se primijeniti na različite razine u stablu i ako je uvjet zadovoljen za gornju razinu, slijedi da je

uvjet zadovoljen i za sve grupe u razinama ispod nje. Stoga je logično krenuti od korijena stabla prema dolje i ispitivati uvjete. Ako je uvjet zadovoljen na većoj razini dobiju se bolja vremena izvođenja, jer je napravljeno manje ispitivanja uvjeta i odbačen je veći broj objekata. R-stablo je takva struktura u kojoj se sve vanjske grane nalaze na istoj dubini, to jest na istoj razini. U unutarnje čvorove stabla se ne spremaju MBR-ovi svih objekata nego samo MBR grupe. Primjer R-stabla prikazan je na slici 3.11.

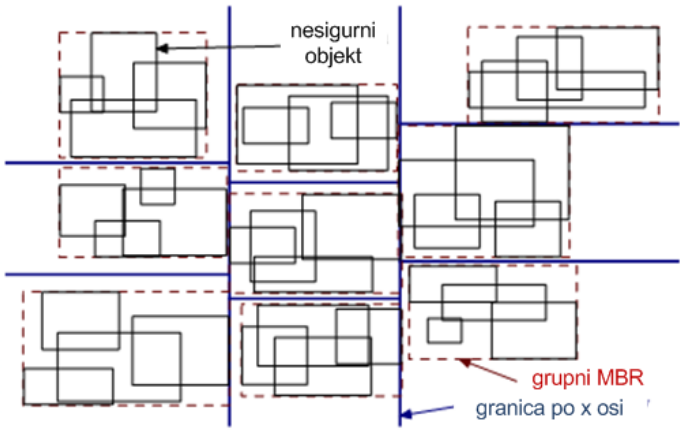


Slika 3.11 Primjer R-stabla

Broj objekata u jednoj vanjskoj grupi, to jest čvoru, nije fiksan, nego se određuje prema broju objekata koje treba umetnuti u stablo. Ukupna visina stabla ovisi o ukupnom broju objekata i broju objekata u jednoj grupi. Za razliku od unutarnjih čvorova gdje se ne spremaju podaci o objektima, u vanjskim čvorovima se sprema MBR svakog objekta, središte nesigurnog objekta i pokazivač na PDF objekta koji se sprema u vanjskoj memoriji radi smanjenja memorijskog prostora potrebnog za stablo. No PDF može biti spremljen i u samome stablu. Svi vanjski čvorovi se udružuju i tvore unutarnje čvorove. U unutarnjim čvorovima se sprema MBR grupa potomaka, broj objekata u pojedinim grupama potomaka, središte grupa potomaka i pokazivač na čvorove potomaka. Algoritam pomoću kojeg se stvara R-stablo (eng. Sort Tile Recursive algorithm) detaljno je opisan u [57]. Algoritam kreće od nižih razina prema gore i na kraju završava s korijenskim čvorom. Primjer konstrukcije R-stabla pomoću ovog algoritma prikazan je na slici 3.12.

Na slici 3.12 prikazano je 36 MBR-ova nesigurnih objekata. Ako u svakom vanjskom čvoru treba biti četiri objekta, slijedi da je potrebno devet vanjskih čvorova. Algoritam u prvoj iteraciji dijeli objekte u devet različitih grupa kao što je prikazano na slici 3.12. Podjela se prvo radi po x koordinatnoj osi i nastoji se podijeliti objekte u tri okomite grupe. Dijeli se na tri grupe, jer je to jednako korijenu iz ukupnog broja grupa. Središta objekata se sortiraju

prema x koordinatama i tako u svakoj grupi imamo po 12 članova. Nakon toga se svaka grupa dijeli u tri nove grupe sortirajući ih prema y koordinatnoj osi i tako se dobije podjela objekata na devet grupa. Više razine se grupiraju na sličan način i sve se ponavlja dok se ne dođe do korijena stabla u kojem se nalaze svi objekti.



Slika 3.12 Konstrukcija R-stabla podjelom objekata u grupe

Razvrstavanje počinje tako da se prvo provjerava korijenski čvor. Korijenski čvor sastoji se od više unosa od koji svaki unos predstavlja jednu grupu ili nad grupu koja se sastoji od više podgrupa s nižih razina. Za svaki unos u korijenskom čvoru, i njegov MBR koji je spremljen u stablu, primjenjuje se neki od algoritama za razvrstavanje. Algoritmi se na grupe primjenjuju na isti način kao što se primjenjuju i na same objekte. Ako je uvjet algoritma zadovoljen za grupu onda je zadovoljen za sve objekte unutar grupe, te za sve objekte u podgrupama.

To je značajno poboljšanje u vremenu izvođenja procesa razvrstavanja, jer se uvjeti ne moraju provjeravati nad cijelim podstablom promatranog čvora. Poboljšanje je bolje što se odbacivanje dogodi na višim razinama. Na primjer ako je u promatranoj grupi bilo 30 čvorova potomaka, a u svakom čvoru je bilo 5 objekata, slijedi da nema potrebe za provjeravanjem 150 objekata. Bitno je napomenuti da se R-stablo konstruira na samome početku i ne mijenja se tijekom iteracija, pa su troškovi konstrukcije R-stabla zanemarivi u odnosu na ukupno vrijeme razvrstavanja.

Algoritam za razvrstavanje pomoću R-stabla nastaje izmjenom uk-means algoritma, tako da se umjesto objekata razvrstavaju grupe, to jest unutarnji čvorovi R-stabla. Procesiranje vanjskih čvorova je slično procesiranju unutarnjih čvorova, samo se mora dodati dio za računanje očekivanih udaljenosti. Razvrstavanje pomoću R-stabla opisano je sljedećim pseudo kodom:

```
Odabrati k različitih grozdova
Ponavljati
  Za svaki čvor stabla
    koristiti neki od algoritama za odbacivanje koristeći
    MBR grupe (čvora)
    ako je preostao samo jedan grozd kandidat
      sve objekte iz grupe dodijeli tome grozdu
    u suprotnom
      ako je unutarnji čvor
        pozvati proceduru UnutarnjiČvor
      u suprotnom
        pozvati proceduru VanjskiČvor
  Izračunati nova središta grozdova
Sve dok središta grozdova ne konvergiraju
```

Slika 3.13 Pseudo kod algoritma pomoću R-stabla

4. Podjela područja skupa objekata i paralelno razvrstavanje simetralnom podjelom prostora

Razvrstavanje nesigurnih objekata vremenski je zahtjevan proces i najviše istraživanja posvećeno je u traženju postupaka koji skraćuju vrijeme izvođenja procesa razvrstavanja. U praktičnim primjenama poželjno je što prije dobiti željene rezultate kako bi se na osnovi njih mogle odrediti buduće akcije, saznati trenutno i predvidjeti buduće stanje proučavanog sustava. U ovoj disertaciji proučavane su prednosti i nedostaci postojećih algoritama za razvrstavanje kako bi se stvorili novi algoritmi koji će skratiti vrijeme izvođenja procesa razvrstavanja. U različitim primjenama broj promatranih nesigurnih objekata može biti jako velik pa se njihovi podaci moraju spremati u bazu podataka. Prvo se nastojalo napraviti takvu strukturu podataka kojoj se može pristupati u što kraćem vremenu, to jest što brže čitati i mijenjati vrijednosti podataka. Ako se koriste SQL baze podataka moguće je koristiti indeksiranje [60] pomoću kojeg se vrijednostima u bazi dodaju indeksi. Dodavanjem indeksa upiti nad bazom podataka izvršavaju se znatno brže i ubrzavaju pribavljanje traženih vrijednosti. Nadalje, istraživano je koji algoritmi imaju najbolji postupak odbacivanja grozdova kao kandidata za pojedini objekt.

Pod samim postupkom odbacivanja grozdova promatraju se dvije različite stvari. Prvo se promatra koji algoritam ima veći postotak odbačenih grozdova, a zatim se promatra brzina procesa odbacivanja grozdova. Neki algoritam može imati veći postotak odbačenih grozdova, ali sam proces odbacivanja može biti sporiji nego kod drugog algoritma. Stoga algoritam može imati dulje vrijeme izvođenja procesa razvrstavanja iako je odbacio veći broj grozdova. Na primjer, algoritam koji koristi Voronojeve dijagrame ima veći postotak odbačenih grozdova nego algoritam MinMax, ali je proces odbacivanja brži kod algoritma MinMax. Stoga je nužno za različiti broj objekata, broj grozdova, veličinu MBR-a i broj uzoraka pokusima provjeriti brzinu izvođenja pojedinog algoritma. Kao što je ranije rečeno MBR je predstavljen pomoću funkcije gustoće vjerojatnosti (eng. Probability density function-PDF) koja je predstavljena skupom uzoraka. Uzorci nastaju tako da se MBR podijeli na određeni broj ćelija i svaki uzorak ima vjerojatnost da se objekt nalazi na tome uzorku. Zbroj vjerojatnosti za sve uzorke unutar MBR-a mora biti jednak jedan. Zbog preciznosti potreban je velik broj uzoraka za predstavljanja PDF-a svakog objekta što znatno povećava vrijeme računanja očekivane udaljenosti, jer se računa udaljenost od svakog uzorka do središta

grozda. Vrijeme računanja očekivane udaljenosti je i po nekoliko stotina puta dulje nego računanje jednostavne euklidske udaljenosti. Stoga su istraživani i načini kako što jednostavnije izračunati očekivanu udaljenost, te dodatno smanjiti vrijeme izvođenja za one grozdove koji nisu odbačeni i za koje je nužno računanje očekivane udaljenosti [54]. U idućim poglavljima predstavljena su dva postupka koja su razvijena u ovoj disertaciji. Svaki od postupaka poboljšava jedan dio procesa razvrstavanja, a njihovom kombinacijom dobiveno je dodatno poboljšanje procesa razvrstavanja.

4.1. Razvrstavanje zasnovano na simetralnoj podjeli prostora, usporedbom i odbacivanjem grozdova

U ovom poglavlju predstavljen je prvi postupak razvrstavanja nesigurnih objekata koji je razvijen u ovoj disertaciji. Postupak koristi simetralnu podjelu prostora za odbacivanje grozdova, pa je nazvan SPP algoritam. Razvrstavanje pomoću SPP algoritma zasnovano je simetralnoj podjeli prostora između dvaju grozdova, i na osnovi te podjele odbacuje se jedan od grozdova kao kandidat za promatrani objekt. Simetrala $B_{p|q}$ je pravac koji je okomit i prolazi polovicom dužine koja spaja središta grozdova c_p i c_q. Pomoću simetrala dvodimenzionalni prostor je podijeljen na dvije ravnine $H_{p/q}$ i $H_{q/p}$. U prvoj ravnini nalazi se grozd c_p, a u drugoj c_q. Iz svega navedenog slijede svojstva:

$$d(x, c_p) < d(x, c_q) \quad \forall x \in H_{p/q} \tag{4.1}$$

$$d(x, c_p) = d(x, c_q) \quad \forall x \in B_{p|q} \tag{4.2}$$

Iz svojstava (4.1) i (4.2) može se zaključiti: ako se neka točka x nalazi u ravnini $H_{p/q}$, onda je ona bliže središtu grozda c_p nego grozda c_q. Vrijedi i obrnuto, ako se neka točka x nalazi u ravnini $H_{q/p}$ onda je ona bliže središtu grozda c_q nego grozda c_p. Drugo svojstvo govori ako se neka točka x nalazi na simetrali $B_{p|q}$, tad je ona jednako udaljena od središta grozdova c_p i c_q. U poglavlju 3.3 je pojašnjeno da algoritam koji koristi Voronojeve dijagrame za svaki objekt o_i iterativno provjerava leži li njegov MBR u potpunosti unutar neke od Voronojevih ćelija. Ako je taj uvjet zadovoljen objekt o_i se pridružuje grozdu čije je središte unutar iste ćelije, a ostali grozdovi se odbacuju bez računanja očekivane udaljenosti.

To je vremenski zahtjevna operacija, jer se mora provjeravati nalazi li se MBR unutar poligona koji čini Voronojevu ćeliju.

Nasuprot tome SPP algoritam za svaki objekt o_i provjerava leži li njegov MBR u potpunosti u ravnini $H_{p/q}$. Ako je taj uvjet zadovoljen grozd c_q se odbacuje kao kandidat za objekt o_i, a grozd c_p se uspoređuje s preostalim grozdovima. Prednost SPP algoritma je u tome što se odbačeni grozd c_q ne uspoređuje s preostalim grozdovima, jer je grozd c_p bliži od njega i on će odbaciti sve preostale grozdove koje bi odbacio i grozd c_q. Na taj način se broj usporedbi između grozdova znatno smanjuje, a time se smanjuje i složenost algoritma. Nakon što se obave sve usporedbe većina grozdova će biti odbačena kao potencijalni kandidati za objekt o_i.

U poglavlju 3.3 je dokazano da odbacivanje pomoću simetrala odbacuje veći broj grozdova nego odbacivanje pomoću algoritma MinMax. Algoritam koji koristi Voronojeve dijagrame i SPP algoritam odbacuju grozdove na sličnom principu, jer su Voronojeve ćelije ograničene sa simetralama. Budući da oba algoritma odbacuju grozdove koristeći slične principe, to znači da će ako jedan algoritam odbaci grozd to učiniti i drugi algoritam. Tako će SPP algoritam odbacivati više grozdova od algoritma MinMax što je i dokazano pokusima u petom poglavlju. Glavni nedostatak algoritma koji koristi Voronojeve dijagrame je sporiji proces odbacivanja grozdova nego kod algoritma MinMax, što umanjuje korist dobivenu od većeg broja odbačenih grozdova.

U SPP algoritmu predstavljen je način brze konstrukcije simetrala, te brži proces odbacivanja grozdova nego kod algoritma pomoću Voronojevih dijagrama. U algoritmu koji koristi Voronojeve dijagrame mora se provjeriti leži li MBR objekta o_i unutar Voronojeve ćelije. Budući da je Voronojeva ćelija poligon mora se provjeriti nalazi li se MBR objekta o_i unutar poligona, što je vremenski vrlo zahtjevno i provjerava se za svaku od Voronojevih ćelija dok se objekt ne pridruži nekom grozdu. Postupak je vrlo brz ako se grozd dodijeli na samome početku pa je obavljeno vrlo malo provjera. No za one objekte koji imaju veliki MBR i ne uspiju se dodijeliti niti jednom grozdu moraju se provjeriti sve ćelije. Za takve grozdove koji nisu odbačeni u metodi Voronojevih dijagrama mogu se koristiti simetrale kako bi se odbacili preostali grozdovi kandidati i smanjio broj računanja očekivanih udaljenosti. Algoritam temeljen na simetralnoj podjeli prostora nema potrebe za konstrukcijom Voronojevih dijagrama nego se samo računaju simetrale.

Za svaki par grozdova c_p i c_q u skupu grozdova $C = \{c_1, c_2, \dots, c_k\}$ simetrala $B_{p|q}$ računa se prema sljedećim jednadžbama:

$$a = -\frac{x_{cp} - x_{cq}}{y_{cp} - y_{cq}} \tag{4.3}$$

$$b = \frac{x_{cp}^2 - x_{cq}^2 + y_{cp}^2 - y_{cq}^2}{2(y_{cp} - y_{cq})} \tag{4.4}$$

$$B_{p|q} = ax + b \tag{4.5}$$

Za računanje simetrale $\mathrm{B_{p|q}}$ potrebne su samo koordinate točaka (x_{cp}, y_{cp}) i (x_{cq}, y_{cq}) koje predstavljaju središta grozdova c_p i c_q. Prvo se izračuna jednadžba pravca koji prolazi kroz točke (x_{cp}, y_{cp}) i (x_{cq}, y_{cq}), a iz jednadžbe pravca može se izračunati jednadžba simetrale koja prolazi polovicom dužine koja spaja središta grozdova i okomita je na njih. Ovakav način računanja simetrala brži je nego u algoritmu pomoću Voronojevih dijagrama, jer se tamo prvo računaju Voronojevi dijagrami pomoću poznatog „qhull“ algoritma, a onda se iz njih tek mogu izračunati simetrale. Osim bržeg računanja simetrala SPP algoritam ima brži proces odbacivanja grozdova. Provjeravanje s koje strane simetrale se nalazi MBR objekta o_i brže je nego provjeravanje nalazi li se MBR objekta o_i unutar Voronojeve ćelije što će se provjeriti pokusima u petom poglavlju.

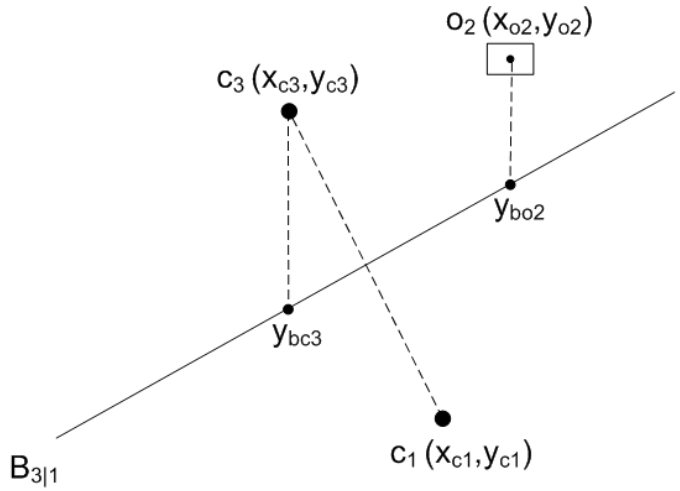
Za par grozdova c_p i c_q konstruira se simetrala $\mathrm{B_{p|q}}$ i provjerava na kojoj strani simetrale se nalazi MBR objekta o_i. Ako se MBR objekta o_i nalazi s iste strane simetrale kao grozd c_p, onda se grozd c_q odbacuje kao kandidat za objekt o_i. Također vrijedi suprotno, ako se MBR objekta o_i nalazi s iste strane simetrale kao grozd c_q, onda se grozd c_p odbacuje kao kandidat za objekt o_i. U slučaju da MBR objekta o_i siječe simetralu niti jedan od grozdova ne može biti odbačen, nego se nastavljaju uspoređivati s preostalim grozdovima kandidatima za objekt o_i. Na kraju svih usporedbi ako je samo jedan grozd ostao kao kandidat onda je proces odbacivanja okončan, a ako je ostalo više kandidata za njih je potrebno ja računati očekivanu udaljenost do objekta o_i. Kako bi se odbacio jedan od grozdova potrebno je napraviti projekcije koordinata grozdova c_p, c_q i MBR-a objekta o_i na simetralu $B_{p|q}$.

Za uspješno odbacivanje jednog od grozdova potrebno ja zadovoljiti sljedeći uvjet:

$$\left(y_{bcp} > y_{cp} \ \ and \ \ y_{boi} > y_{oi}\right) \ or \ \left(y_{bcp} < y_{cp} \ \ and \ \ y_{boi} < y_{oi}\right) \tag{4.6}$$

gdje je y_{cp} koordinata grozda c_p, y_{bcp} je y koordinata projekcije grozda c_p na pravac simetrale, y_{oi} je koordinata objekta o_i, a y_{boi} je y koordinata projekcije objekta o_i na pravac simetrale $B_{p|q}$.

Jednadžbom (4.6) može se provjeriti nalazi li se MBR objekta o_i na istoj strani simetrale $B_{p|q}$ kao grozd c_p, i ako je uvjet zadovoljen grozd c_q se odbacuje kao kandidat za objekt o_i. Primjer odbacivanja grozdova pomoću simetralne podjele prostora prikazan je na slici 4.1.

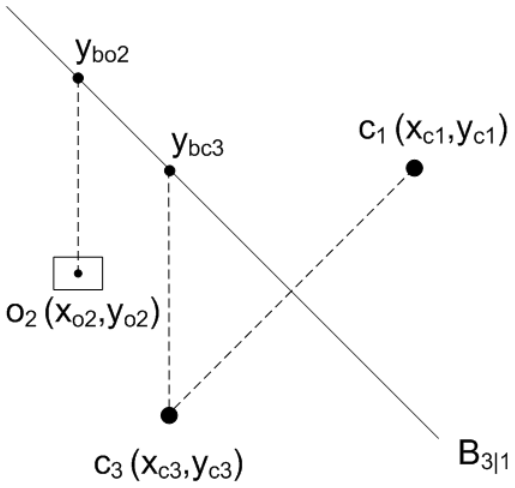


Slika 4.1 Prvi primjer odbacivanja grozdova simetralnom podjelom prostora

Kao što je prikazano na slici 4.1 koordinata x_{c3} grozda c_3 uvrštava se u jednadžbu pravca simetrale i kao rezultat dobije se koordinata y_{bc3}. Isto tako koordinata x_{o2} objekta o_{o2} uvrštava se u jednadžbu pravca simetrale i kao rezultat se dobije koordinata y_{bo2}. Radi jednostavnosti prikazano je samo uvrštavanje koordinate koja predstavlja središte MBR-a, no u praksi i pokusima u petom poglavlju uvrštavaju se i četiri koordinate s vrhova MBR-a. Na taj način osigurava se da MBR ne siječe simetralu. Ako su neke koordinate s ruba MBR-a na jednoj strani simetrale, a neke na drugoj strani simetrale, tad se može zaključiti da MBR siječe simetralu i niti jedan od grozdova ne može biti odbačen. U tom slučaju oba grozda trebaju se uspoređivati se preostalim grozdovima sve dok ne budu odbačeni.

Nakon izračuna projekcije grozdova i objekta na simetralu provjerava se je li zadovoljen uvjet za odbacivanje grozdova prema jednadžbi (4.6). Sa slike 4.1 se vidi da je zadovoljeni drugi *ili* uvjet jednadžbe (4.6). Vidi se da je y_{bc3} koordinata projekcije grozda na simetralu manja od y_{c3} koordinate grozda, isto tako y_{bo2} koordinata projekcije objekta na simetralu je manja od y_{o2} koordinate objekta. Budući da je drugi uvjet zadovoljen grozd c_1 se može odbaciti kao kandidat za objekt o_2.

Suprotna situacija, u kojoj su dobivene koordinate veće pa je zadovoljen prvi uvjet jednadžbe (4.6) prikazana je na slici 4.2. Vidi se da je y_{bc3} koordinata projekcije grozda na simetralu veća od koordinate grozda y_{c3}, isto tako y_{bo2} koordinata projekcije objekta na simetralu je veća od koordinate objekta y_{o2}. Budući da je sada zadovoljen prvi uvjet jednadžbe (4.6) grozd c_1 se može odbaciti kao kandidat za objekt o_2. Nakon usporedbe svih preostalih parova grozdova većina grozdova je odbačena kao potencijalni kandidati za objekt o_2. Ako je među kandidatima ostalo više od jednog grozda potrebno je izračunati očekivane udaljenosti za preostale grozdove, kako bi se objekt o_2 dodijelio grozdu sa najmanjom očekivanom udaljenosti.



Slika 4.2 Drugi primjer odbacivanja grozdova simetralnom podjelom prostora

U slučaju da u skupu $C = \{c_1, c_2, \dots, c_k\}$ postoji 50 grozdova tada je najveći broj usporedbi koje se između njih mogu napraviti jednak $50 \times 50 = 2500$. Uspoređivanjem svih grozdova broj usporedbi bi bio prevelik i proces odbacivanja grozdova bio bi spor. Tu nastupa glavna prednost algoritma temeljenog na simetralnoj podjeli prostora. U slučaju kada se jedan od grozdova odbaci kao kandidat za objekt o_i, ne moraju se praviti usporedbe između

odbačenog grozda i preostalih grozdova iz skupa $C = \{c_1, c_2, \dots, c_k\}$. Na primjer, ako se grozd c_q odbaci kao kandidat u usporedbi s grozdom c_p, nema potrebe za usporedbom između grozda c_q i preostalih grozdova kandidata. Budući da je grozd c_p pomoću jednadžbe (4.6) dokazano bliži objektu o_i, on će odbaciti sve one grozdove koje bi odbacio i grozd c_q u kasnijim usporedbama. Stoga se rade samo usporedbe između grozda c_p i preostalih grozdova, sve do trenutka kad i on bude odbačen. Broj usporedbi koje se ne moraju napraviti ovisan je o broju uspješno odbačenih grozdova, jer ako grozd nije odbačen za njega se moraju raditi usporedbe s preostalim grozdovima sve dok ne bude odbačen. Broj izbjegnutih usporedbi koje se ne moraju napraviti N_{ius} opisan je sljedećom jednadžbom:

$$N_{ius} = \sum_{i=1}^{n} N_{ogr}(k-1) \tag{4.7}$$

gdje je N_{ogr} broj odbačenih grozdova za neki objekt o_i, n je broj objekata, a k je ukupan broj grozdova.

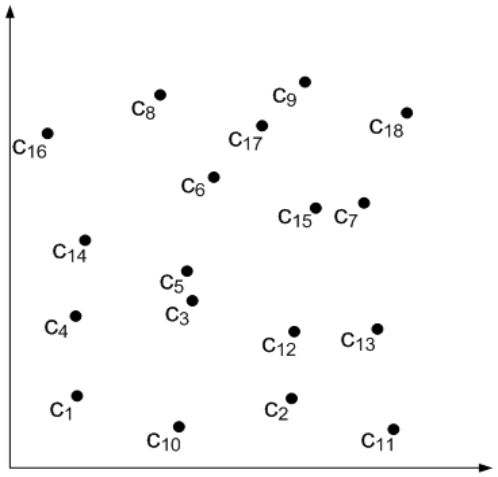
Na taj način značajno je skraćen broj usporedbi koje se moraju napraviti i ubrzan proces odbacivanja grozdova. Budući da usporedbe pripadaju u proces odbacivanja grozdova i oduzimaju značajan dio vremena dobiveno je skraćenje vremena razvrstavanja. Razvrstavanje temeljeno na simetralnoj podjeli prostora opisano je sljedećim pseudo kodom:

```
Izračunati simetrale za sve grozdove C={c1, ... , cm}
Za svaki grozd cp ,cq  činiti
  ako je MBRi na istoj strani simetrale Bp/q kao grozd
  cp
      ukloni cq kao kandidata
      ukloni cq iz iteracijske petlje (for)
  u suprotnom ako je MBRi na istoj strani simetrale
  kao grozd cq
      ukloni cp kao kandidata
      ukloni cp iz iteracijske petlje (for)
Za sve preostale grozdove izračunati ED
```

Slika 4.3 Pseudo kod algoritma temeljenog na simetralnoj podjeli prostora

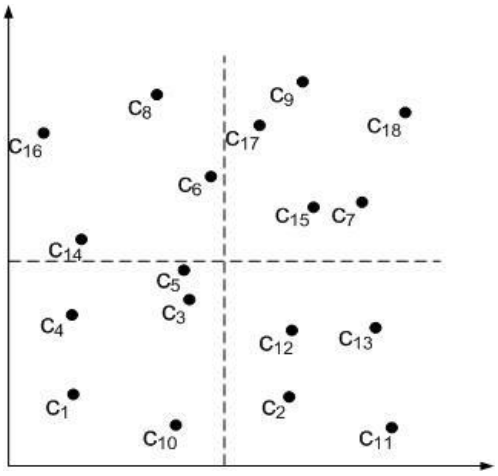
4.2. Podjela područja skupa objekata određivanjem prostornih odnosa objekata

Podjela ukupnog područja skupa objekata (eng. Segmentation of Data Set Area -SDSA), kao što i samo ime kaže, dijeli područje skupa objekata na male pravokutnike jednakih površina. U sljedećih nekoliko koraka pojašnjen je proces podjele. Na slici 4.4 prikazano je ukupno područje skupa objekata i raspored grozdova u području. Zbog jednostavnosti objekti nisu prikazani, jer je njihov broj znatno veći i slika bi postala nejasna. No bitno je napomenuti da su objekti raspoređeni po čitavom području skupa objekata kao i grozdovi.



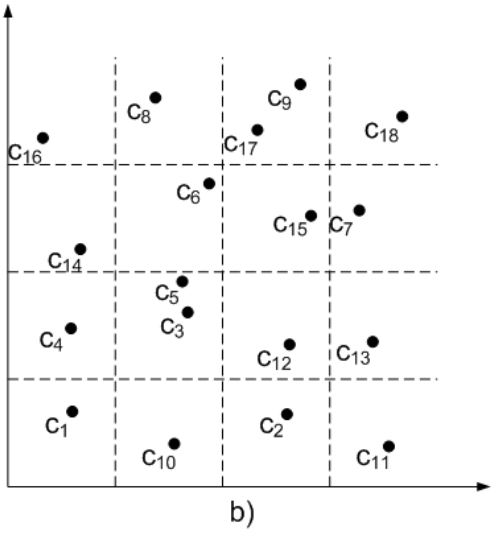
Slika 4.4 Ukupno područje skupa objekata s prikazanim grozdovima

U idućem koraku ukupno područje skupa objekata podijeljeno je na četiri dijela jednakih površina kao što je prikazano na slici 4.5.



Slika 4.5 Podjela ukupnog područja skupa objekata na četiri jednaka pravokutna područja

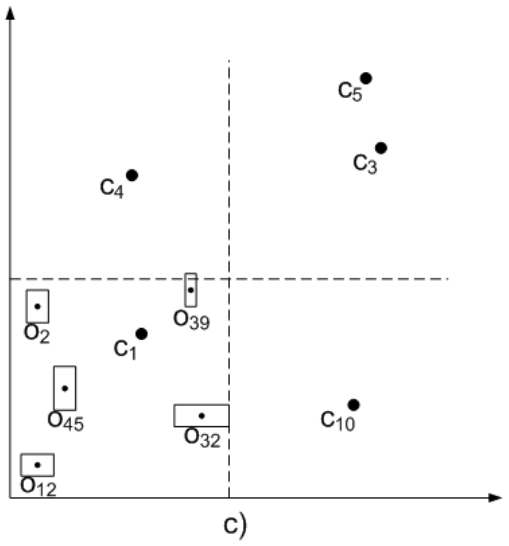
U idućem koraku se svaki od četiri dobivena dijela sa slike 4.5 dijeli na četiri nova i jednaka dijela kao što je prikazano na slici 4.6. Ukupno područje skupa objekata sada je podijeljeno na 16 jednakih dijelova. To će biti i konačan broj, jer je nužno da se u svakom pravokutniku nalazi barem jedan grozd, no poželjno je da ih ima i više kako bi se osiguralo ispravno razvrstavanje što će biti objašnjeno kasnije. Nakon podjele grozdovi i objekti podijeljeni su u 16 grupa, ovisno o tome u kojem se pravokutnom području nalaze.



Slika 4.6 Podjela ukupnog područja skupa objekata na 16 jednakih pravokutnih područja

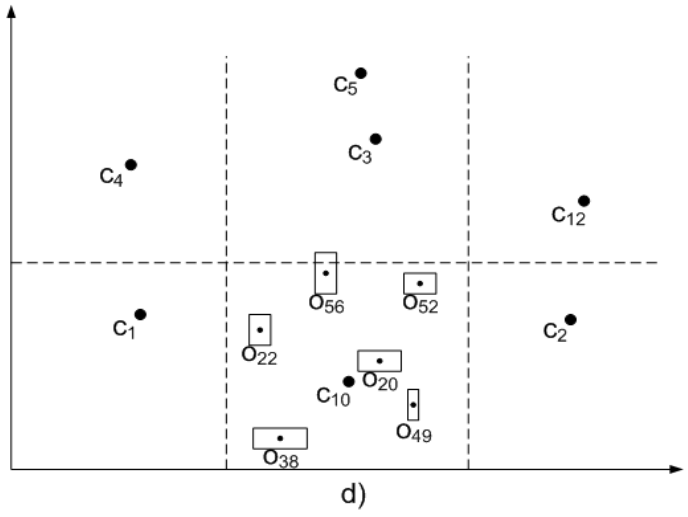
Broj pravokutnih područja na slici 4.6 jednak je 16, no broj pravokutnika na koje se dijeli područje skupa objekata nije fiksan. Broj pravokutnih područja ovisan je o broju objekata i grozdova u ukupnom području skupa objekata, te njihovim međusobnim odnosima. Osim o broju grozdova, broj pravokutnih područja ovisan je o položaju grozdova u promatranom pravokutnom području te o položaju grozdova u susjednim područjima. Razvrstavanje se obavlja tako da se svako pravokutno područje promatra odvojeno. Algoritam u prvom koraku uzima jedan pravokutnik i promatra samo objekte koji se nalaze u tome pravokutniku kao što je prikazano na slici 4.7. Budući da se promatraju samo objekti u jednom pravokutniku nije nužno promatrati sve grozdove nego samo grozdove iz tog i susjednih pravokutnika, što je zorno prikazano na slici 4.7. Ovisnost broja pravokutnika o broju i položaju grozdova proizlazi iz činjenice da je nužno da u svakom pravokutniku bude jedan grozd ili da grozdovi u susjednim pravokutnicima budu tako raspoređeni da osiguravaju ispravno razvrstavanje. Grozdovi moraju biti u određenom području zbog toga da niti jedan grozd iz vanjskih

pravokutnika, koji se trenutno ne promatraju, ne bude bliži bilo kojem objektu iz promatranog pravokutnika.



Slika 4.7 Razvrstavanje jednog pravokutnog područja s objektima i tri susjedna područja s grozdovima

Slika 4.8 prikazuje promatrano pravokutno područje i objekte unutar njega. U ovom slučaju promatrano pravokutno područje ima pet susjednih pravokutnih područja i nužno je promatrati grozdove iz tog i susjednih područja. Svi grozdovi koji se nalaze izvan promatranih područja automatski su odbačeni za sve objekte unutar promatranog područja bez računanja očekivane udaljenosti.

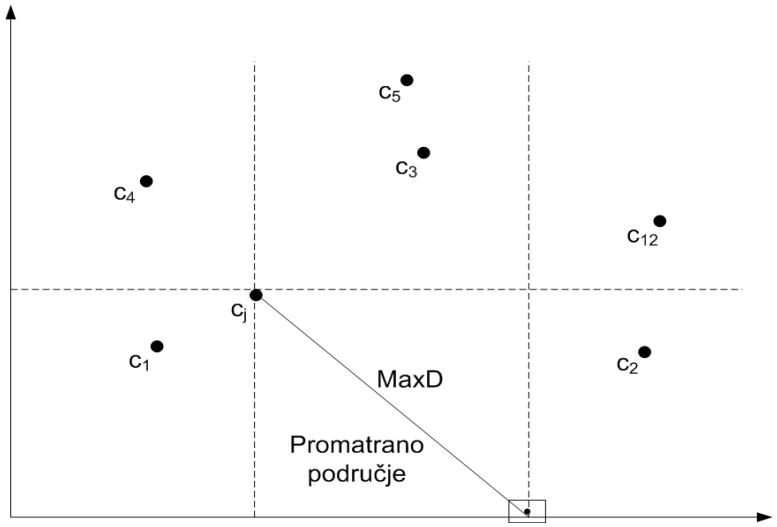


Slika 4.8 Razvrstavanje jednog pravokutnog područja s objektima i pet susjednih područja s grozdovima

Algoritam za podjelu područja skupa objekata mora osigurati da niti jedan grozd koji se ne nalazi u promatranim pravokutnicima, nazovimo ga vanjski grozd, ne bude bliži bilo kojem objektu u promatranom području od grozdova iz unutarnjih pravokutnih područja, nazovimo ih unutarnjim grozdovima. U algoritmu se mogu dogoditi dva sljedeća slučaja:

1. U promatranom pravokutnom području s objektima nalazi se barem jedan grozd.
2. U promatranom pravokutnom području s objektima se ne nalazi niti jedan grozd.

Ako se u promatranom pravokutnom području nalazi barem jedan grozd maksimalna udaljenost o tog grozda do objekata u promatranom pravokutniku opisana je jednadžbom (4.8). Na slici 4.9 prikazan je slučaj maksimalne udaljenosti grozda i objekata unutar promatranog pravokutnog područja.



Slika 4.9 Maksimalna udaljenost objekta i grozda unutar promatranog pravokutnog područja

Ako su dimenzije pravokutnika $a \times b$, i ako je $a \approx b$ što je zadovoljeno u većini primjena, tada je maksimalna udaljenost $MaxD(o_i, c_{un})$ između unutarnjeg grozda c_{un} i nekog objekta o_i u promatranom pravokutniku opisana sljedećom jednadžbom:

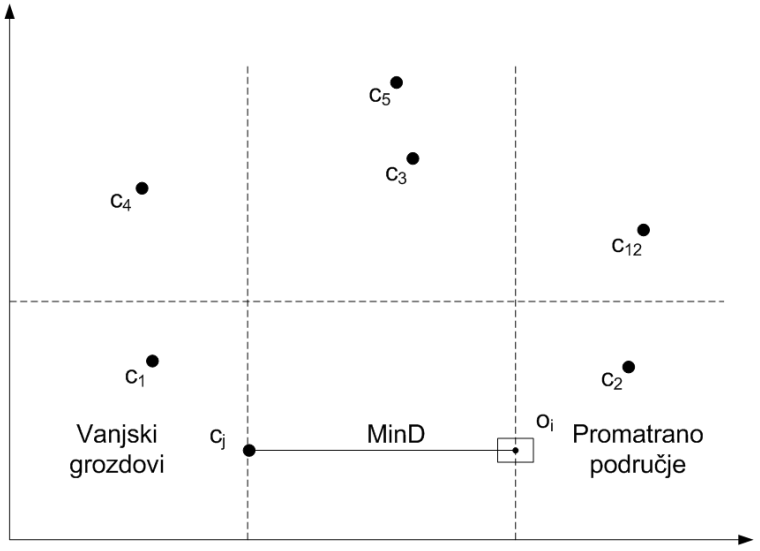
$$MaxD(o_i, c_{un}) = \sqrt{a^2 + a^2} = a\sqrt{2} \tag{4.8}$$

Minimalna udaljenost $MinD(o_i, c_{vanj})$ između vanjskog grozda c_{vanj} i nekog objekta o_i u promatranom pravokutniku opisana sljedećom jednadžbom:

$$MinD(o_i, c_{vanj}) = a \tag{4.9}$$

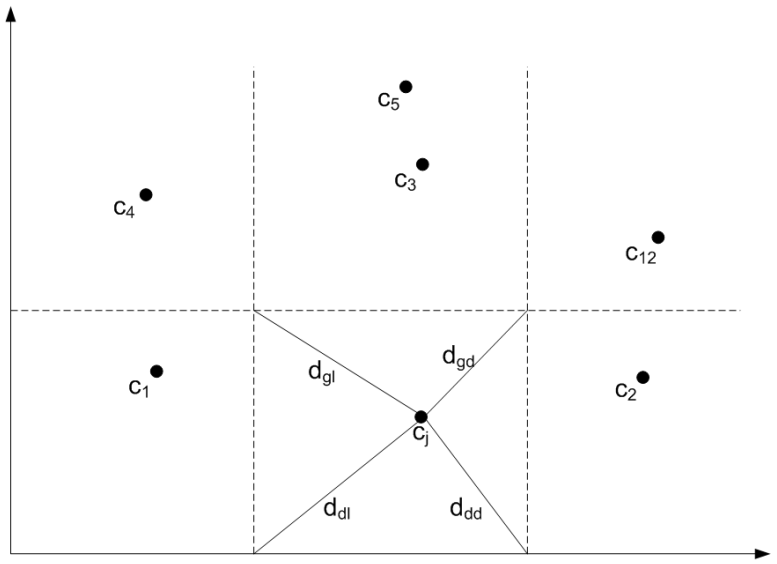
Da bi podjela područja skupa objekata bila ispravna maksimalna udaljenost jednog od unutarnjih grozdova do bilo kojeg objekta u promatranom pravokutniku mora biti manja od minimalne udaljenosti vanjskih grozdova:

$$MaxD(o_i, c_{un}) < MinD(o_i, c_{vanj}) = a \tag{4.10}$$



Slika 4.10 Minimalna udaljenost vanjskog grozda i objekta u promatranom pravokutnom području

Iz svega navedenog slijedi da najmanja maksimalna udaljenost jednog od unutarnjih grozdova c_{un} do bilo kojeg objekta o_i u promatranom pravokutniku mora biti manja od širine stranica pravokutnika a. Budući da se maksimalne udaljenosti pojavljuju u vrhovima pravokutnika potrebno je provjeriti udaljenosti od unutarnjih grozdova c_{un} do vrhova pravokutnika kao što je prikazano na slici 4.11. Ako postoji grozd u promatranom području provjeravaju se udaljenosti od vrhova područja do tog grozda, te udaljenosti od vrhova do grozdova iz susjednih pravokutnih područja. A ako ne postoji grozd u promatranom području provjeravaju se samo udaljenosti od vrhova do grozdova iz susjednih pravokutnih područja.



Slika 4.11 Udaljenosti od unutarnjeg grozda do vrhova promatranog pravokutnog područja

Za svaki od četiri vrha promatranog područja provjerava se udaljenost do grozda. Ako se pronađe grozd čija je udaljenost do vrha pravokutnika manja od a, prelazi se na sljedeći vrh. Uvjet za ispravno razvrstavanje je zadovoljen ako se u promatranom pravokutniku ili nekom od susjednih unutarnjih pravokutnika pronađe grozd čija je udaljenosti do vrha pravokutnika manja od a. Ako se za sva četiri vrha pronađu grozdovi čija udaljenost manja od a, može se zaključiti da je pravokutno područje ispravno podijeljeno. Nakon toga prelazi se na sljedeće područje i provjeravaju se isti uvjeti, i tako za sva područja. Ako sva pravokutna područja zadovolje uvjet da je udaljenost do svih vrhova u pravokutnom području do najbližeg unutarnjeg grozda manja od a, ukupno područje skupa objekata je ispravno podijeljeno. Budući da je broj grozdova značajno veći od broja pravokutnika, u svakom od njih nalazi se nekoliko grozdova (ovisno o broju i rasporedu grozdova te broju pravokutnika), pa su traženi uvjeti najčešće zadovoljeni. Problem se jedino javlja kad je broj grozdova približno jednak broju pravokutnika, pa je broj grozdova koji se nalaze u pravokutnim područjima jednak jedan. Ako uvjet nije zadovoljen za neko područje mogu se napraviti dva sljedeća rješenja. U prvom rješenju u promatranje se mora uzeti jedno vanjsko područje koje je najbliže promatranome vrhu, pa grozdovi unutar novog područja postaju unutarnji grozdovi. Time je povećan broj susjednih pravokutnika koji se uzimaju u promatranje, ali je osigurano ispravno razvrstavanje objekata.

Drugo rješenje koje se može poduzeti je smanjenje broja pravokutnika. Budući da je broj pravokutnika smanjen, broj grozdova u pojedinom pravokutnom području je veći i postoji veća vjerojatnost da će uvjet maksimalne udaljenosti do grozda u biti zadovoljen. No sa smanjenjem broja pravokutnika povećava se broj promatranih relacija između objekata i grozdova. Stoga je preporučljivo koristiti prvo rješenje. Ovisno o pojedinom slučaju i ovisno o broju i rasporedu grozdova odabire se optimalan broj pravokutnika. Povećanjem broja pravokutnih područja smanjuje se broj objekata i grozdova po pojedinom području, a time se smanjuje broj promatranja relacija između objekata i grozdova, pa se treba pronaći kompromis u broju područja. Na primjer, ako se u području skupa objekata nalazi 1600 objekata i 64 grozda tada je broj relacije između njih koji se može promatrati N_{uk} jednak:

$$N_{uk} = 1600 \times 64 = 102400 \tag{4.11}$$

Ako se koristi algoritam podjele područja skupa objekata i područje se podijeli na 16 pravokutnih područja, tad je prosječan broj objekata po području jednak 100, a prosječan broj grozdova po području jednak je četiri. Budući da se svako područje promatra odvojeno broj relacija između objekata i grozdova ovisi o tome radi li se o unutarnjem ili vanjskom pravokutnom području. Ako se radi o području koje se nalazi na vrhu ukupnog područja skupa objekata, broj susjednih pravokutnih područja jednak je tri kao što je prikazano na slici 4.7. U tome slučaju broj područja s grozdovima jednak je četiri, te je broj relacija između objekata i grozdova jednak:

$$N_{vrp} = nkp = 100 \times 4 \times 4 = 1600 \tag{4.12}$$

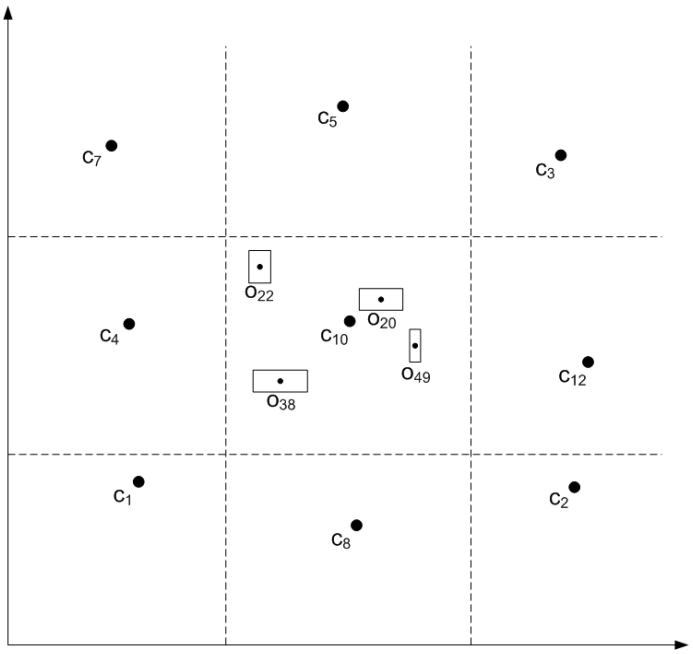
gdje je n broj objekata u pravokutnom području, k je prosječan broj grozdova po pravokutnom području i p je broj promatranih pravokutnih područja s grozdovima.

Ako se radi o vanjskom pravokutniku koji se nalazi na vanjskom dijelu područja, broj susjednih pravokutnika jednak je pet kao što je prikazano na slici 4.8. U tome slučaju broj pravokutnika s grozdovima jednak je šest, te je broj relacija između objekata i grozdova jednak:

$$N_{vp} = nkp = 100 \times 4 \times 6 = 2400 \tag{4.13}$$

Ako se radi o unutarnjem pravokutniku koji se nalazi u unutarnjem dijelu područja, broj susjednih pravokutnika jednak je osam kao što je prikazano na slici 4.12. U tome slučaju broj pravokutnika s grozdovima jednak je devet, te je broj relacija između objekata i grozdova jednak:

$$N_{up} = nkp = 100 \times 4 \times 9 = 3600 \tag{4.14}$$



Slika 4.12 Razvrstavanje jednog pravokutnog područja s objektima i osam susjednih područja s grozdovima.

Ukupno područje skupa objekata podijeljeno na šesnaest pravokutnih područja. Slijedi da je ukupno područje skupa objekata podijeljeno tako da ima četiri vršna pravokutna područja, osam vanjskih i četiri unutarnja pravokutna područja. Slijedi da je ukupan broj relacija između objekata i grozdova jednak:

$$N_{uk} = 4N_{vrp} + 8N_{vp} + 4N_{up} = 40000 \tag{4.15}$$

Vidimo da je podjelom područja skupa objekata ukupan broj relacija smanjen s 102400 na 40000, što je značajno smanjenje.

Iz svega navedenog može se navesti nekoliko prednosti algoritma podjele ukupnog područja skupa objekata:

1. Podjelom se smanjuje broj relacija između objekata i grozdova koje se moraju promatrati.
2. Grozdovi izvan promatranih pravokutnih područja odbačeni su za sve objekte unutar pravokutnika bez računanja očekivanih udaljenosti.
3. Algoritam se kombinirati s postojećim algoritmima za razvrstavanje tako da se oni primjene nad pojedinim pravokutnim područjem.
4. Pravokutna područja međusobno su nezavisna.
5. Budući da su pravokutna područja nezavisna razvrstavanje pojedinog područja može se provoditi nezavisno o ostalim područjima. To otvara mogućnost za paralelno razvrstavanje.
6. Ovisno o računalu svako pravokutno područje može se razvrstavati kao poseban proces ako računalo ima dovoljno jezgri. A ako nema dovoljno jezgri, svaka jezgra procesira jednak broj područja.
7. Paralelnim procesiranjem se uz isti hardver vrijeme izvođenja maksimalno se može skratiti onoliko puta koliki je broj paralelnih procesa.

Razvrstavanje pomoću podjele ukupnog područja skupa objekata određivanjem prostornih odnosa objekata opisano je sljedećim pseudo kodom:

Za svako pravokutno područje činiti
  za svaki grozd c_p ,c_q unutar područja činiti
    ako je MBR_i na istoj strani simetrale $B_{p/q}$ kao grozd c_p
      tada ukloni c_q kao kandidata
    u suprotnom ako je MBR_i na istoj strani simetrale $B_{p/q}$ kao grozd c_q
      tada ukloni c_p kao kandidata
Za sve ostale kandidate izračunati ED

Slika 4.13 Pseudo kod za serijsko izvođenje algoritma SDSA

Pseudo kod sa slike 4.13 predstavlja serijsko izvođenje algoritma SDSA, dok je paralelno izvođenje predstavljeno sljedećim pseudo kodom:

Za svako pravokutno područje činiti
  ako postoji slobodna jezgra
    razvrstaj područje kao paralelan proces
  u suprotnom
    dodaj područje postojećem procesu
  za svaki grozd c_p ,c_q unutar područja činiti
    ako je MBR_i na istoj strani simetrale $B_{p/q}$ kao
    grozd c_p
      tada ukloni c_q kao kandidata
    u suprotnom ako je MBR_i na istoj strani simetrale
      $B_{p/q}$ kao grozd c_q
      tada ukloni c_p kao kandidata
Za sve ostale kandidate izračunati ED

Slika 4.14 Pseudo kod za paralelno izvođenje algoritma SDSA

U petom poglavlju provedeni su pokusi koji koriste algoritam SDSA za razvrstavanje nesigurnih objekata. U pokusima se algoritam SDSA kombinira s dva algoritma za razvrstavanje nesigurnih objekata. Kombinacijom s algoritmom MinMax nastao je novi SDSA-MinMax algoritam, a kombinacijom s algoritmom SPP koji je također razvijen u ovoj disertaciji nastao je novi SDSA-SPP algoritam. Pokusima u petom poglavlju je dokazano da algoritam SDSA poboljšava svojstva MinMax i SPP algoritma.

Glavno svojstvo algoritma SDSA, a to je mogućnost paralelne obrade, također je provjereno u petom poglavlju. Serijski algoritam prilagođen je za paralelno izvođenje na dvije i četiri jezgre. Uspoređena su vremena izvođenja procesa razvrstavanja kao serijskog procesa pomoću algoritma SDSA-SPP, kao paralelnog procesa na dvije jezgre pomoću algoritma SDSA-SPP2 i na samome kraju kao paralelnog procesa na četiri jezgre pomoću algoritma SDSA-SPP4. Za očekivati je da će se najbolja vremena dobiti paralelnim procesiranjem na četiri jezgre. Pokusima u petom poglavlju provjereni su svi algoritmi razvijeni u ovoj disertaciji i uspoređeni s trenutno najboljim algoritmima.

5. Pokusi i analiza rezultata

Pokusi su provedeni kako bi se pokazalo da algoritmi predloženi u ovoj disertaciji imaju kraća vremena izvođenja procesa razvrstavanja od postojećih algoritama. Najbolji postojeći algoritmi s najkraćim vremenom izvođenja bili su algoritam MinMax i algoritam koji koristi Voronojeve dijagrame. Algoritam MinMax ima brži proces odbacivanja grozdova nego algoritam koji koristi Voronojeve dijagrame, ali je broj uspješno odbačenih grozdova manji. Na vrijeme izvođenja tih dvaju algoritama utječe računanje očekivane udaljenosti. Ako računanje očekivane udaljenosti zahtijeva više vremena algoritam pomoću Voronojevih dijagrama biti će brži od algoritma MinMax. Budući da računanje očekivane udaljenosti ovisi o broju uzoraka kojim je predstavljena funkcija gustoće vjerojatnosti PDF, o njima će ovisiti i vrijeme izvođenja pojedinog algoritma. Ako je broj uzoraka veći računanje očekivane udaljenosti je dulje i preporučuje se koristiti Voronojeve dijagrame. Za manji broj uzoraka preporučuje se koristiti algoritam MinMax. Pokusima je provjereno koji algoritam daje bolje rezultate za različit broj objekata, grozdova, veličinu minimalnog područja nesigurnosti i broj uzoraka kojim je predstavljen PDF.

Pokusima je također uspoređen algoritam SPP razvijen u ovoj disertaciji s navedenim postojećim algoritmima. Pokazano je kako SPP algoritam znatno skraćuje vrijeme izvođenja procesa razvrstavanja. Kako bi se napravila dodatna poboljšanja u procesu razvrstavanja u ovoj disertaciji predstavljen je i drugi postupak koji koristi podjelu područja skupa objekata (eng. Segmentation of Data Set Area- SDSA). Kao što je objašnjeno u ranijim poglavljima on dijeli područje skupa objekata na pravokutnike jednakih površina i na svaki pravokutnik primjenjuje neki od algoritama za razvrstavanje. Na primjer, može se kombinirati s algoritmom MinMax, algoritmom koji koristi Voronojeve dijagrame ili algoritmom temeljenom na simetralnoj podjeli prostora. Algoritam SDSA smanjuje broj relacija između objekata i grozdova koje treba promatrati i dodatno poboljšava proces odbacivanja grozdova i skraćuje vrijeme izvođenja. Budući da sam proces podjele ima udio u ukupnom vremenu, podjela se isplati samo u onim uvjetima kada je broj objekata i grozdova dovoljno velik. Za manji broj grozdova i objekata SDSA algoritam ne daje značajna poboljšanja u vremenu izvođenja, jer je udio podjele u ukupnom vremenu utjecajan. Pokusima je provjereno koji je minimalan broj objekata i grozdova u kojima se isplati koristiti algoritam podjele područja skupa objekata. Algoritam SDSA je kombiniran s ranije navedenim algoritmima za

razvrstavanje i uspoređena su vremena izvođenja tih algoritama kad se koristi podjela ukupnog područja skupa objekata i kad se ona ne koristi.

Najvažnija prednost koju donosi algoritam SDSA je mogućnost razvrstavanja kao više paralelnih procesa. Ranije je rečeno da su pravokutna područja međusobno nezavisna i da se svako od njih može razvrstavati nezavisno o ostalim pravokutnim područjima. U ovom radu područje skupa objekata podijeljeno je na šesnaest pravokutnih područja, pa se razvrstavanje može napraviti kao šesnaest nezavisnih paralelnih procesa. No zbog ograničenja u računalnoj moći, jer je korišteno računalo s četiri jezgre, proces razvrstavanja se izvodi kao četiri paralelna procesa. Bitno je napomenuti da ovakav način paralelnog razvrstavanja ne zahtijeva dodatne troškove, jer se koristi isto računalo kao i kod serijskih procesa. Algoritam SDSA koristi prednosti novih procesora koji imaju više jezgri i imaju mogućnost paralelnog izvođenja. Još značajnija ubrzanja mogla bi se dobiti korištenjem klaster ili grid računala, gdje je broj paralelnih procesa koji se mogu pokrenuti jako velik, pa se svaki pravokutnik može razvrstavati kao paralelni proces. Ako se proces razvrstavanja odvija kao četiri paralelna procesa ubrzanje vremena izvođenja prema Ahmdalovom zakonu najviše je četiri puta, no u praktičnim primjenama ono će biti manje, jer se dio vremena troši na komunikaciju između glavnog programa i paralelnih procesa i prebacivanje podataka iz paralelnih procesa u glavni program.

Pokusi su provedeni za sve kombinacije algoritma SDSA kao serijskog procesa, te istog algoritma kao paralelnog procesa na dvije jezgre i četiri jezgre. Mjerena su vremena izvođenja za različit broj objekata, grozdova, veličinu minimalnog područja nesigurnosti i broj uzoraka. Pokusima je dokazano kako se paralelnim razvrstavanjem dobiju značajna poboljšanja u vremenu izvođenja u odnosu na serijski proces. Za sve pokuse korišten je osnovni skup objekata koji se sastoji od n objekata. Svi objekti smješteni su u dvodimenzionalnom prostoru s koordinatama $[0{,}100] \times [0{,}100]$. Svaki objekt opisan je s minimalnim područjem nesigurnosti MBR koje je ograničeno s maksimalnom duljinom stranica. Za svaki nesigurni objekt duljine stranica minimalnog područja nesigurnosti generirane su slučajnim odabirom, s tim da je maksimalna duljina stranice ograničena. MBR svakog objekta podijeljen je na $\sqrt{s} \times \sqrt{s}$ pravokutnih ćelija, gdje je s broj uzoraka koji se koristi za predstavljanje funkcije gustoće vjerojatnosti PDF. Svaki uzorak ima vjerojatnost da se objekt nalazi na tom uzorku i zbroj vjerojatnosti svih uzoraka mora biti jedan. Objekti su raspoređeni po ukupnom području skupa objekata i među njima se mogu odabrati predstavnici za početni skup od k grozdova.

Kako bi se povećala točnost, pokusi se ponavljaju 20 puta i kao konačan rezultat uzima se srednja vrijednost svih rezultata. Rezultati koji su dobiveni različitim algoritmima međusobno se uspoređuju kako bi se osiguralo da algoritmi daju isti rezultat i osigurala ispravnost razvrstavanja. Pokusi su provedeni u MATLAB-u i na računalu PC s procesorom Intel Core i7-870 2.93GHz i s 4GB radne memorije. Za potrebe pokusa korišten je osnovni skup parametara koji je prikazan u tablici 5.1, i na njima su provedeni osnovni pokusi. Vrijednosti osnovnog skupa parametara slične su kao u ostalim pokusima s razvrstavanjem nesigurnih objekata [56]. Za ostale pokuse mijenja se jedan od parametara dok ostali parametri zadržavaju osnovne vrijednosti.

Tablica 5.1 Osnovne vrijednosti svih parametara korištenih u pokusima

Parametri	**Opis**	**Vrijednost**
n	Broj nesigurnih objekata	10000
k	Broj grozdova	49
d	Maksimalna duljina stranica MBR-a	10
s	Broj uzoraka kojim je predstavljen PDF	196

Nakon provedenih pokusa nad osnovnim skupom parametara prikazanih u tablici 5.1 i mjerenja vremena izvođenja različitih algoritama, provedeni su i pokusi u kojima se mijenja broj objekata, a svi ostali parametri su zadržali osnovne vrijednosti. Sljedeći pokusi su oni u kojima se mijenjao broj grozdova, a svi ostali parametri zadržali su osnovne vrijednosti. Isti postupak se ponavlja i za preostale parametre. Mijenjala se veličina područja nesigurnosti, a ostali parametri ostajali su isti pa se mijenjao broj uzoraka kojima je predstavljena funkcija gustoće vjerojatnosti, a ostali parametri su ostali isti. Predloženim pokusima provjerena su vremena izvođenja procesa razvrstavanja pojedinih algoritama u različitim situacijama. Nastojalo se provjeriti koji algoritam ima najbolja svojstva u pojedinim primjenama. Pokusima se trebalo dokazati kako algoritam temeljen na simetralnoj podjeli prostora ima bolje rezultate nego algoritam MinMax i algoritam koji koristi Voronojeve dijagrame. Nadalje, kombiniranje navedenih algoritama s algoritmom SDSA dovelo je do kraćeg vremena izvođenja. Na kraju su provedeni pokusi s paralelnim procesiranjem u kojima je

dobiveno dodatno skraćenje vremena izvođenja procesa razvrstavanja. Skraćenje je naravno ovisno o broju paralelnih procesa.

Implementacija svih pokusa provedena je pomoću programskog koda danog u prilozima na CD-u. Najprije su provedeni pokusi koji uspoređuju vremena razvrstavanja triju različitih algoritama. Uspoređeni su algoritam MinMax, algoritam koji koristi Voronojeve dijagrame i algoritam temeljen na simetralnoj podjeli prostora. Za svaki od algoritama napravljene su MATLAB m-datoteke. U datoteci *MinMaxPrunning.m* nalazi se programski kod algoritma MinMax, u datoteci *VoronoiPrunning.m* nalazi se programski kod algoritma koji koristi Voronojeve dijagrame i u datoteci SPP_Prunning.m programski kod algoritma temeljenog na simetralnoj podjeli prostora. Sva tri algoritma primaju parametre navedene u tablici 5.1 i mogu koristiti simulirani ili stvarni skup prostornih podataka. Pokusi u petom poglavlju koriste simulirani skup podataka, dok pokusi u šestom poglavlju koriste stvarni skup podataka koji se sastoji od zemljopisnih koordinata u gradu Osijeku.

Za provedbu pokusa sa algoritmom SDSA napravljene su dvije MATLAB m-datoteke koje uspoređuju vremena razvrstavanja algoritma MinMax i algoritma SPP u slučaju kad se oni kombiniraju s algoritmom za podjelu područja skupa objekata. Datoteke su dane kao prilog na CD-u. U datoteci *SDSAMinMaxPrunning.m* nalazi se programski kod algoritma SDSA-MinMax nastalog kombinacijom algoritma SDSA i algoritma MinMax, a u datoteci *SDSA_SPP_Prunning.m* nalazi se programski kod algoritma SDSA-SPP nastalog kombinacijom algoritma SDSA i algoritma SPP.

Za provedbu pokusa koji uspoređuju vremena razvrstavanja algoritma SDSA-SPP kao serijskog procesa, te istog algoritma kada se izvodi kao paralelni proces na dvije i četiri jezgre napravljene su još dvije MATLAB m-datoteke. Datoteke su također dane kao prilog na CD-u. U datoteci *SDSA_SPP_Prunning_2.m* nalazi se programski kod algoritma koje se izvodi kao paralelni proces razvrstavanja na dvije jezgre. Algoritam je nazvan SDSA-SPP2. U datoteci *SDSA_SPP_Prunning_4.m* nalazi se programski kod algoritma koje se izvodi kao paralelni proces razvrstavanja na četiri jezgre. Algoritam je nazvan SDSA-SPP4. Rezultati su uspoređeni sa serijskim algoritmom koji je ranije spomenut i koji se nalazi u *SDSA_SPP_Prunning.m* datoteci.

5.1. Pokusi s različitim algoritmima za razvrstavanje

U ovom poglavlju provedeni su pokusi koji uspoređuju vremena razvrstavanja algoritma MinMax, algoritma koji koristi Voronojeve dijagrame i algoritma temeljenog na simetralnoj podjeli prostora. Najprije su provedeni pokusi s osnovnim skupom parametara i uspoređeni rezultati triju algoritama za razvrstavanje. Nakon toga provedeni su pokusi u kojima se mijenja jedan od parametara, dok ostali parametri zadržavaju osnovne vrijednosti.

5.1.1. Pokusi s osnovnim skupom parametara

Pokusi su provedeni s osnovnim parametrima prikazanima u tablici 5.1. Ponovljeni su dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja i rezultati su prikazani u tablici 5.2. U tablici su prikazani naziv algoritma, vrijeme izvođenja u sekundama i NED koji predstavlja broj računanja očekivane udaljenosti po objektu po iteraciji. Za odbačene grozdove ne računa se očekivana udaljenost što smanjuje NED. Kod algoritma uk-means računa se očekivana udaljenost između svakog objekta i svakog grozda, pa je broj računanja očekivanih udaljenosti po objektu po iteraciji jednak broju grozdova, to jest $\mathrm{NED} = 49$. Iz tablice je vidljivo da je NED kod promatranih algoritama znatno manji, pa su oni znatno brži nego algoritam uk-means. NED je smanjen s 49 na 1.307 što predstavlja smanjenje od 97.332%. Kraće vrijeme izvođenja procesa razvrstavanja bitno je u brojnim primjenama kao što su senzorske mreže, nadzor zračnog prometa i nadzor hitne pomoći. U tim primjenama je vrijeme kritično i potrebno je što prije dobiti rezultate razvrstavanja.

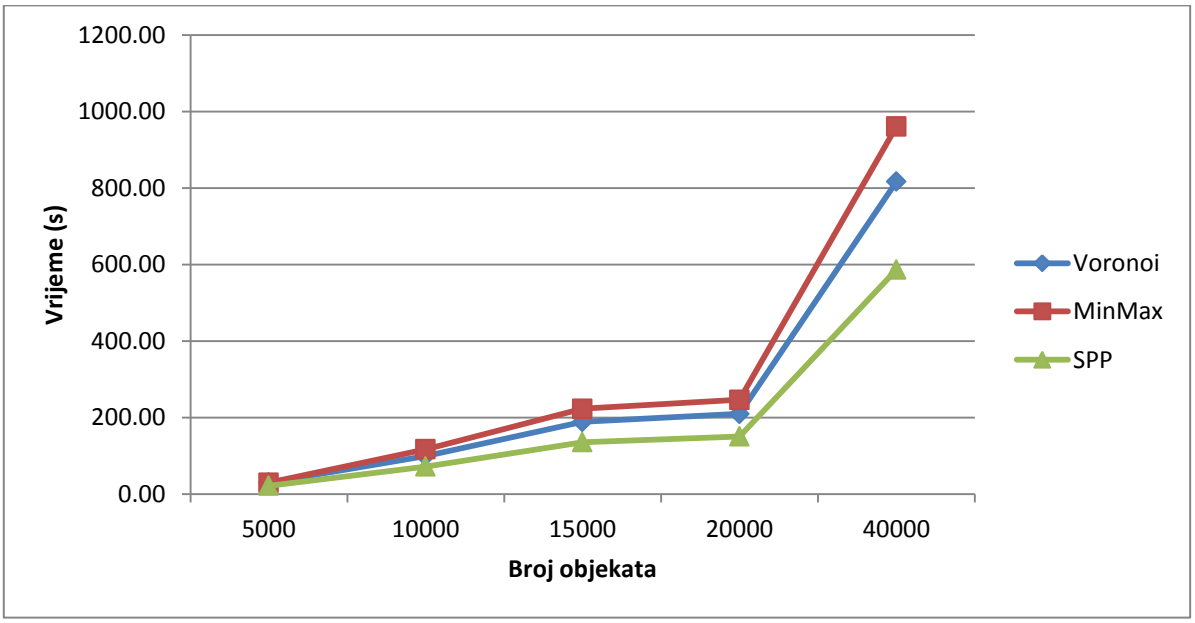
Tablica 5.2 Vremena izvođenja i NED triju algoritama korištenjem osnovnog skupa parametara prikazanog u tablici 5.1

Naziv	**Vrijeme izvođenja (s)**	**NED**
Algoritam MinMax	117.206	2.112
Algoritam koji koristi Voronojeve dijagrame	99,896	1.307
Algoritam temeljen na simetralnoj podjeli prostora-SPP	71.965	1.307

Algoritam pomoću Voronojevih dijagrama i algoritam temeljen na simetralnoj podjeli prostora dijele sličan princip za odbacivanje grozdova pa im je NED jednak. Iz tablice 5.1 se vidi da je algoritam SPP brži od algoritma koji koristi Voronojeve dijagrame, jer ima brži proces odbacivanja grozdova. Algoritam pomoću Voronojevih dijagrama mora provjeravati nalazi li se MBR nekog objekta unutar poligona koji čini Voronojevu ćeliju, što je vremenski zahtjevnije nego promatranje projekcije vrhova MBR-a na simetralu. Algoritam MinMax ima veći NED, jer odbacuje manji broj grozdova od druga dva algoritma. Ranije je u poglavlju 3.2 pokazano da algoritam MinMax ima slabija svojstva pri odbacivanju grozdova. Zbog toga se u algoritmu MinMax mora računati veći broj očekivanih udaljenosti po objektu po iteraciji pa je njegovo vrijeme izvođenja dulje nego kod algoritma SPP.

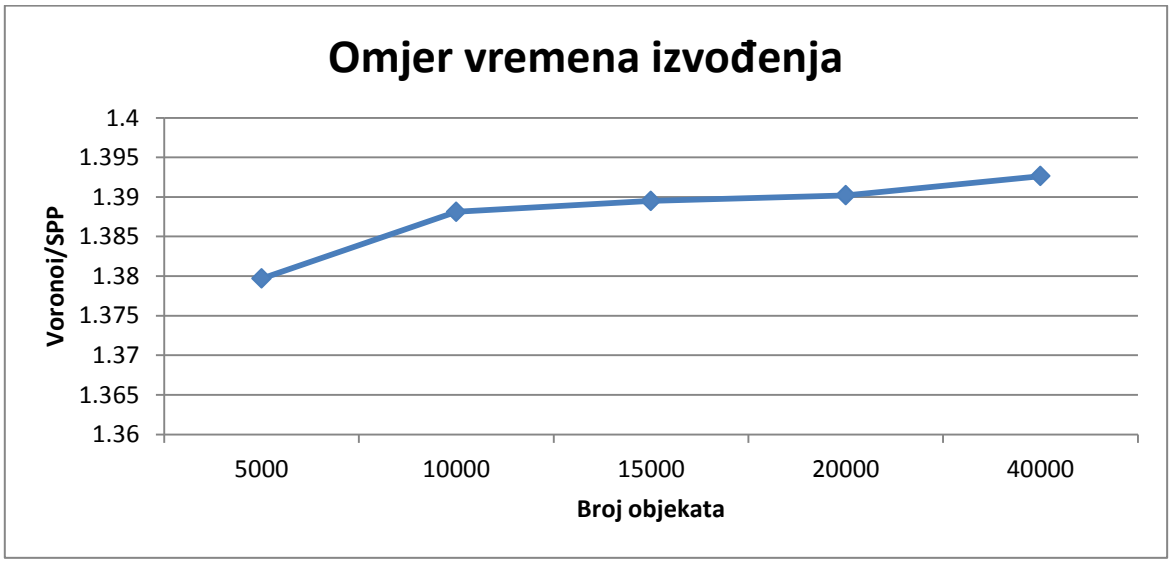
5.1.2. Pokusi u kojima se mijenja broj objekata

U ovim pokusima mijenja se broj objekata dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju sa 5000 objekata i završavaju sa 40000 objekata. Za svaki broj objekata pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.1.



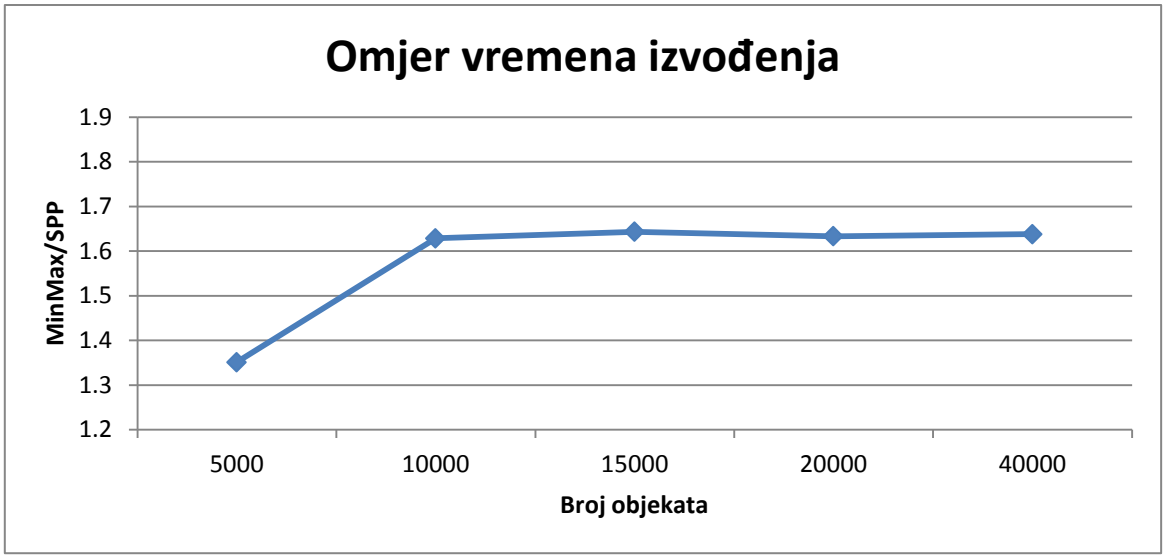
Slika 5.1 Vremena izvođenja triju algoritama za različit broj objekata

Sa slike 5.1 je vidljivo da algoritam temeljen na simetralnoj podjeli prostora u svim slučajevima ima kraća vremena izvođenja od preostala dva algoritma. Povećanjem broja objekata povećava se i broj relacija između objekata i grozdova koje treba promatrati, pa SPP algoritam pokazuje bolja svojstva, jer ima najbrži proces odbacivanja grozdova. Na slici 5.2 prikazan je omjer vremena izvođenja algoritma koji koristi Voronojeve dijagrame i algoritma temeljenog na simetralnoj podjeli prostora. Sa slike je vidljivo da se s povećanjem broja objekata povećava njihov omjer, što znači da algoritam SPP brže razvrstava veće skupove objekata. Algoritam SPP ima bolja vremena izvođenja u čitavom rasponu broja objekata, što ga čini boljim u primjenama s većim brojem objekata.



Slika 5.2 Omjer vremena izvođenja algoritma koji koristi Voronojeve dijagrame i algoritma SPP za različit broj objekata

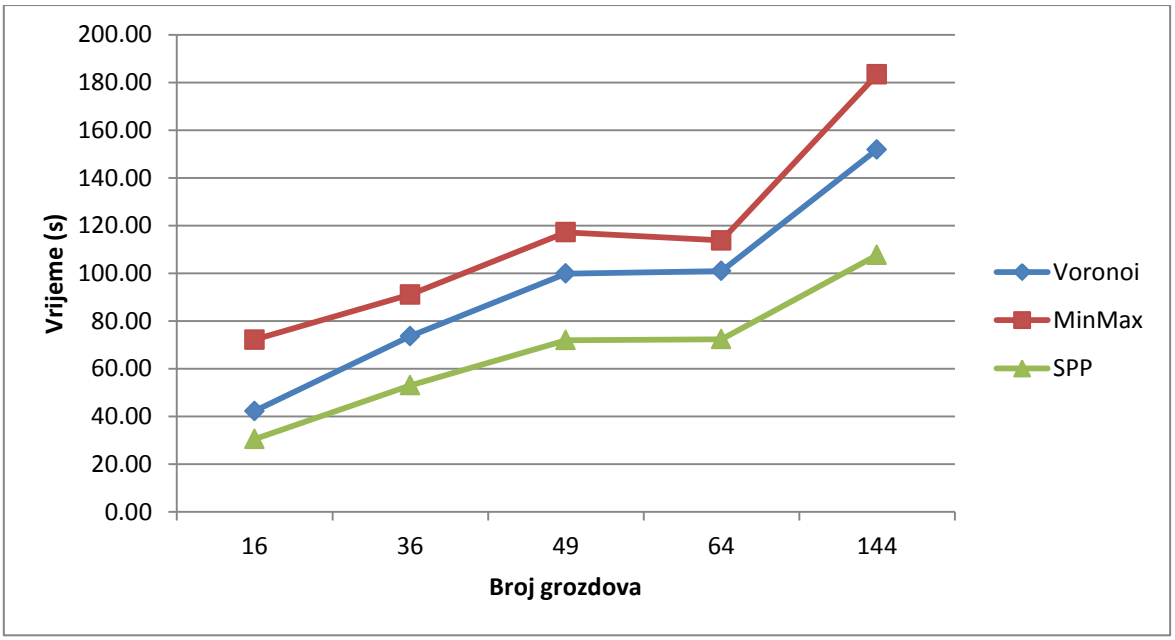
Na slici 5.3 prikazan je omjer vremena izvođenja algoritma MinMAx i algoritma SPP. Vidljivo da se s povećanjem broja objekata povećava njihov omjer, što znači da algoritam SPP brže razvrstava veće skupove objekata, što ga čini boljim i od algoritma MinMax.



Slika 5.3 Omjer vremena izvođenja algoritama MinMax i SPP za različit broj objekata

5.1.3. Pokusi u kojima se mijenja broj grozdova

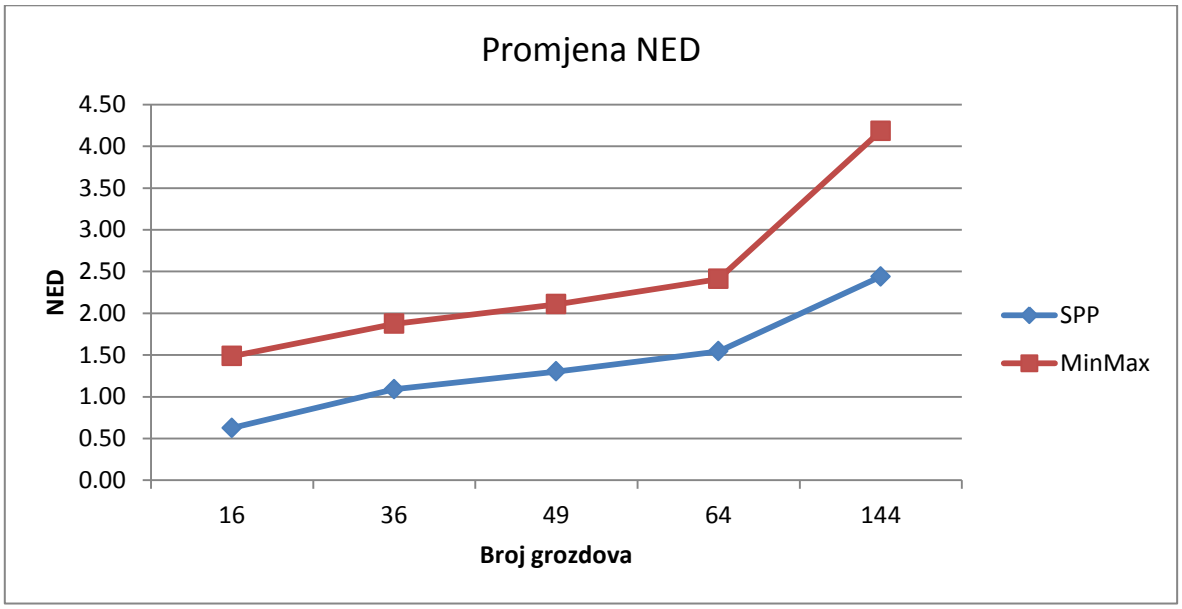
U ovim pokusima mijenja se broj grozdova dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju sa 16 i završavaju sa 144 grozda. Za svaki broj grozdova pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.4.



Slika 5.4 Vremena izvođenja triju algoritama za različit broj grozdova

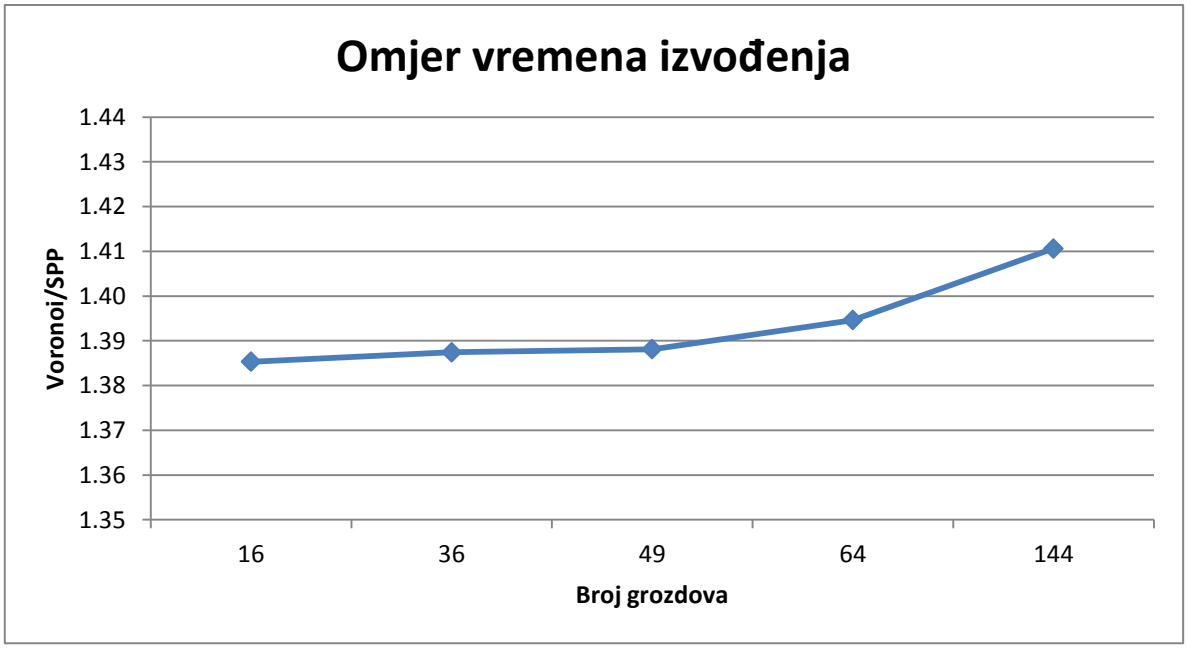
Sa slike 5.4 je vidljivo da algoritam temeljen na simetralnoj podjeli prostora ima kraća vremena izvođena od preostala dva algoritma kad se mijenja broj grozdova. Povećanjem broja grozdova povećava se i broj relacija između objekata i grozdova koje treba promatrati, pa algoritam SPP pokazuje bolja svojstva, jer ima najbrži proces odbacivanja grozdova.

Što je broj grozdova veći, oni su međusobno bliži i manja je vjerojatnost za uspješnim odbacivanjem grozdova. Posljedično, povećava se broj računanja očekivanih udaljenosti NED kao što je prikazano na slici 5.5. Iako se s brojem grozdova povećao NED, sva tri algoritma su i dalje učinkovitija od algoritma uk-means. Najveći NED = 4.17 ima algoritam MinMax kad je broj grozdova jednak 144, što je znatno manje od 49 u slučaju algoritma uk-means.



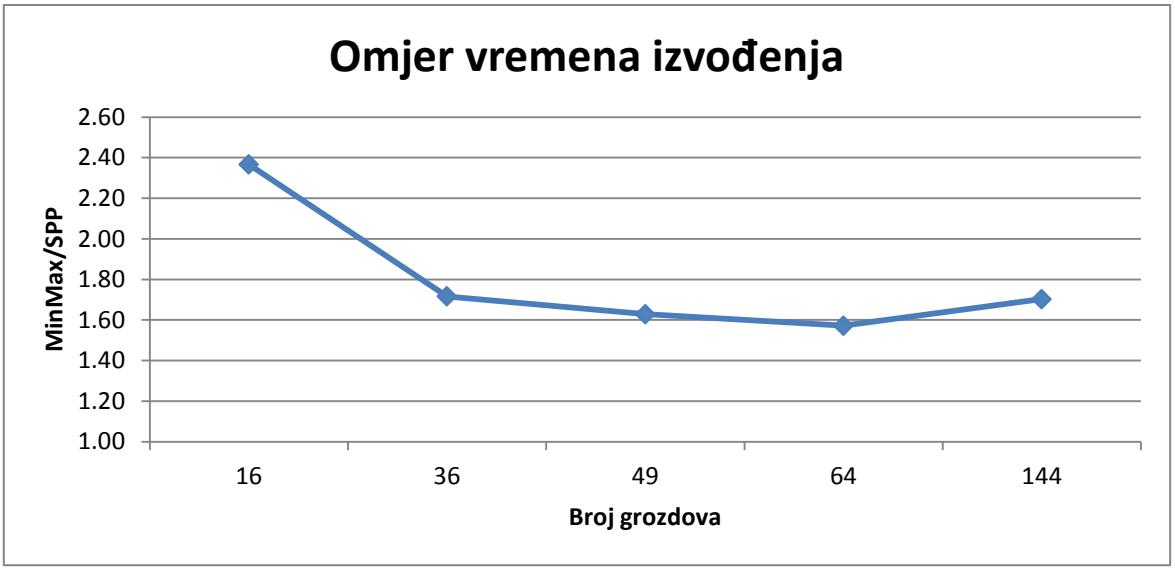
Slika 5.5 Promjena NED-a s promjenom broja grozdova

Na slici 5.6 prikazan je omjer vremena izvođenja algoritma pomoću Voronojevih dijagrama i algoritma SPP za različit broj grozdova. Sa slike je vidljivo da se s povećanjem broja grozdova povećava njihov omjer. Vrijeme izvođenja algoritma SPP kraće je u čitavom rasponu broja grozdova, što ga čini prihvatljivijim u svim primjenama u kojima je povećan broj grozdova.

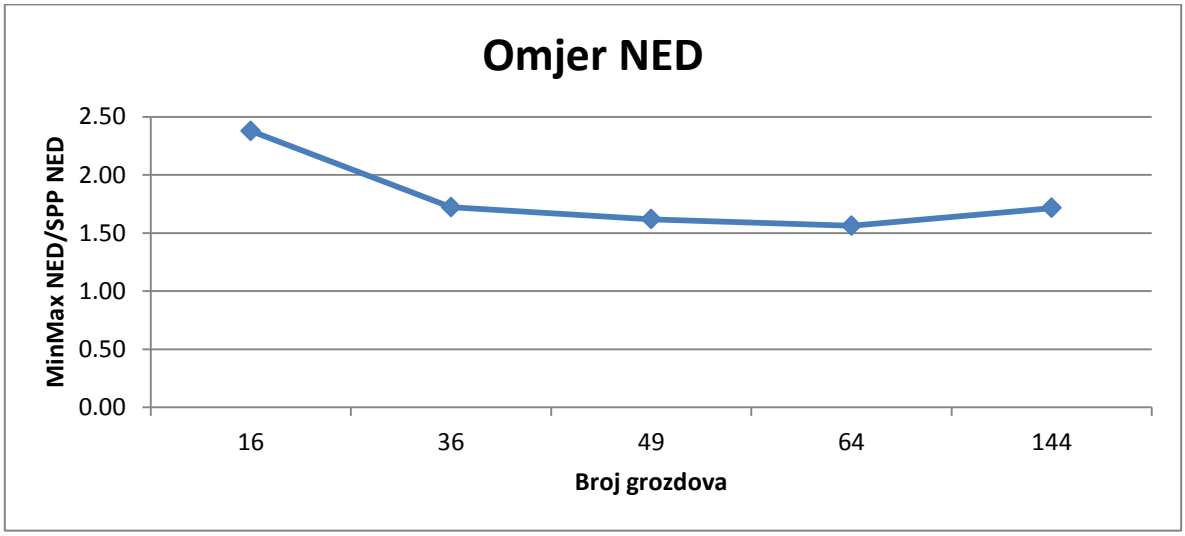


Slika 5.6 Omjer vremena izvođenja algoritma koji koristi Voronojeve dijagrame i algoritma SPP za različit broj grozdova

Na slici 5.7 prikazan je omjer vremena izvođenja algoritma MinMax i algoritma SPP za različit broj grozdova. Sa slike je vidljivo da se s povećanjem broja grozdova povećava njihov omjer, što znači da algoritam SPP i ovome slučaju daje bolje rezultate. No najveći omjer je za šesnaest grozdova, jer u tome slučaju algoritam MinMax ima loša svojstva odbacivanja i puno veći NED nego algoritam SPP kao što je prikazano na slici 5.8. Razlog tome je malen broj grozdova i loše definirane gornje i donje granice kod algoritma MinMax. Algoritam SPP ima kraća vremena izvođenja od algoritma MinMax u čitavom rasponu broja grozdova, što ga čini prihvatljivijim u situacijama kojima je velik broj grozdova.



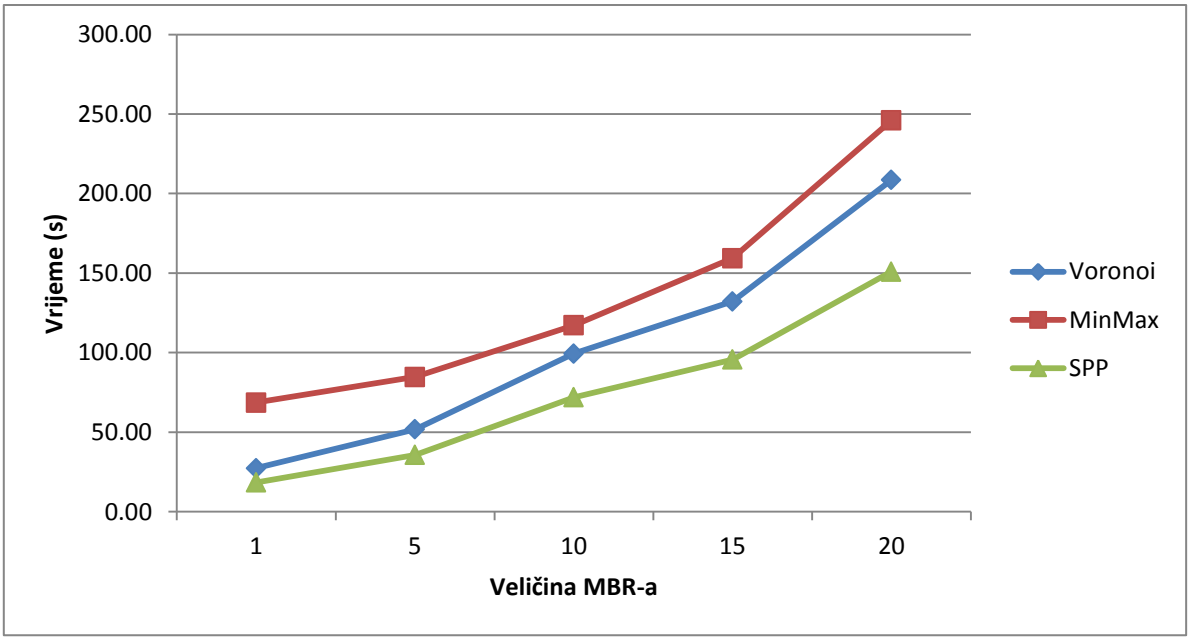
Slika 5.7 Omjer vremena izvođenja algoritama MinMax i SPP za različit broj grozdova



Slika 5.8 Omjer NED koeficijenta algoritama MinMax i SPP za različit broj grozdova

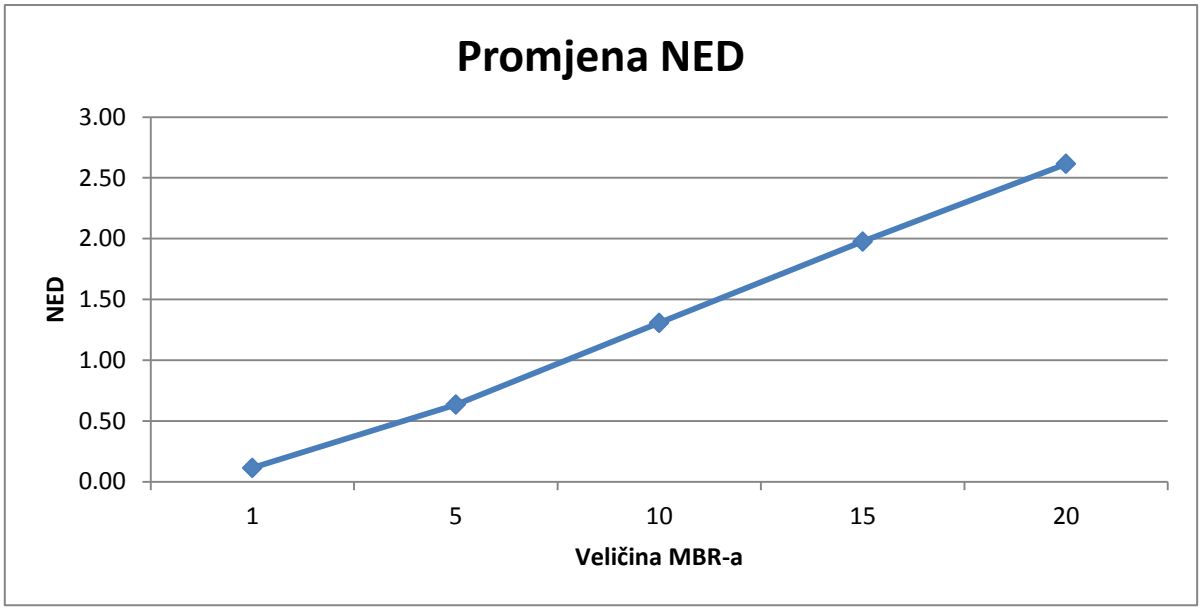
5.1.4. Pokusi u kojima se mijenja veličina MBR-a

U ovim pokusima mijenja se veličina minimalnog područja nesigurnosti MBR dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju s maksimalnom dužinom stranice jednakom 1 i završavaju s maksimalnom duljinom stranice jednakom 20. Za različite duljine stranica minimalnog područja nesigurnosti pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.9.

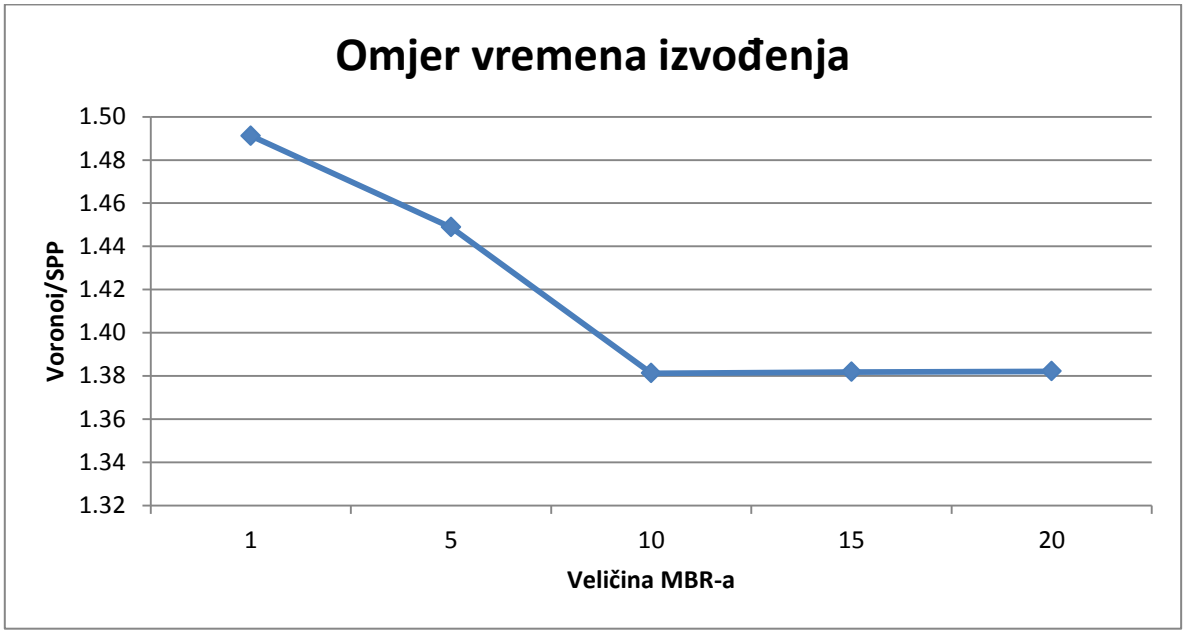


Slika 5.9 Vremena izvođenja triju algoritama za različite veličine minimalnog područja nesigurnosti

Povećanjem minimalnog područja nesigurnosti veća je vjerojatnost da će se stranice MBR-a presijecati sa simetralama, što dovodi do manjeg broja odbačenih grozdova. Isto tako povećava se maksimalna udaljenost od objekta do grozda, što dovodi do slabijeg odbacivanja grozdova i kod algoritma MinMax. Posljedica svega navedenog je veći broj računanja očekivanih udaljenosti pa NED raste kao što je pokazano na slici 5.10. Sa slike 5.9 je vidljivo da algoritam temeljen na simetralnoj podjeli prostora ima kraća vremena izvođenja od preostala dva algoritma za različite veličine MBR-a. No omjer vremena izvođenja između algoritma koji koristi Voronojeve dijagrame i algoritma SPP se neznatno mijenja kao što je prikazano na slici 5.11. Budući da je broj objekata i grozdova isti, broj relacija između njih ostaje isti kako se mijenja veličina MBR-a, pa vremenska razlika zbog bržeg postupka odbacivanja kod algoritma SPP ostaje ista.



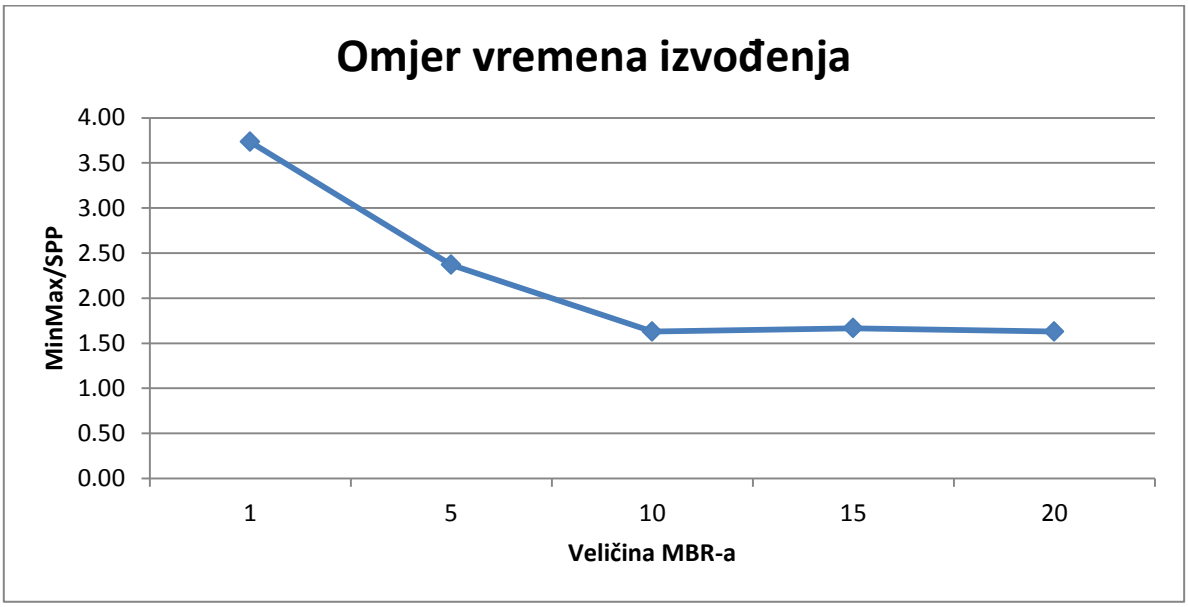
Slika 5.10 Promjena NED s veličinom minimalnog područja nesigurnosti



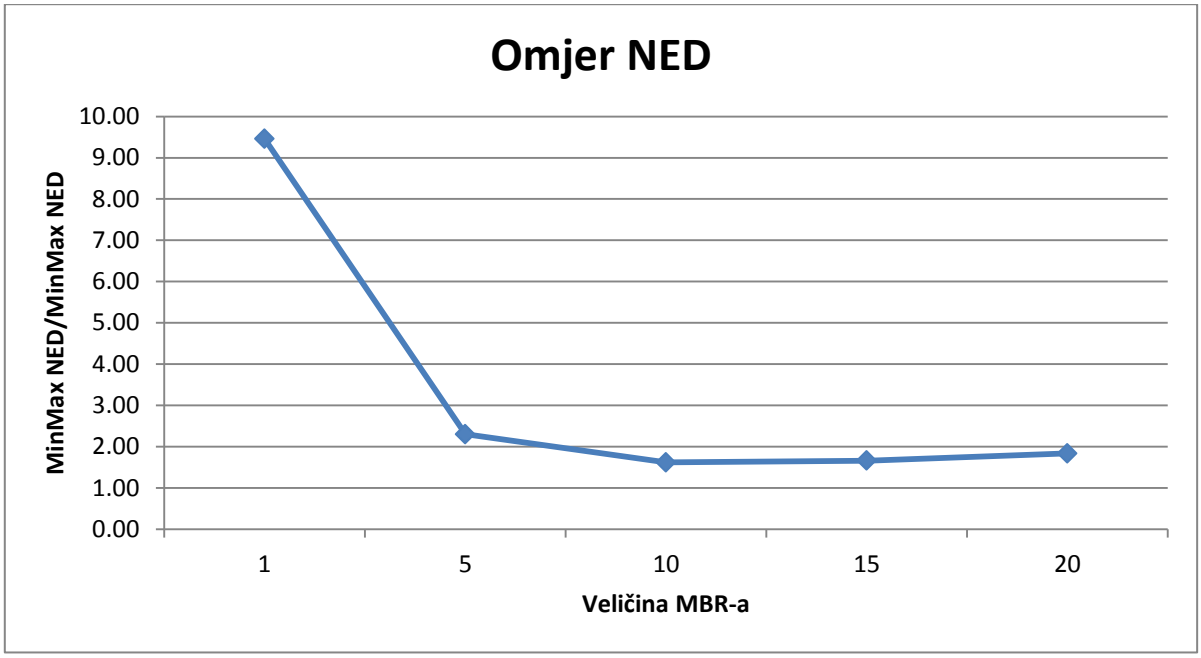
Slika 5.11 Omjer vremena izvođenja algoritma koji koristi Voronojeve dijagrama i algoritma SPP za različite veličine minimalnog područja nesigurnosti

Isti slučaj je na slici 5.12 gdje je prikazan omjer vremena izvođenja algoritma MinMax i algoritma SPP za različite veličine stranica MBR-a. I u ovome slučaju vremenska razlika zbog bržeg postupka odbacivanja kod SPP algoritma ostaje ista, jer je broj objekata i grozdova uvijek isti. Iznimka su minimalne veličine MBR-a gdje algoritam MinMax ima lošija svojstva

odbacivanja grozdova i NED je puno veći nego kod algoritma SPP kao što je prikazano na slici 5.13. Zbog toga je omjer vremena izvođenja veći za manje veličine MBR-a. Može se zaključiti kako algoritam SPP ima najkraća vremena izvođenja za sve veličine minimalnog područja nesigurnosti.



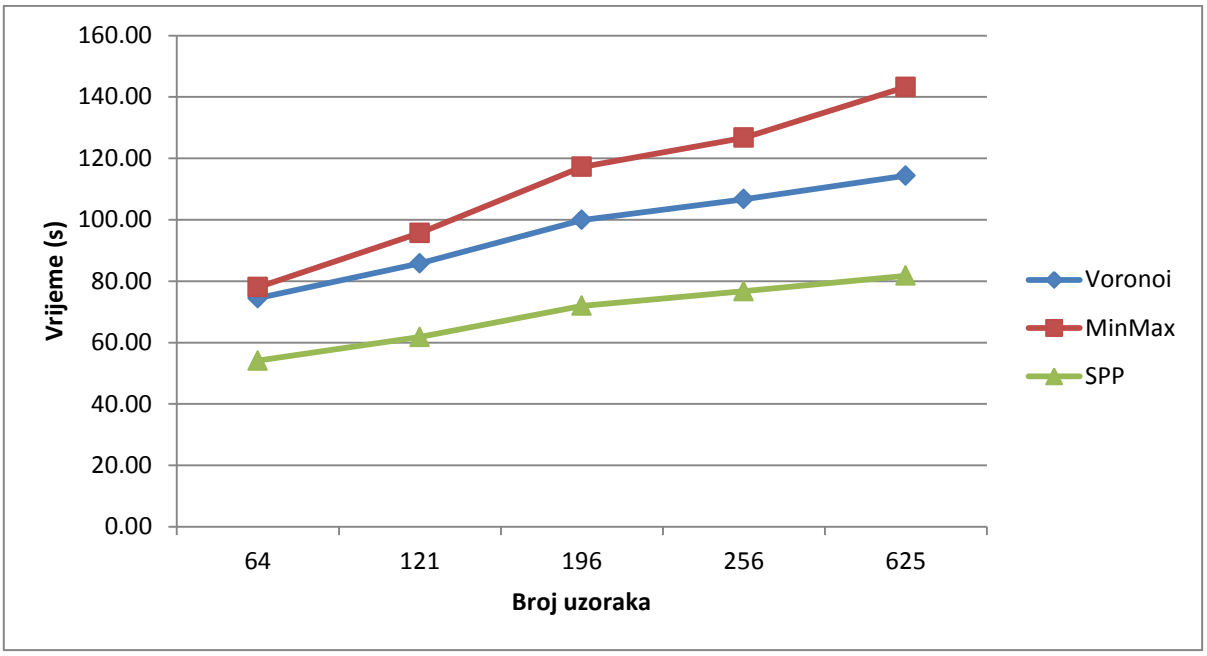
Slika 5.12 Omjer vremena izvođenja algoritama MinMax i SPP za različite veličine minimalnog područja nesigurnosti



Slika 5.13 Omjer NED koeficijenta algoritama MinMax i SPP za različite veličine minimalnog područja nesigurnosti

5.1.5. Pokusi u kojima se mijenja broj uzoraka PDF-a

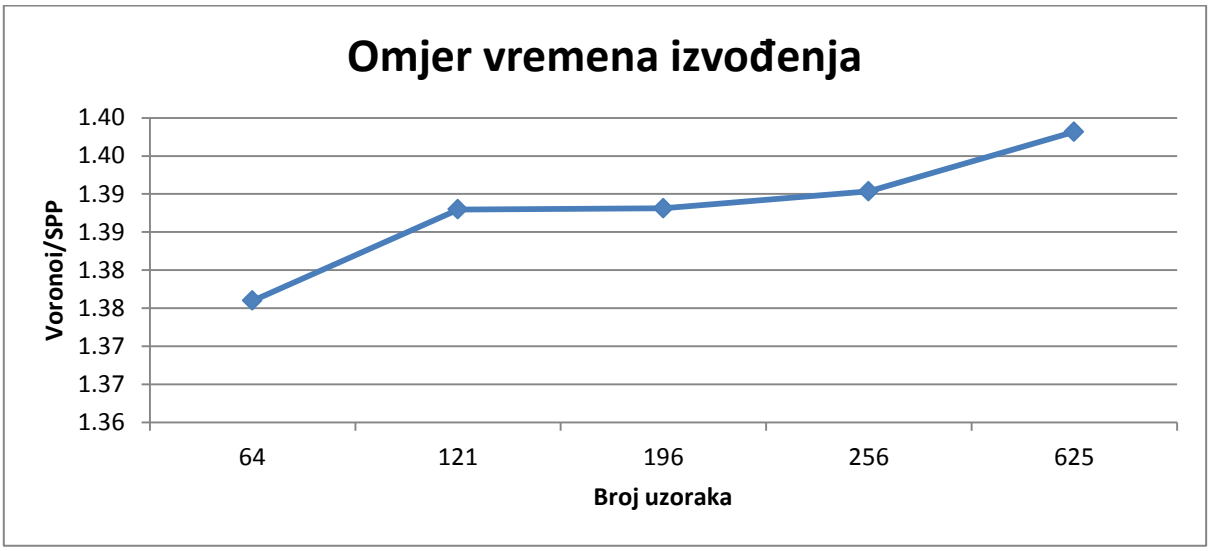
U ovim pokusima mijenja se broj uzoraka kojim je predstavljena funkcija gustoće vjerojatnosti PDF, dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju sa 64 uzorka za pojedini PDF i završavaju s maksimalna 625 uzorka. Za različite brojeve uzoraka PDF-a pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.14.



Slika 5.14 Vremena izvođenja triju algoritama za različit broj uzoraka

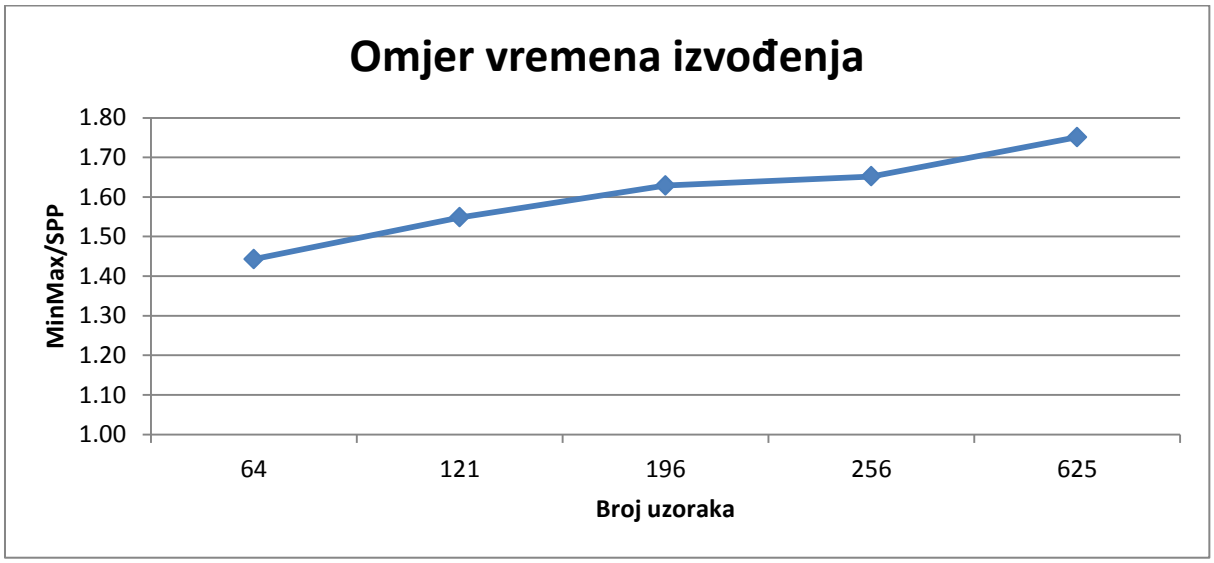
Povećanje broja uzoraka ne utječe na broj uspješno odbačenih grozdova, ali utječe na vrijeme računanja očekivane udaljenosti ED. Budući da se mora izračunati udaljenost između svakog uzorka i središta grozda, jasno je da će s povećanjem broja uzoraka računanje očekivane udaljenosti trajati dulje. Algoritam pomoću Voronojevih dijagrama i algoritam SPP imaju isti NED tako da uvijek računaju isti broj očekivanih udaljenosti pa je utjecaj broja uzoraka na obje metode isti. Kao što je ranije rečeno algoritam SPP ima brži proces odbacivanja grozdova pa i u ovom slučaju ima kraća vremena izvođenja. Algoritam MinMax ima veći NED, što znači da odbacuje manji broj grozdova, pa je broj računanja očekivanih udaljenosti kod ove metode veći. Posljedično tome, broj uzoraka ima veći utjecaj na vrijeme izvođenja algoritma MinMax, jer povećava vrijeme računanja očekivanih udaljenosti. Zbog toga se s povećanjem broja uzoraka vrijeme izvođenja MinMax algoritma značajnije povećava nego kod druga dva algoritma.

Omjer vremena izvođenja algoritma pomoćuVoronojevih dijagrama i algoritma SPP neznatno se povećava s brojem uzoraka PDF-a kao što je prikazano na slici 5.15. Budući da je i u ovom slučaju broj objekata i grozdova isti, broj relacija se ne mijenja s brojem uzoraka, pa vremenska razlika zbog bržeg postupka odbacivanja kod simetralnog algoritma ostaje ista.



Slika 5.15 Omjer vremena izvođenja algoritma koji koristi Voronojeve dijagrame i SPP algoritma za različit broj uzoraka

Omjer vremena izvođenja algoritama MinMax i SPP za različit broj uzoraka prikazan je na slici 5.16. Omjer se povećava, jer je računanje ED kod algoritma MinMax zahtjevnije s povećanjem broja uzoraka. Sa slike 5.16 je vidljivo kako omjer raste od 1.44 do 1.75.



Slika 5.16 Omjer vremena izvođenja algoritama MinMax i SPP za različit broj uzoraka

5.2. Pokusi sa algoritmom SDSA

U ovom poglavlju provedeni su pokusi koji uspoređuju vremena razvrstavanja algoritama MinMax i SPP u slučaju kad se oni kombiniraju s algoritmom za podjelu područja skupa objekata. Uspoređena su vremena izvođenja algoritma SPP s korištenjem podjele i bez korištenja podjele ukupnog područja skupa objekata. Algoritam nastao kombinacijom algoritma SDSA i algoritma SPP naziva se SDSA-SPP. Na isti način uspoređena su vremena izvođenja algoritma MinMax. Algoritam koji je nastao kombinacijom algoritma MinMax i algoritma SDSA nazvan je SDSA-MinMax. Nije bilo potrebe za provjerom kombinacije algoritma SDSA i algoritma koji koristi Voronojeve dijagrame, jer podjela na njega ima isti učinak kao i na preostala dva algoritma što je pokazano u [2]. Najprije su provedeni pokusi nad osnovnim skupom parametara i uspoređeni su rezultati dobiveni pojedinim algoritmom. Nakon toga provedeni su pokusi u kojima se mijenja jedan od parametara, dok ostali parametri zadržavaju osnovne vrijednosti kao i u poglavlju 5.1.

5.2.1. Pokusi s osnovnim skupom parametara

Pokusi su provedeni s parametrima prikazanim u tablici 5.1. Ponovljeni su dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja i rezultati su prikazani u tablici 5.3. U tablici su prikazani naziv algoritma, vrijeme izvođenja u sekundama i NED koji predstavlja broj računanja očekivane udaljenosti po objektu po iteraciji.

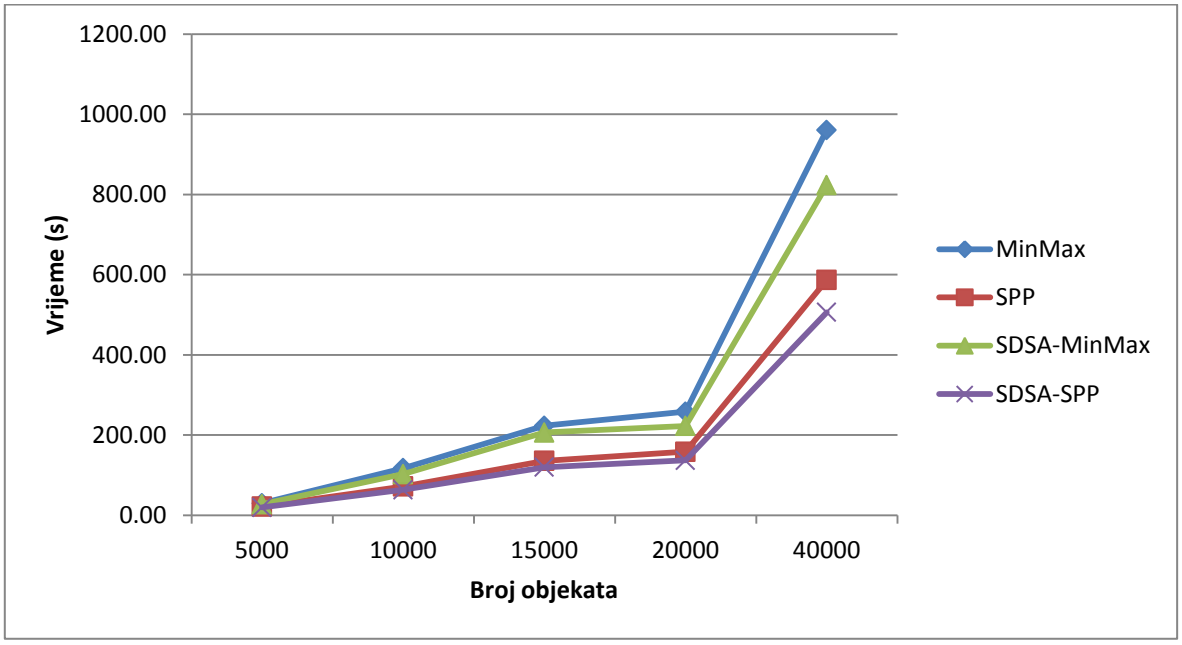
Tablica 5.3 Vremena izvođenja osnovnih i kombiniranih algoritama korištenjem osnovnog skupa parametara prikazanog u tablici 5.1

Naziv	**Vrijeme izvođenja (s)**	**NED**
Algoritam MinMax	117.206	2.113
Algoritam SPP	71.965	1.307
Algoritam SDSA-MinMax	102.441	2.113
Algoritam SDSA-SPP	63.678	1.307

Osnovni i kombinirani algoritmi imaju isti princip odbacivanja grozdova pa im je NED jednak, no iz gornje tablice je vidljivo da kombinirani algoritmi koji koriste podjelu područja skupa objekata imaju kraća vremena izvođenja, jer se podjelom smanjuje broj relacija između objekata i grozdova koje se moraju promatrati. Također je vidljivo da algoritam SDSA-SPP ima kraće vrijeme izvođenja nego algoritam SDSA-MinMax, što dodatno potvrđuje bolja svojstva algoritma SPP razvijenog u ovoj disertaciji.

5.2.2. Pokusi u kojima se mijenja broj objekata

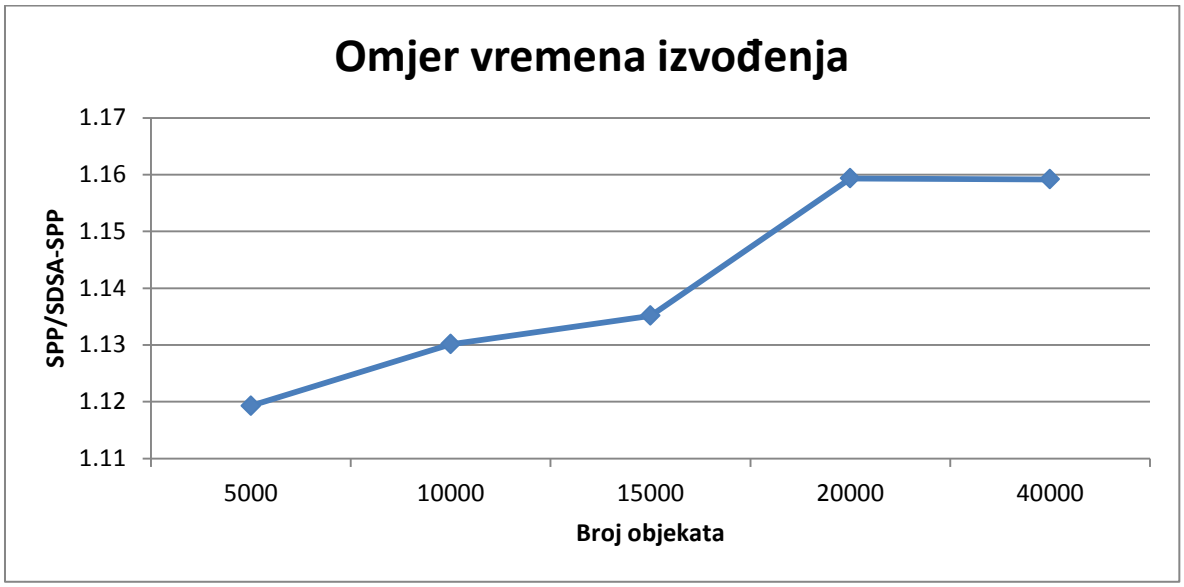
U ovim pokusima mijenja se broj objekata dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi kreću sa 5000 objekata i završavaju sa 40000 objekata. Za svaki broj objekata pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.17.



Slika 5.17 Vremena izvođenja osnovnih i kombiniranih algoritama za različit broj objekata

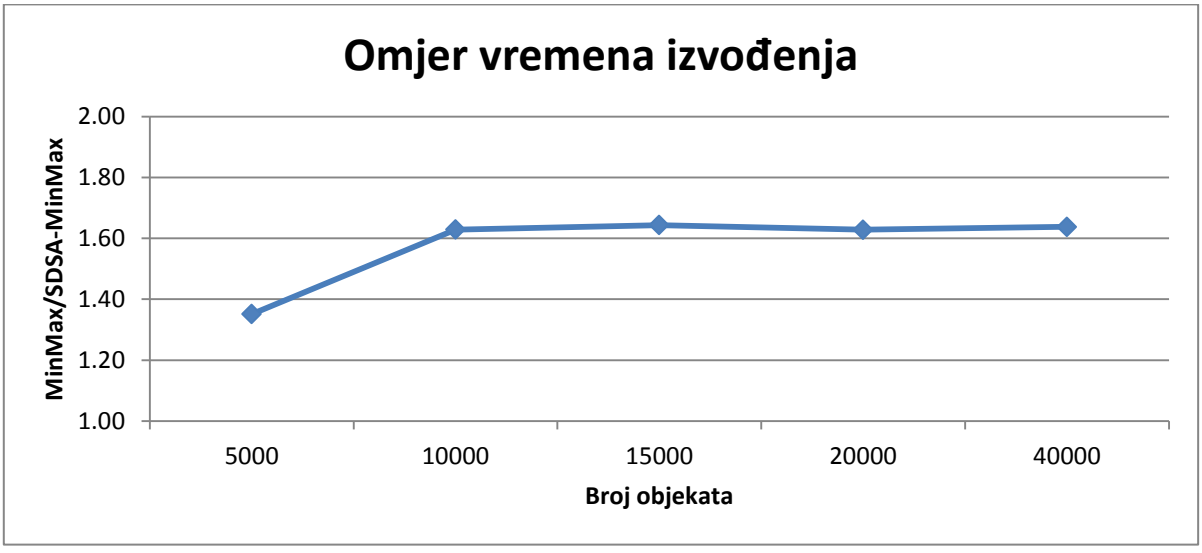
Sa slike 5.17 je vidljivo da kombinacija s algoritmom SDSA daje bolje rezultate u oba slučaja, nego kad se koriste osnovni algoritmi. Povećanjem broja objekata povećava se i broj relacija između objekata i grozdova koje treba promatrati. Algoritam SDSA tad dolazi do izražaja, jer se njegovim korištenjem smanjuje broj relacija između objekata i grozdova koje se moraju promatrati. Na slici 5.18 prikazan je omjer vremena izvođenja algoritama SPP i

SDSA-SPP. Sa slike je vidljivo da se s povećanjem broja objekata povećava njihov omjer, što znači da kombinacija daje bolje rezultate za veći broj objekata. Kombinacija s algoritmom SDSA daje bolja vremena izvođenja u čitavom rasponu broja objekata, što kombinaciju čini boljom od osnovnog algoritma.



Slika 5.18 Omjer vremena izvođenja algoritama SPP i SDSA-SPP za različit broj objekata

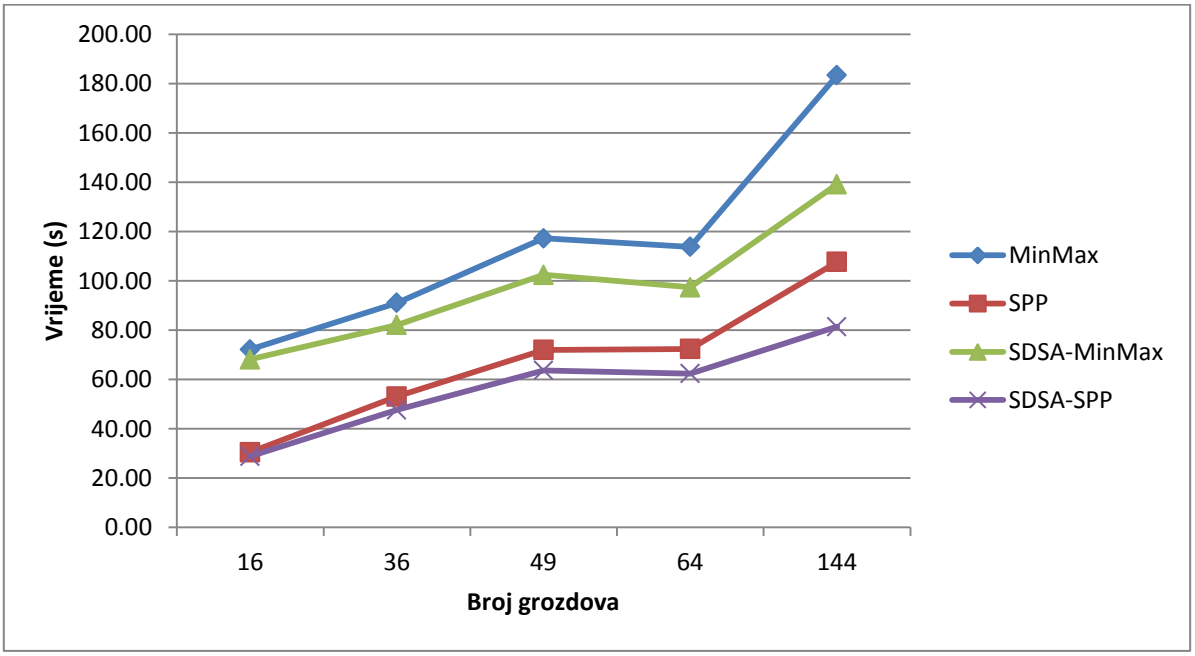
Na slici 5.19 prikazan je omjer vremena izvođenja algoritama MinMax i SDSA-MinMax. I na ovoj slici je vidljivo da se s povećanjem broja objekata povećava njihov omjer, što znači da kombinacija s algoritmom SDSA ima bolje rezultate, nego osnovni algoritam.



Slika 5.19 Omjer vremena izvođenja algoritama MinMax i SDSA-MinMax za različit broj objekata

5.2.3. Pokusi u kojima se mijenja broj grozdova

U ovim pokusima mijenja se broj grozdova dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju sa 16 i završavaju sa 144 grozda. Za svaki broj grozdova pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.20.

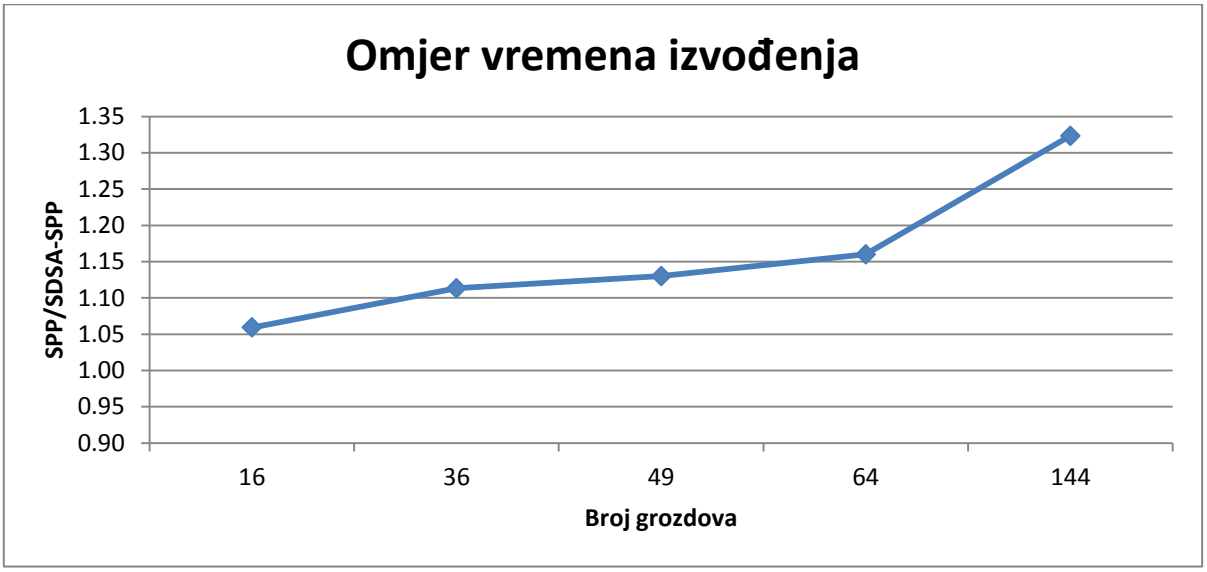


Slika 5.20 Vremena izvođenja osnovnih i kombiniranih algoritama za različit broj grozdova

Sa slike 5.20 je vidljivo da kombinacija algoritma SDSA s osnovnim algoritmima ima kraća vremena izvođenja nego osnovni algoritmi. Povećanjem broja grozdova povećava se i broj relacija između objekata i grozdova koje treba promatrati. Time algoritam SDSA dolazi do izražaja, jer na samome početku odbacuje većinu grozdova za pojedini objekt i smanjuje broj promatranih relacija među njima. Povećanjem broja grozdova povećava se njihov broj u pojedinom području pa je lakše zadovoljiti uvjete za ispravnu podjelu područja skupa objekata. Zbog većeg broja grozdova uvjeti podjele ranije su osigurani pa se ne mora povećavati broj pravokutnih područja koja se moraju promatrati, što može biti slučaj u primjenama sa manjim brojem grozdova.

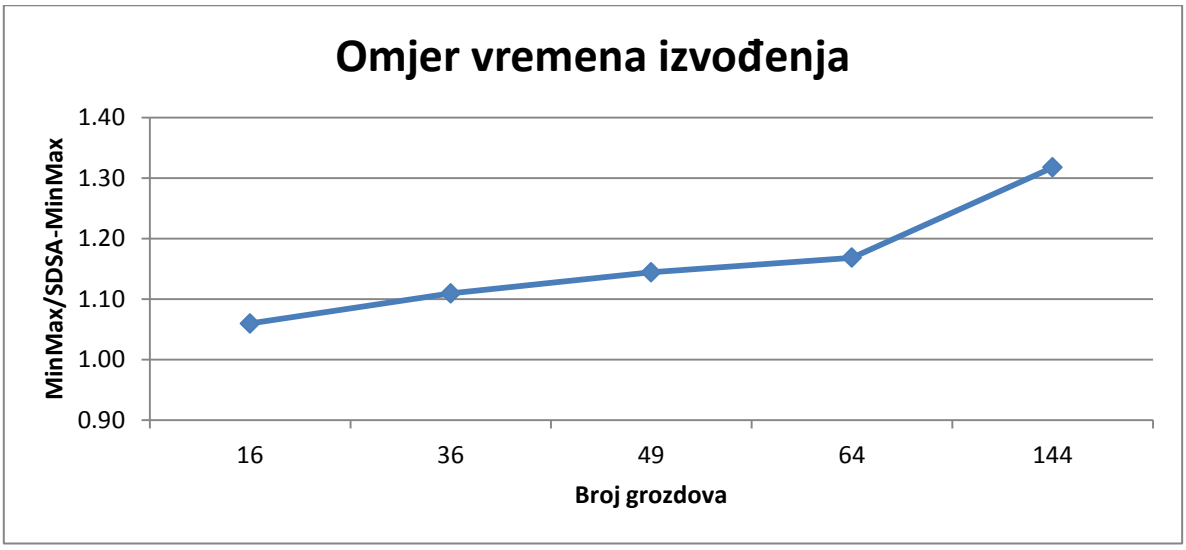
Na slici 5.21 prikazan je omjer vremena izvođenja algoritama SPP i SDSA-SPP za različit broj grozdova. Sa slike je vidljivo da se s povećanjem broja grozdova povećava njihov omjer, što znači da podjela na samome početku odbaci veći broj grozdova pa algoritam mora raditi

manji broj provjera. Kombinacija s algoritmom SDSA ima bolja vremena izvođenja u čitavom rasponu broja grozdova, što kombinaciju čini boljom od osnovnog algoritma u svim primjenama.



Slika 5.21 Omjer vremena izvođenja algoritama SPP i SDSA-SPP za različit broj grozdova

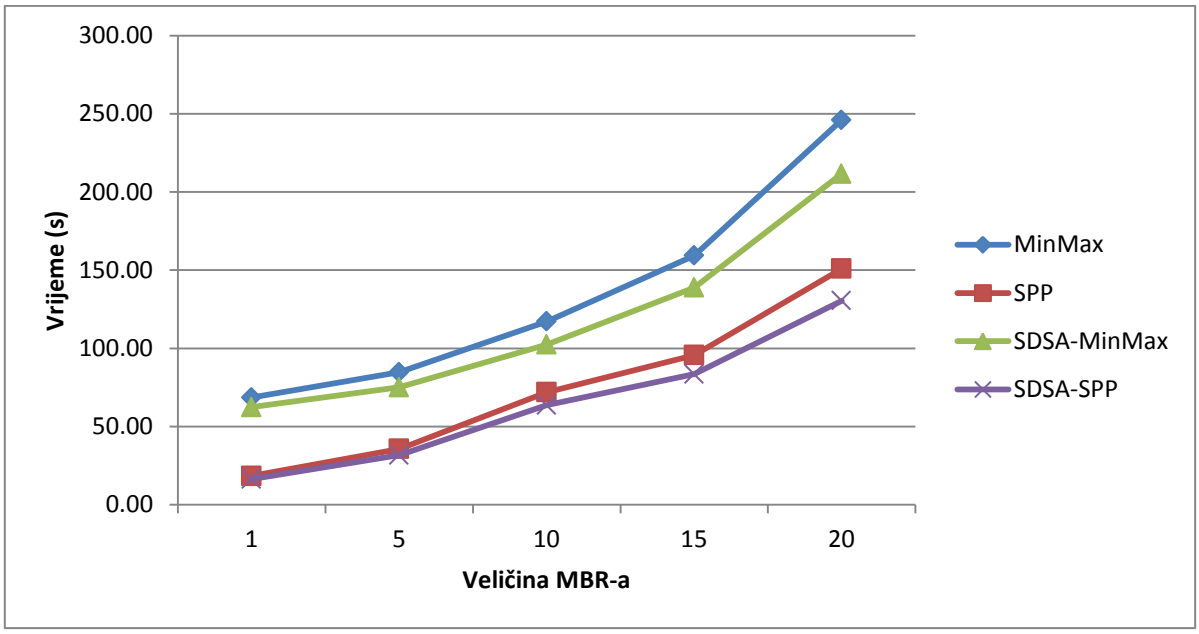
Na slici 5.22 prikazan je omjer vremena izvođenja algoritama MinMax i SDSA-MinMax za različit broj grozdova. Vidljivo je da se s povećanjem broja grozdova povećava njihov omjer. Vremena izvođenja bolja su u čitavom rasponu broja grozdova, što kombinaciju čini boljom u svim primjenama u kojima se mijenja broj grozdova.



Slika 5.22 Omjer vremena izvođenja algoritama MinMax i SDSA-MinMax za različit broj grozdova

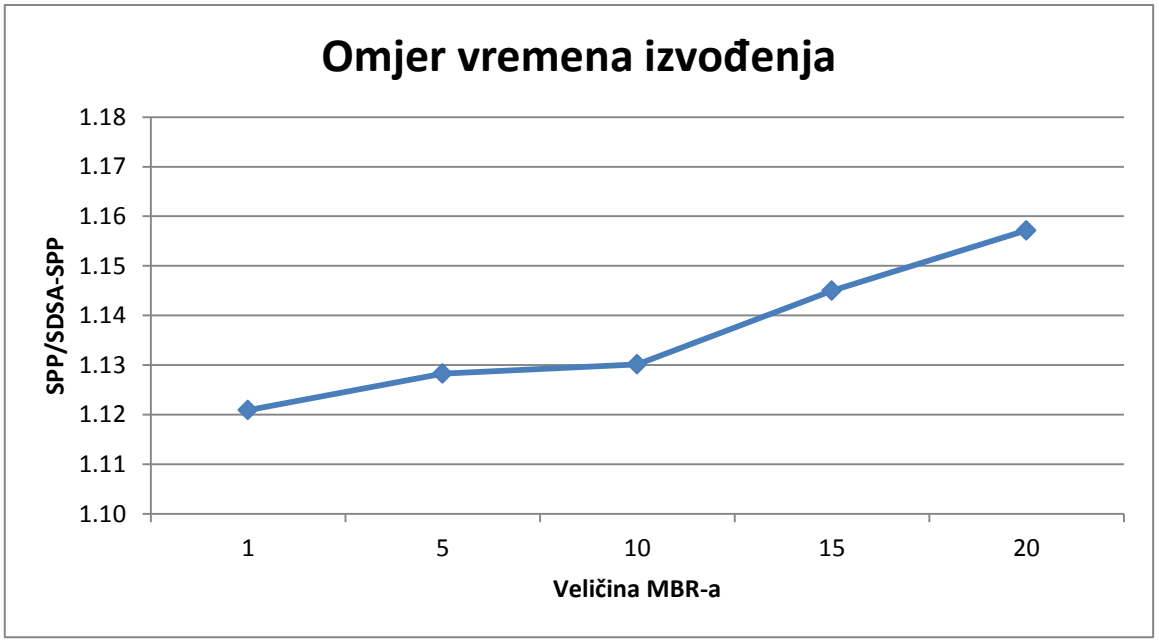
5.2.4. Pokusi u kojima se mijenja veličina MBR-a

U ovim pokusima mijenja se veličina minimalnog područja nesigurnosti MBR dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju s maksimalnom duljinom stranice jednakom 1 i završavaju s maksimalnom duljinom stranice jednakom 20. Za različite duljine stranica minimalnog područja nesigurnosti pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.23.

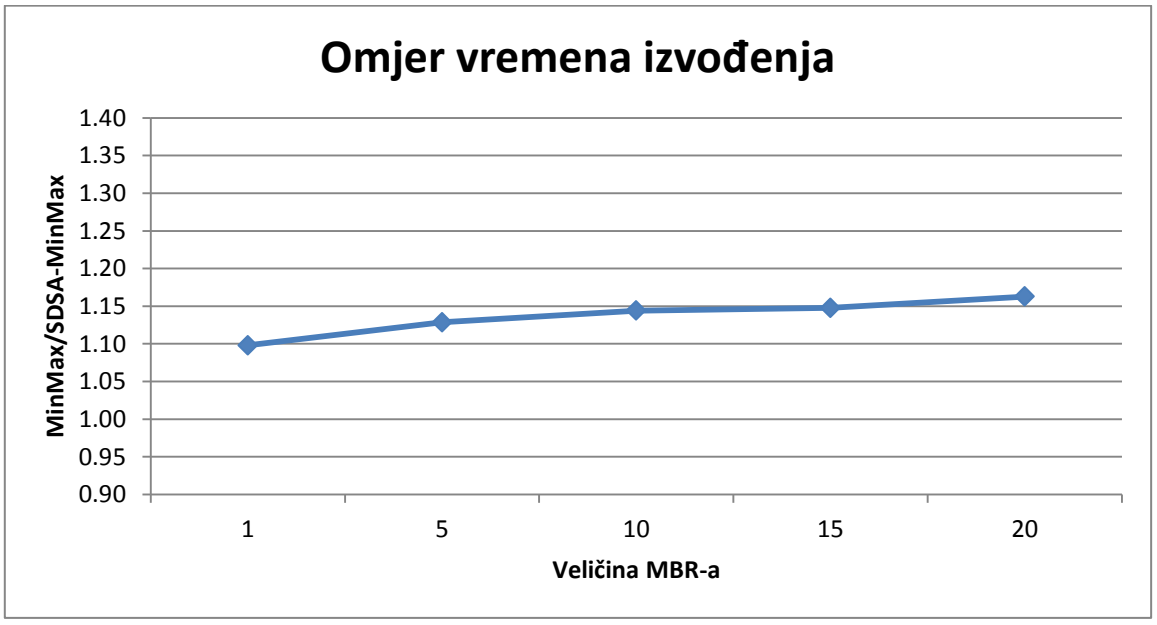


Slika 5.23 Vremena izvođenja osnovnih i kombiniranih algoritama za različite veličine minimalnog područja nesigurnosti

Povećanjem minimalnog područja nesigurnosti veća je vjerojatnost da će se stranice MBR-a presijecati sa simetralama, što dovodi do manjeg broja odbačenih grozdova. Sa slike 5.23 je vidljivo da oba algoritma u kombinaciji s algoritmom SDSA imaju kraća vremena izvođenja, za različite veličine MBR-a, nego kao osnovni algoritmi. No omjer vremena izvođenja između osnovnih i kombiniranih algoritama se neznatno mijenja kao što je prikazano na slikama 5.24 i 5.25. Budući da je broj objekata i grozdova isti, broj relacija između njih ostaje isti kako se mijenja veličina MBR-a, pa vremenska razlika zbog bržeg postupka odbacivanja korištenjem algoritma SDSA ostaje ista.



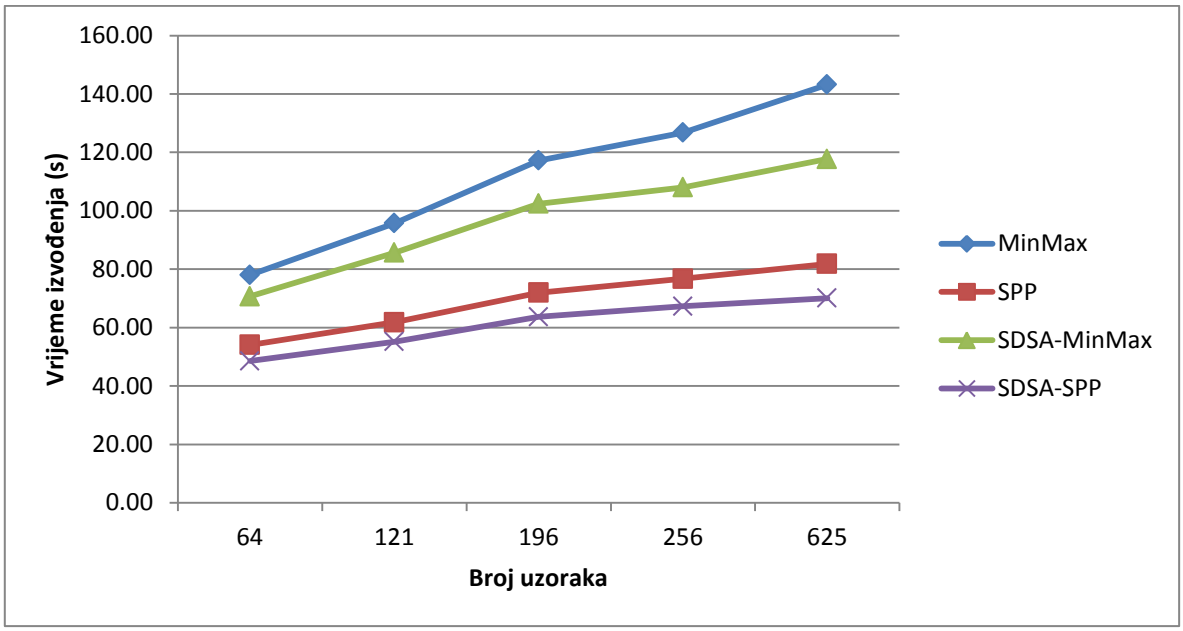
Slika 5.24 Omjer vremena izvođenja algoritama SPP i SDSA-SPP za različite veličine minimalnog područja nesigurnosti



Slika 5.25 Omjer vremena izvođenja algoritama MinMax i SDSA-MinMax za različite veličine minimalnog područja nesigurnosti

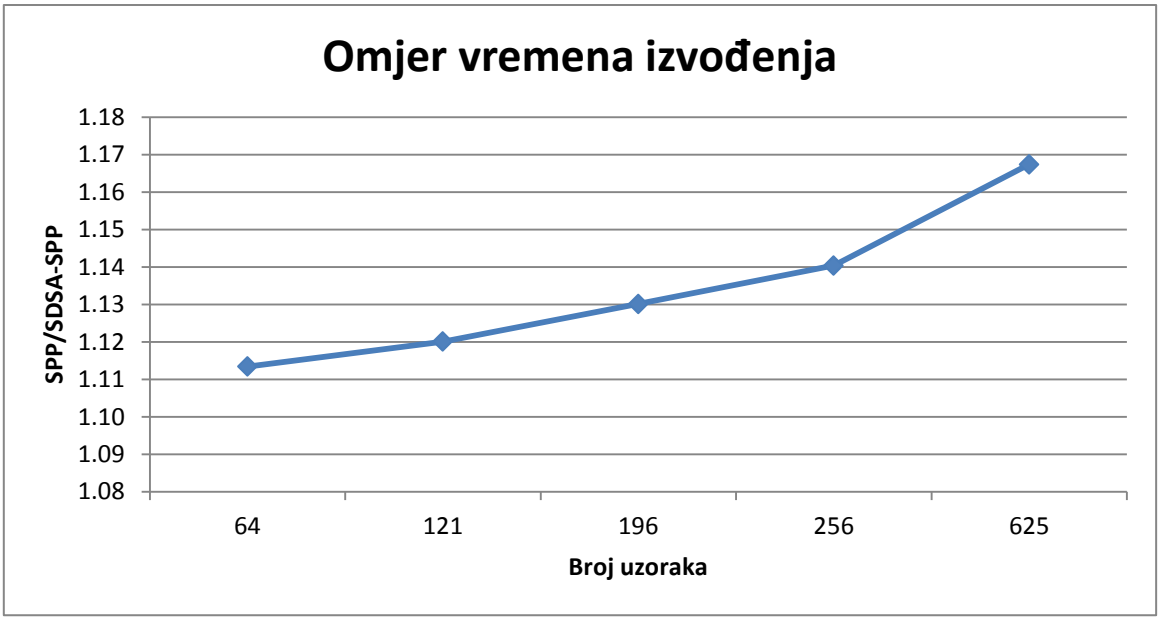
5.2.5. Pokusi u kojima se mijenja broj uzoraka PDF-a

U ovim pokusima mijenja se broj uzoraka kojim je predstavljena funkcija gustoće vjerojatnosti PDF, dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju sa 64 uzorka za pojedini PDF i završavaju s maksimalna 625 uzorka. Za različite brojeve uzoraka PDF-a pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.26.

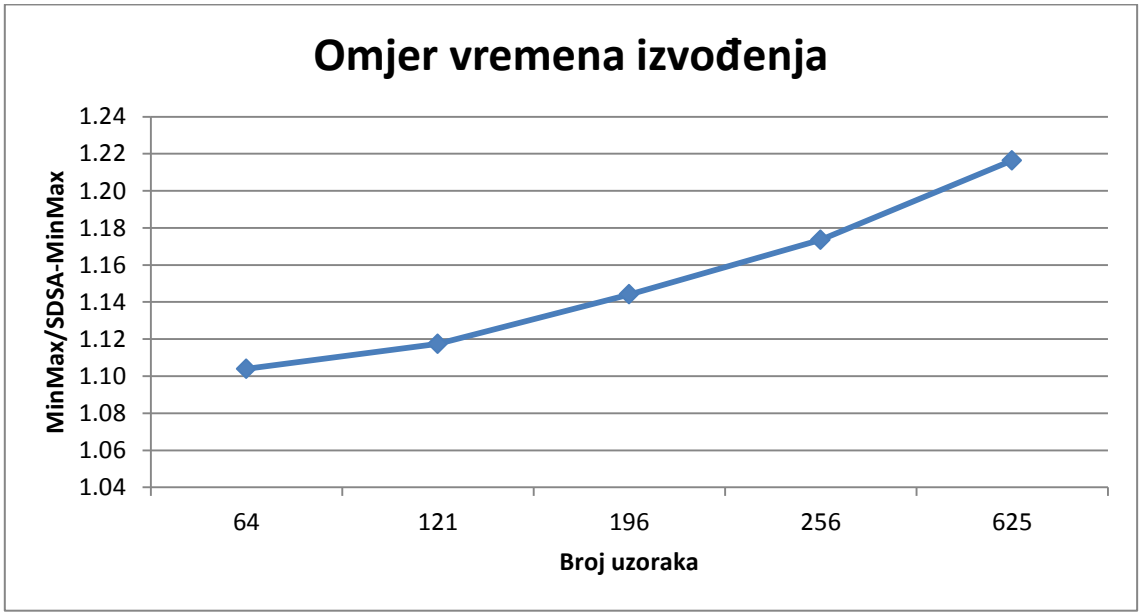


Slika 5.26 Vremena izvođenja osnovnih i kombiniranih algoritama za različit broj uzoraka

Povećanje broja uzoraka ne utječe na broj uspješno odbačenih grozdova, ali utječe na vrijeme računanja očekivane udaljenosti ED. Budući da se mora izračunati udaljenost između svakog uzorka i središta grozda, jasno je da će s povećanjem broja uzoraka računanje očekivane udaljenosti trajati duže. Osnovni i kombinirani algoritmi imaju isti NED tako da uvijek računaju isti broj očekivanih udaljenosti pa je utjecaj broja uzoraka na oba algoritma isti. No korištenjem podjele na samome početku se odbace grozdovi izvan promatranih područja pa i u ovom slučaju kombinacija s algoritmom SDSA ima nešto kraća vremena izvođenja. Omjer vremena izvođenja algoritama SPP i SDSA-SPP te algoritama MinMax i SDSA-MinMax neznatno se povećava s povećanjem broja uzoraka PDF-a kao što je prikazano na slikama 5.27 i 5.28. To pokazuje da su svojstva kombinacije s algoritmom SDSA podjednaka za sve vrijednosti broja uzoraka.



Slika 5.27 Omjer vremena izvođenja algoritama SPP i SDSA-SPP za različit broj uzoraka



Slika 5.28 Omjer vremena izvođenja algoritama MinMax i SDSA-MinMax za različit broj uzoraka

5.3. Pokusi s paralelnim procesima razvrstavanja

U ovom poglavlju provedeni su pokusi koji uspoređuju vremena razvrstavanja algoritma SDSA-SPP kao serijskog procesa, te istog algoritma kada se izvodi kao paralelni proces na dvije i četiri jezgre. Algoritam koje se izvodi kao paralelni proces razvrstavanja na dvije jezgre nazvan je SDSA-SPP2, a algoritam koji se izvodi kao paralelni proces razvrstavanja na četiri jezgre nazvan je SDSA-SPP4. Najprije su provedeni pokusi nad osnovnim skupom parametara prikazanim u tablici 5.1. Uspoređeni su rezultati dobiveni serijskim i paralelnim procesima razvrstavanja. Nakon toga provedeni su pokusi u kojima se mijenja jedan od parametara, dok ostali parametri zadržavaju osnovne vrijednosti.

5.3.1. Pokusi s osnovnim skupom parametara

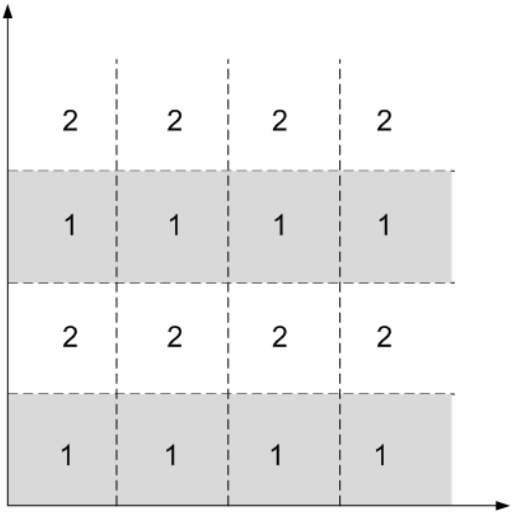
Pokusi su provedeni s parametrima prikazanim u tablici 5.1. Ponovljeni su dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja i rezultati su prikazani u tablici 5.4. U tablici su prikazani naziv algoritma i vrijeme izvođenja u sekundama.

Tablica 5.4 Vremena izvođenja serijskih i paralelnih algoritama korištenjem parametara prikazanih u tablici 5.1

Naziv	**Vrijeme izvođenja (s)**
SDSA-SPP	63.678
SDSA-SPP2	41.096
SDSA-SPP4	29.767

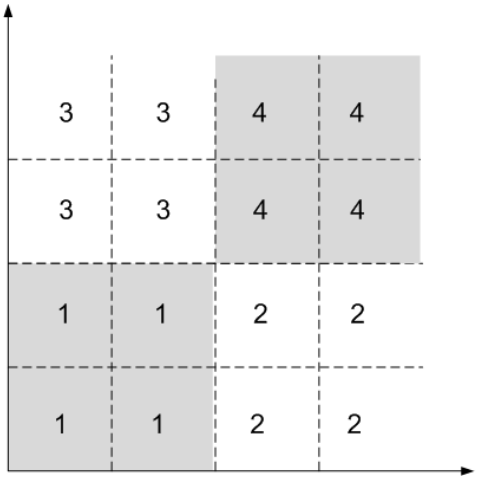
Iz tablice 5.4 je vidljivo da se paralelnim izvođenjem dobiju kraća vremena nego serijskim izvođenjem. Dobiveni rezultati jasni su sami po sebi, jer je za očekivati da se razvrstavanje izvodi brže ako na njemu istovremeno rade dva ili četiri procesora. Isto tako razvrstavanje je brže ako na njemu rade četiri umjesto dva procesa, što je pokazano pokusima. Serijski proces se istovremeno može izvoditi na samo jednoj jezgri računala, dok se paralelni procesi istovremeno mogu izvoditi na dvije ili četiri jezgre i time brže razvrstati objekte. Na

slici 5.29 prikazan je način procesiranja pojedinog pravokutnog područja na dva paralelna procesa.



Slika 5.29 Prikaz procesiranja pojedinog pravokutnog područja korištenjem dvije jezgre

Pravokutna područja označena sivom bojom i brojem 1 procesiraju se s prvom jezgrom, a područja označena bijelom bojom i brojem 2 procesiraju se s drugom jezgrom. Korištenim rasporedom svaka jezgra ima isti broj vanjskih i unutarnjih područja kako bi jezgre bile ravnomjerno opterećene i obavile razvrstavanje u približno istom vremenu. Unutarnja pravokutna područja imaju više susjednih područja, pa njihovo razvrstavanje traje dulje nego kod vanjskih područja koja imaju manje susjednih područja s grozdovima. Stoga je bitno ispravno rasporediti pravokutna područja po jezgrama računala. Na slici 5.30 prikazan je način kako se razvrstavaju pojedina pravokutna područja na četiri jezgre računala.

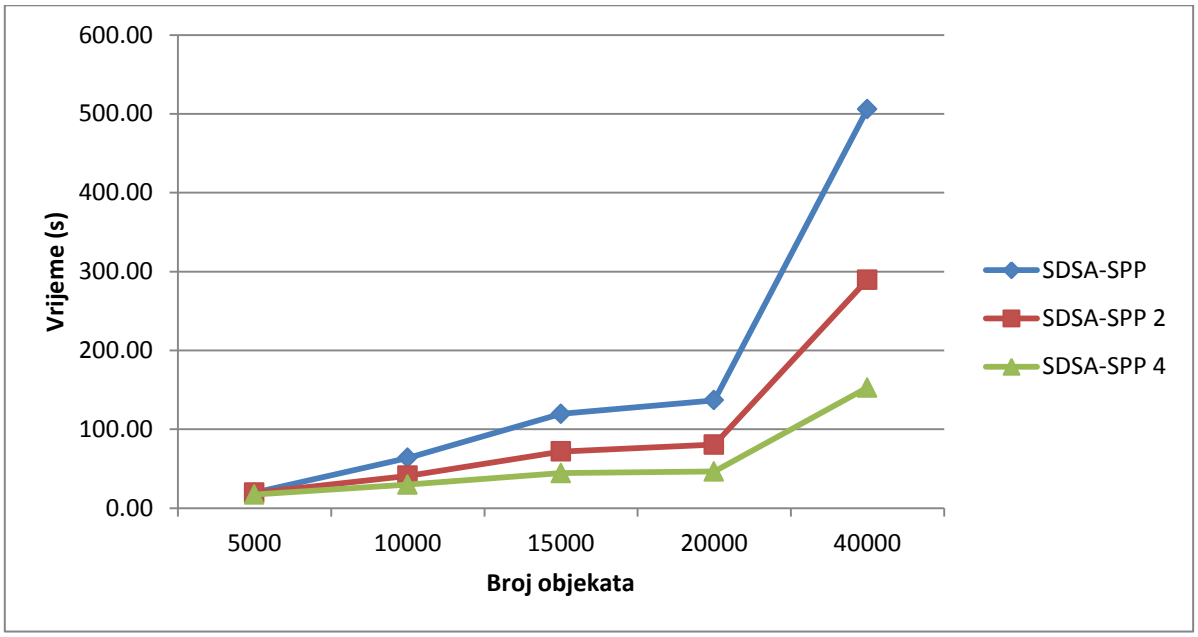


Slika 5.30 Prikaz procesiranja pojedinog pravokutnog područja korištenjem četiri jezgre

Pravokutna područja označena sivom bojom procesiraju se s prvom i četvrtom jezgrom, a pravokutna područja označena bijelom bojom procesiraju se s drugom i trećom jezgrom. I u ovom slučaju svaka jezgra ima isti broj vanjskih i unutarnjih pravokutnih područja, te su sve jezgre ravnomjerno opterećene i obavljaju razvrstavanje u približno istom vremenu. Navedeni način procesiranja pravokutnih područja korišten je u svim idućim pokusima.

5.3.2. Pokusi u kojima se mijenja broj objekata

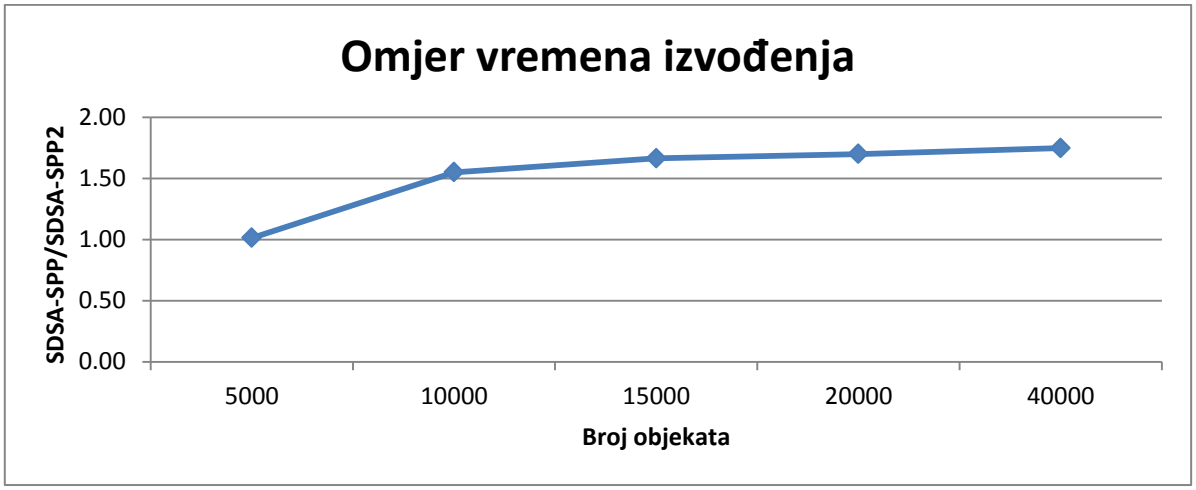
U ovim pokusima mijenja se broj objekata dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju sa 5000 objekata i završavaju sa 40000 objekata. Za svaki broj objekata pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.31.



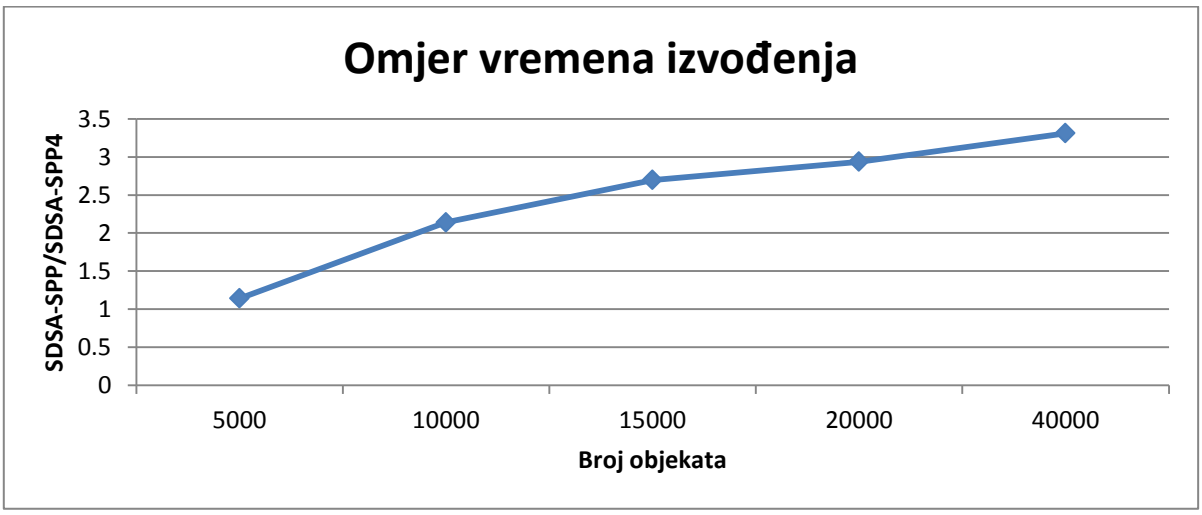
Slika 5.31 Vremena izvođenja serijskih i paralelnih algoritama korištenjem osnovnih parametara prikazanih u tablici 5.1

Sa slike 5.31 je vidljivo da za mali broj objekata, paralelno razvrstavanje ima neznatno bolje rezultate od serijskog razvrstavanja. Razlog tome je što komunikacija između paralelnih procesa i glavnog programa ima značajan udio u ukupnom vremenu izvođenja. Za veći broj objekata komunikacija između procesa i glavnog programa ima neznatan udio u ukupnom vremenu izvođenja, i paralelno razvrstavanje ima znatno kraća vremena izvođenja. Vidimo da

paralelni proces na dvije jezgre ima kraće vrijeme izvođenja nego serijski proces. Posljedično, paralelni proces na četiri jezgre ima kraće vrijeme izvođenja nego paralelni proces na dvije jezgre. Povećanjem broja objekata komunikacija između procesa ima sve manji utjecaj i time paralelni procesi dolaze do izražaja. Na slikama 5.32 i 5.33 prikazani su omjeri vremena izvođenja algoritama SDSA-SPP i SDSA-SPP2, te algoritama SDSA-SPP i SDSA-SPP4 za različit broj objekata. Sa slika je vidljivo da se s povećanjem broja objekata povećava njihov omjer, što znači da paralelno razvrstavanje daje bolje rezultate za veći broj objekata, jer se smanjuje utjecaj komunikacije između procesa i glavnog programa na ukupno vrijeme izvođenja. Paralelno razvrstavanje daje bolja vremena izvođenja u čitavom rasponu broja objekata, što ga čini boljim od serijskog razvrstavanja, naročito za veliki broj objekata.



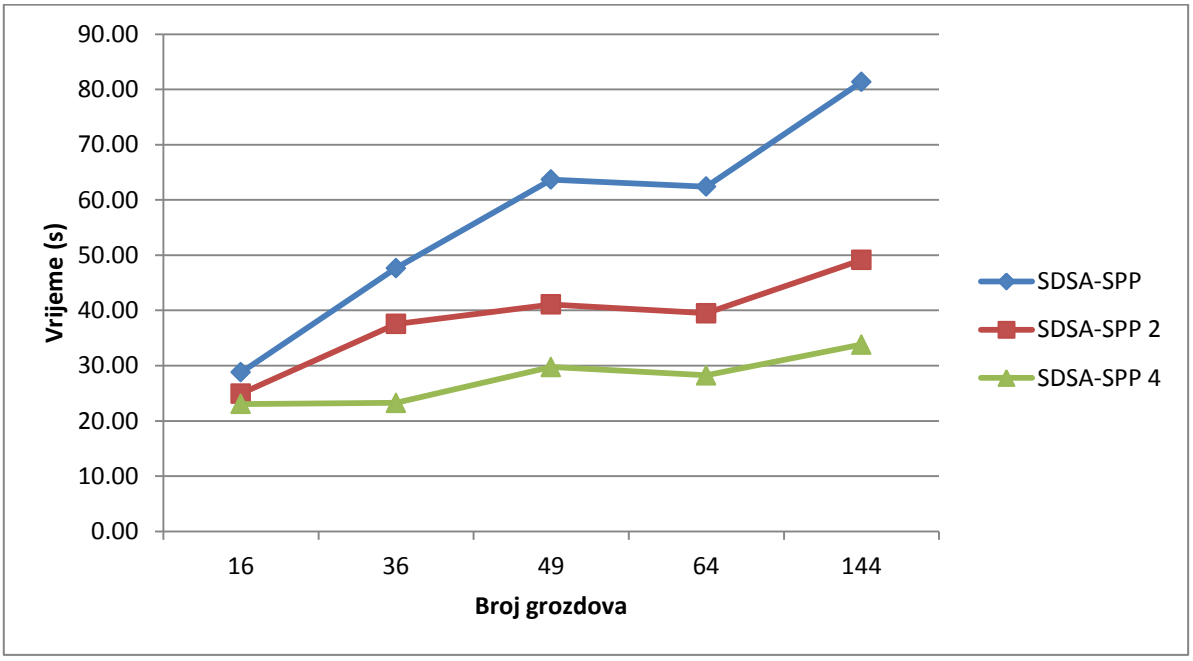
Slika 5.32 Omjer vremena izvođenja algoritama SDSA-SPP i SDSA-P2 za različit broj objekata



Slika 5.33 Omjer vremena izvođenja algoritama SDSA-SPP i SDSA-SPP4 za različit broj objekata.

5.3.3. Pokusi u kojima se mijenja broj grozdova

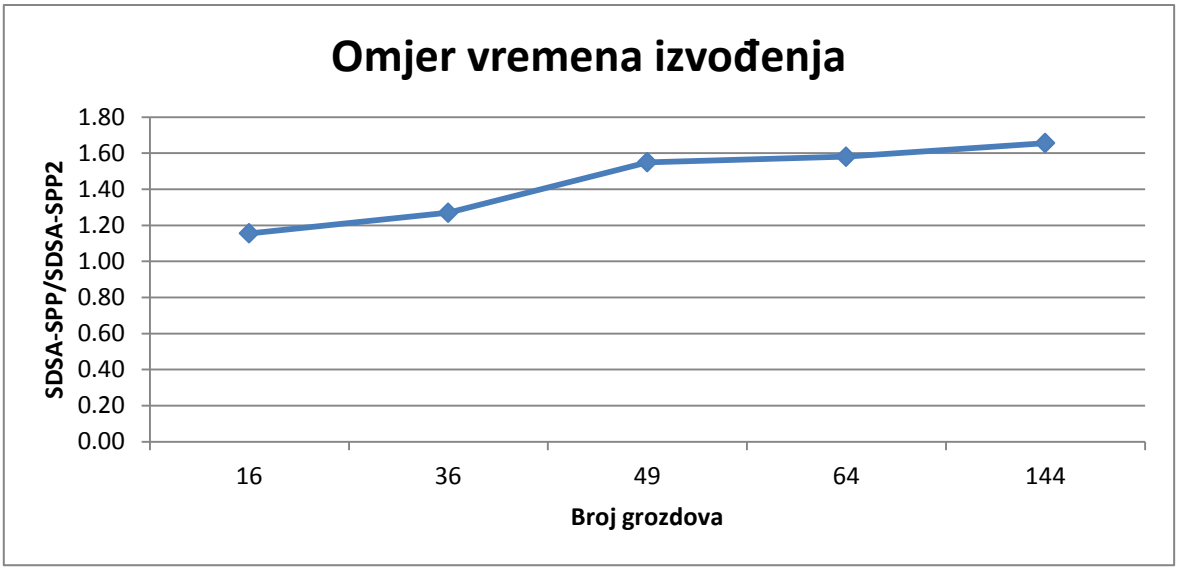
U ovim pokusima mijenja se broj grozdova dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju s 16 i završavaju s 144 grozda. Za svaki broj grozdova pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.34.



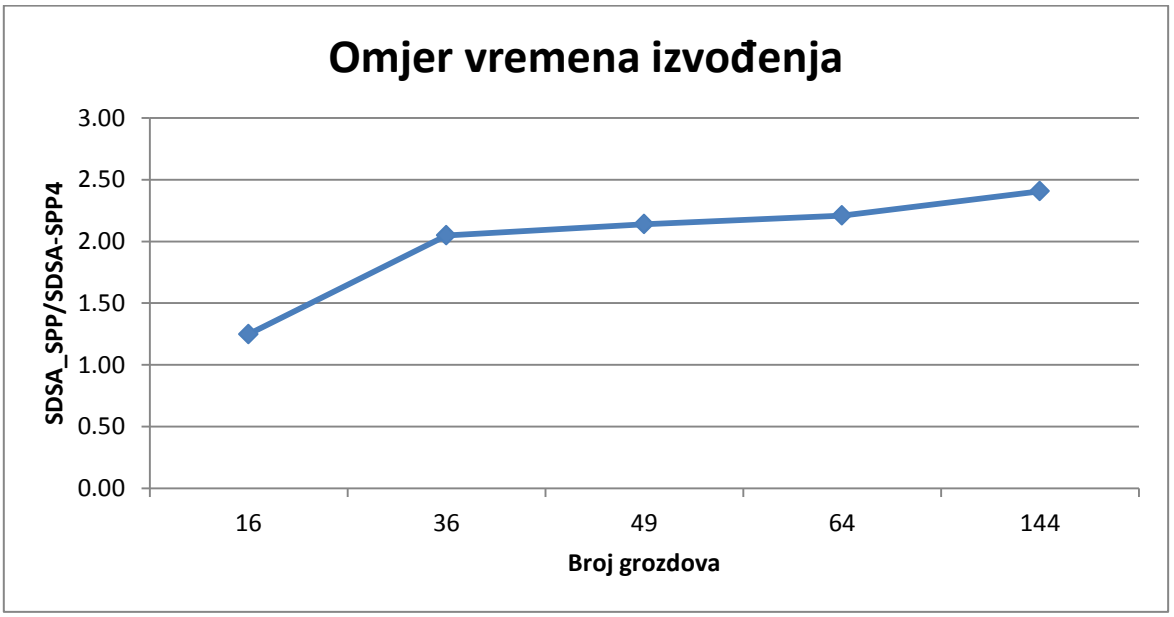
Slika 5.34 Vremena izvođenja serijskih i paralelnih algoritama za različit broj grozdova

Sa slike 5.34 je vidljivo da paralelni algoritmi i u ovome slučaju imaju kraća vremena izvođenja. Povećanjem broja grozdova povećava se i broj relacija između objekata i grozdova koje treba promatrati pa se povećava vrijeme izvođenja. Također se povećava broj očekivanih udaljenosti koje se moraju izračunati. No u paralelnim algoritmima to se vrijeme za računanje ED dijeli na dva ili četiri dijela, ovisno o broju jezgri, pa je samo računanje očekivane udaljenosti brže.

Na slikama 5.35 i 5.36 prikazani su omjeri vremena izvođenja algoritama SDSA-SPP i SDSA-SPP2, te algoritama SDSA-SPP i SDSA-SPP4 za različit broj grozdova. Sa slike je vidljivo da se s povećanjem broja grozdova povećava omjer serijskih i paralelnih algoritama. To je zbog toga što se sve veće vrijeme izvođenja, koje je posljedica većeg broja grozdova, u paralelnim procesima dijeli na dva ili četiri dijela. Time se paralelnim procesiranjem dobiva znatno skraćenje vremena izvođenja kako se povećava broj grozdova.



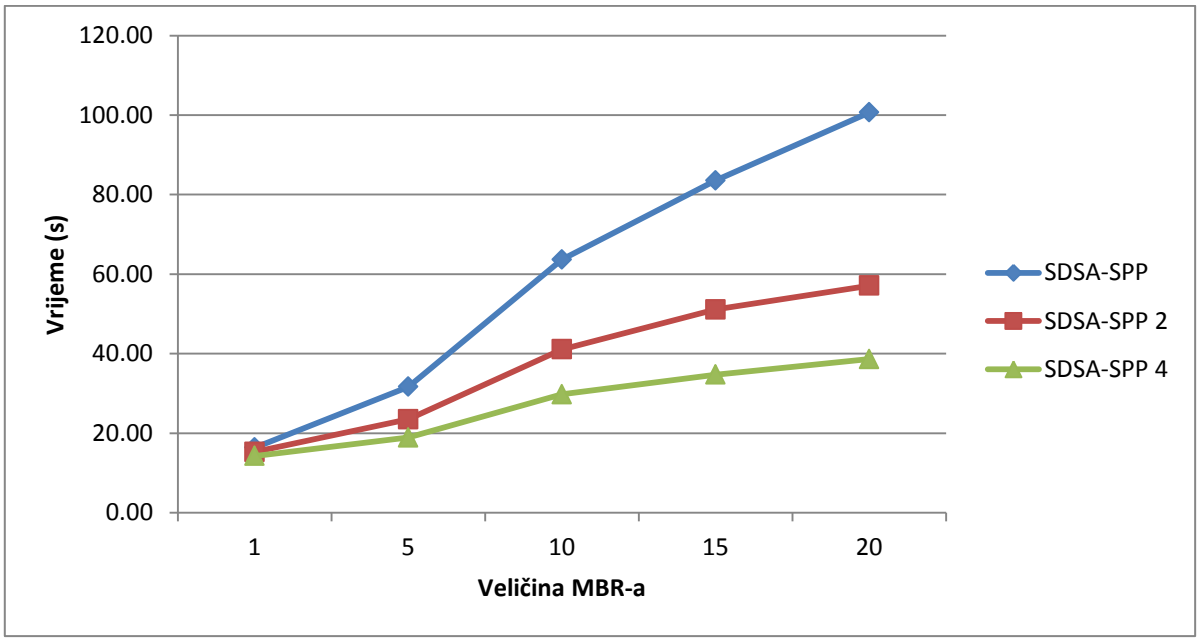
Slika 5.35 Omjer vremena izvođenja algoritama SDSA-SPP i SDSA-SPP2 za različit broj grozdova



Slika 5.36 Omjer vremena izvođenja algoritama SDSA-SPP i SDSA-SPP4 za različit broj grozdova

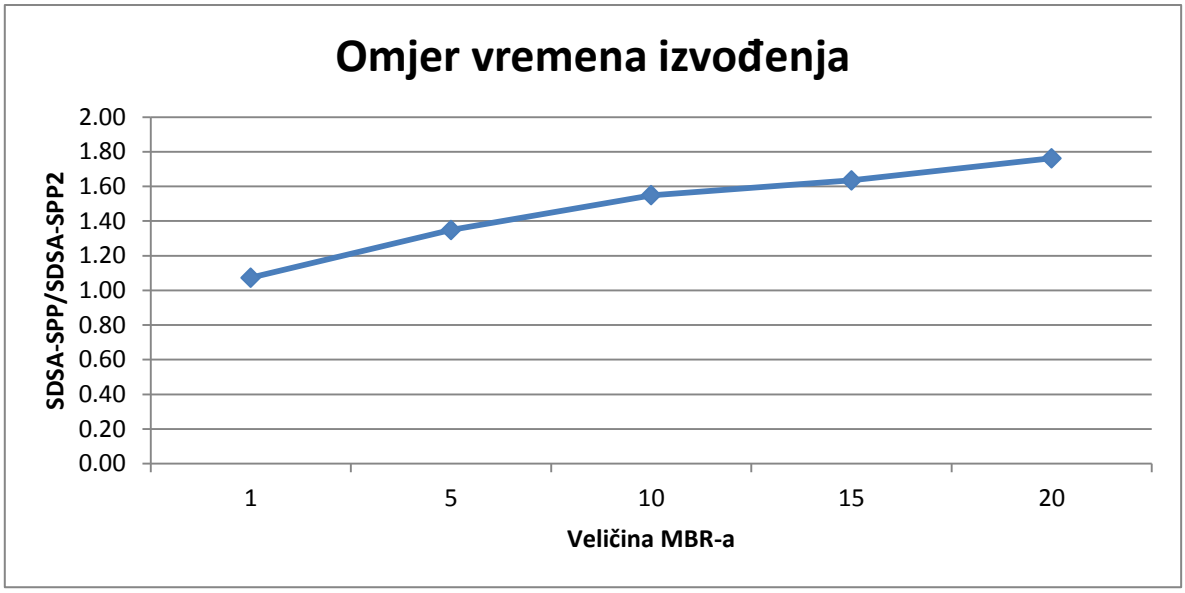
5.3.4. Pokusi u kojima se mijenja veličina MBR-a

U ovim pokusima mijenja se veličina minimalnog područja nesigurnosti MBR dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi počinju s maksimalnom duljinom stranice jednakom 1 i završavaju s maksimalnom duljinom stranice jednakom 20. Za različite duljine stranica minimalnog područja nesigurnosti pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.37.

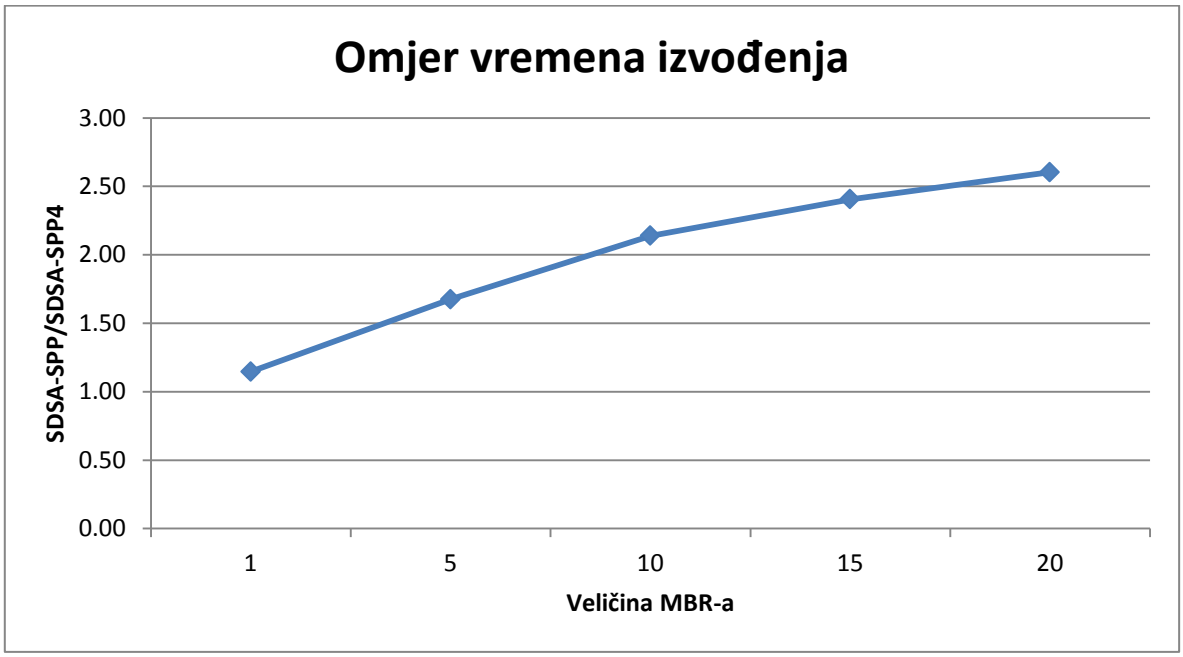


Slika 5.37 Vrijeme izvođenja serijskih i paralelnih algoritama za različite veličine minimalnog područja nesigurnosti

Povećanjem minimalnog područja nesigurnosti veća je vjerojatnost da će se stranice MBR-a presijecati simetralama, što dovodi do manjeg broja odbačenih grozdova. Broj očekivanih udaljenosti koje se moraju računati se povećava. I u ovome slučaju računanje ED se u paralelnim algoritmima dijeli na dva ili četiri dijela čime se smanjuje ukupno vrijeme izvođenja. Na slikama 5.38 i 5.39 prikazani su omjeri vremena izvođenja algoritama SDSA-SPP i SDSA-SPP2, te algoritama SDSA-SPP i SDSA-SPP4 za različite veličine stranica minimalnog područja nesigurnosti MBR. Sa slika je vidljivo da se s povećanjem veličine stranica MBR-a povećava njihov omjer. Dodatno vrijeme izvođenja, koje je posljedica većeg minimalnog područja nesigurnosti, dijeli se na dva ili četiri dijela. Time se paralelnim procesiranjem dobiva znatno skraćenje vremena izvođenja.



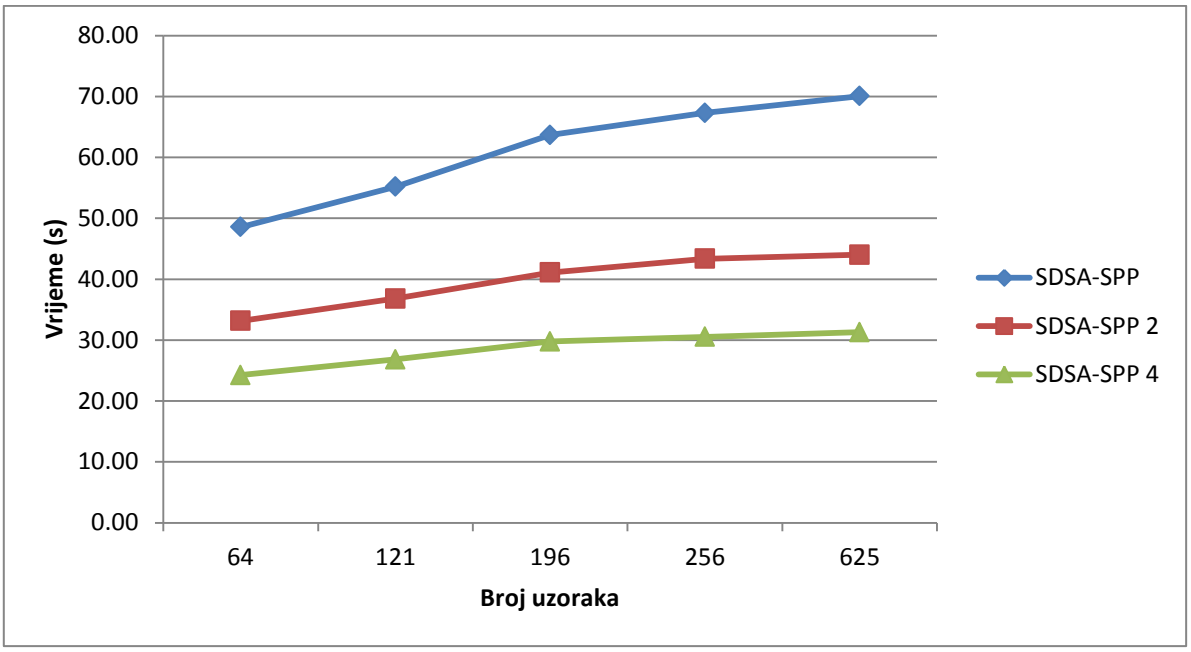
Slika 5.38 Omjer vremena izvođenja algoritama SDSA-SPP i SDSA-SPP2 za različite veličine minimalnog područja nesigurnosti



Slika 5.39 Omjer vremena izvođenja algoritama SDSA-SPP i SDSA-SPP4 za različite veličine minimalnog područja nesigurnosti

5.3.5. Pokusi u kojima se mijenja broj uzoraka PDF-a

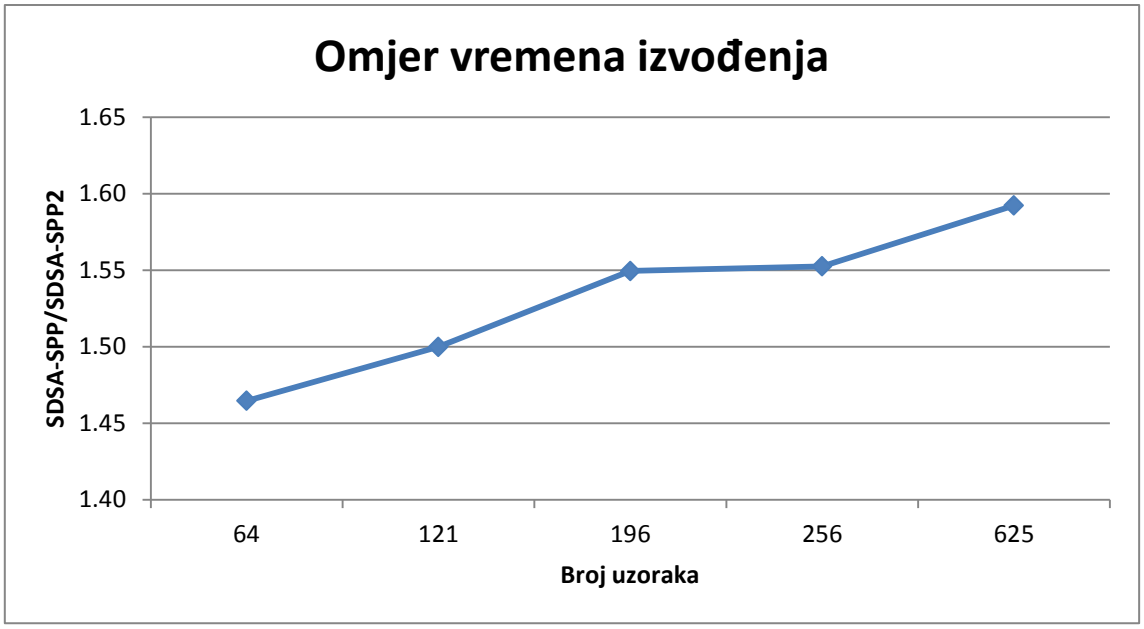
U ovim pokusima mijenja se broj uzoraka kojim je predstavljena funkcija gustoće vjerojatnosti PDF, dok ostali parametri zadržavaju osnovne vrijednosti. Pokusi kreću s 64 uzoraka za pojedini PDF i završavaju s maksimalna 625 uzorka. Za različite brojeve uzoraka PDF-a pokusi su ponovljeni dvadeset puta kako bi se dobila prosječna vrijednost vremena izvođenja koja su prikazana na slici 5.40.



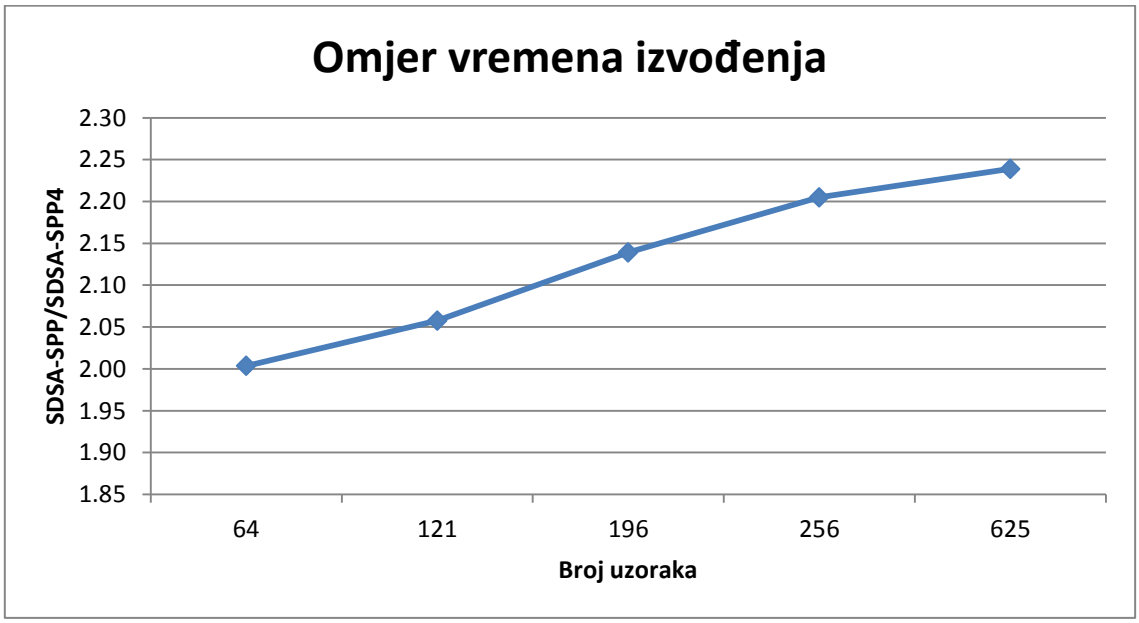
Slika 5.40 Vrijeme izvođenja serijskih i paralelnih algoritama za različit broj uzoraka

Povećanje broja uzoraka ne utječe na broj uspješno odbačenih grozdova, ali utječe na vrijeme računanja očekivane udaljenosti ED. Budući da se mora izračunati udaljenost između svakog uzorka i središta grozda, jasno je da će s povećanjem broja uzoraka računanje očekivane udaljenosti trajati dulje. Paralelnim procesiranjem to se vrijeme dijeli na dva ili četiri dijela, te se dobivaju značajne uštede u vremenu izvođenja.

Na slikama 5.41 i 5.42 prikazani su omjeri vremena izvođenja algoritama SDSA-SPP i SDSA-SPP2, te algoritama SDSA-SPP i SDSA-SPP4 za različit broj uzoraka kojima je predstavljena funkcija gustoće razdiobe. Sa slika je vidljivo da se povećanjem broja uzoraka povećava omjer serijske i paralelnih metoda. Dodatno vrijeme izvođenja, koje je posljedica većeg minimalnog područja nesigurnosti, dijeli se na dva ili četiri dijela. Time se paralelnim procesiranjem dobiva znatno skraćenje vremena izvođenja.



Slika 5.41 Omjer vremena izvođenja algoritama SDSA-SPP i SDSA-SPP2 za različit broj uzoraka



Slika 5.42 Omjer vremena izvođenja algoritama SDSA-SPP i SDSA-SPP4 za različit broj uzoraka

6. Simetralno razvrstavanja objekata uslužnih sustava

Postupci koji su predstavljeni u četvrtom poglavlju upotrijebljeni su za razvrstavanje objekata uslužnog sustava. Primjeri takvih sustava su taxi službe, hitna pomoć, nadzor zračnog prometa, cestovnog prometa i tako dalje. Uslužni sustavi su po svojoj prirodi dinamički i stalno se mijanjaju zahtjevi koje oni moraju ispuniti. Mogu se dogoditi izvanredne situacije za koje sustav, ako ih ranije ne predvidi, neće biti spreman. Kako bi se mogli pripremiti za buduće zahtjeve i izvanredne situacije rezultati razvrstavanja iskorišteni su opisivanje trenutnog stanja i predviđanje budućeg stanja uslužnog sustava. U tu svrhu korišteni su postojeći podaci o objektima uslužnog sustava. Pomoću razvrstavanja objekata uslužnog sustava nastoje se otkriti korisni podaci i pronaći sličnosti među njima. Otkrivanjem korisnih podataka može se saznati trenutno stanje uslužnog sustava i predvidjeti buduće stanje. Proučavanjem različitih situacija i podataka iz prošlosti možemo predvidjeti buduće stanje uslužnog sustava u sličnim situacijama. Predviđanje se može iskoristiti i pripremiti uslužni sustav kako bi što djelotvornije mogao odgovoriti na zahtjeve u budućnosti.

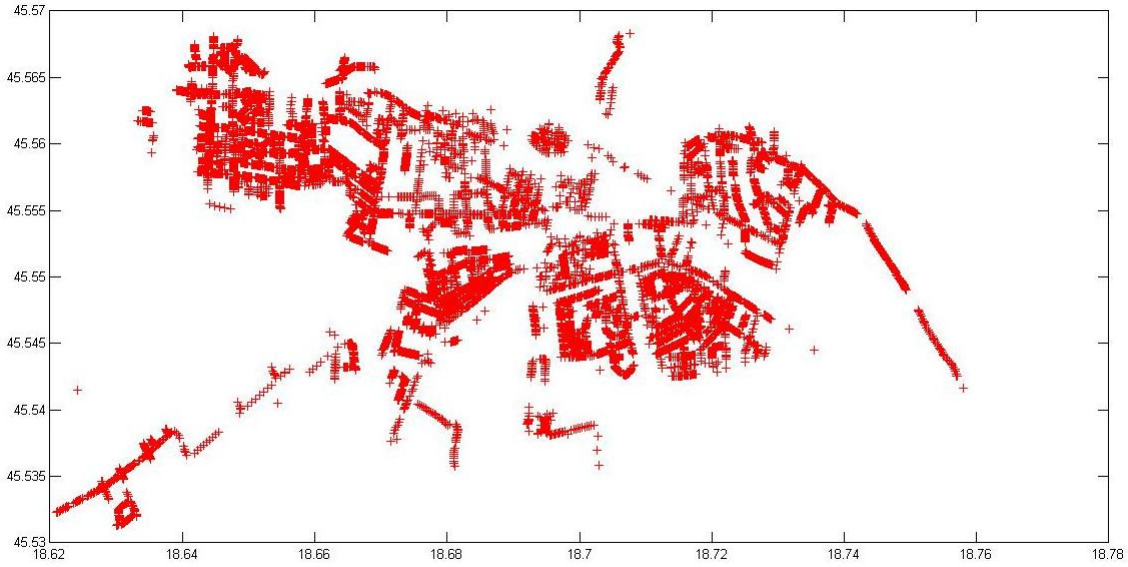
U ovoj disertaciji proučavano je djelovanje uslužnih sustava u Osijeku pa je za čuvanje podataka napravljena baza podataka koja sadrži zemljopisne koordinate svih kućnih brojeva u Osijeku. U bazi su spremljene sve postojeće ulice u gradu Osijeku, a za svaki kućni broj sprema se naziv ulice kojoj pripada i njegova zemljopisna širina i dužina. Za potrebe razvrstavanja ulice su podijeljene na dijelove i svaki dio ulice predstavlja minimalno područje nesigurnosti jednog nesigurnog objekta. Veličina minimalnog područja nesigurnosti određena je veličinom pojedinog dijela ulice. Budući da dijelovi ulica nisu jednake veličine dobivaju se nesigurni objekti s različitim veličinama minimalnog područja nesigurnosti. No pokusima u petom poglavlju pokazano je da SPP algoritam ima dobra svojstva za sve veličine minimalnog područja nesigurnosti. Nesigurni objekti uzorkovani su tako da svaki kućni broj predstavlja jedan uzorak u funkciji gustoće vjerojatnosti. Broj uzoraka kojim je predstavljena funkcija gustoće vjerojatnosti ovisi o tome na koliko je dijelova podijeljena jedna ulica, to jest koliko kućnih brojeva ima u tome dijelu ulice. Vjerojatnost da se objekt nalazi u nekom uzorku predstavljat će broj zadataka koje je uslužni sustav obavio na tome uzorku. Što je broj obavljenih zadataka na nekome uzorku veći, veća je vjerojatnost da će u budućnosti trebati obaviti zadatak na istome uzorku.

Kako bi se čuvali podaci o zadacima uslužnog sustava napravljena je baza podataka u kojoj se nalaze vremena svih zadataka koje je uslužni sustav obavio na pojedinom uzorku. Iz baze podataka se za svaki uzorak može saznati broj i vrijeme obavljenih zadataka, a iz njih se mogu izračunati vjerojatnosti za svaki uzorak. Na osnovu postojećih podataka o pojedinom uzorku napraviti će se predviđanje stanja uslužnog sustava. Pomoću predviđanja može se saznati gdje će se u budućnosti ukazati potreba za uslužnim sustavom, što će znatno uštedjeti sredstva za održavanje uslužnog sustava i vrijeme reakcije (vrijeme potrebno da se obavi zadatak). Razvrstavanjem postojećih podataka određena su središta grozdova, koja određuju mjesta na kojima je najveća koncentracija zadataka za uslužni sustav. Uslužni sustav može iskoristiti predviđanje i rasporediti svoje djelatnike u središtima grozdova kako bi što djelotvornije mogli ispuniti zahtjeve za uslužnim sustavom. U idućim poglavljima izložiti će se rezultati sustavne analize prostorno-vremenskih značajki uslužnog sustava. Pod vremenskim značajkama podrazumijevaju se različiti vremenski okviri u kojima se mijenjaju zahtjevi za uslužnim sustavom. Budući da su potrebe za uslužnim sustavom ovisne o različitim događajima u gradu, napraviti će se predviđanja za pojedine izvanredne situacije koje značajno mijenjaju zahtjeve za uslužnim sustavom. Na osnovu provedenih analiza predložiti će se načini za poboljšanje djelovanja uslužnog sustava.

6.1. Zemljopisne koordinate kućnih brojeva u Osijeku

Uslužni sustavi moraju obavljati zadatke u stvarnom svijetu, pa je u baze podataka potrebno spremiti pozicije i vremena obavljana tih zadataka. Za proces razvrstavanja nije dovoljno spremiti naziv ulice i kućni broj na kojem je obavljen zadatak, jer ti podaci računalu nisu razumljivi i teško ih je matematički obraditi. Računalu su potrebni brojčani podaci koje može obraditi. U ovom radu promatran je uslužni sustav koji obavlja zadatke u Osijeku pa je bilo potrebno pribaviti zemljopisne koordinate svih kućnih brojeva u gradu. Budući da takvi podaci ne postoje na jednome mjestu ili nisu javno dostupni izrađena je skripta koja ima mogućnost pribaviti zemljopisne koordinate većine gradova u svijetu. Skripta koristi usluge Yahoo-ovog servisa PlaceFinder. Na samome servisu ne može se pronaći popis zemljopisnih koordinata. Servis nudi sučelje kojemu morate zadati naziv grada, ulicu i kućni broj, a kao rezultat dobiti .xml datoteku sa zemljopisnim koordinatama samo jednog kućnog broja. Ručno zadavanje svih ulica i kućnih brojeva bio bi nemoguć zadatak, jer se radi o velikim količinama podataka. U ovoj disertaciji razvijena je skripta koja automatizira postupak

dobivanja zemljopisnih koordinata. Skripta kao ulazni parametar zahtijeva popis ulica u gradu i kao rezultat vraća zemljopisne koordinate kućnih brojeva i sprema ih u bazu. Servis ne javlja koliko pojedina ulica ima kućnih brojeva pa je to sama skripta morala prepoznati i prestati s radom na kraju ulice. Također, u pojedinim ulicama ne postoje pojedini kućni brojevi i to su također situacije koje skripta mora prepoznati. Kao ulazni parametar predane su sve ulice u Osijeku i dobivene su zemljopisne koordinate potrebne za proces razvrstavanja i praćenja stanja uslužnog sustava. Kako bi se dokazala ispravnost koordinata one su iscrtane i kao rezultat dobiveni su obrisi grada Osijeka prikazani na slici 6.1.



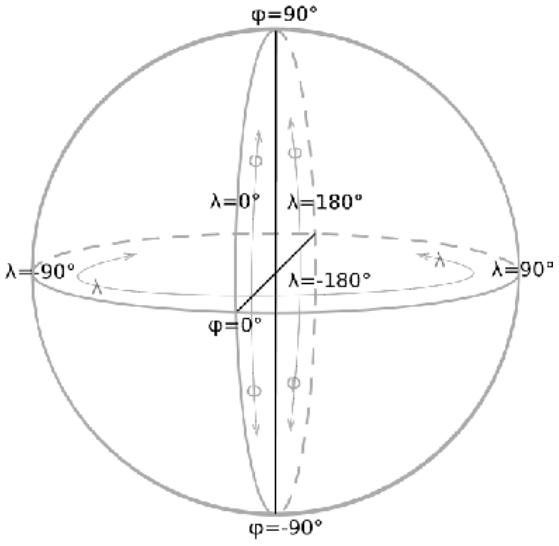
Slika 6.1 Zemljopisne koordinate grada Osijeka dobivene razvijenom skriptom

Prije spremanja zemljopisnih koordinata u bazu obavljena je podjela ulica na dijelove, to jest na nesigurne objekte. Svaka ulica uzorkovana je tako da se u njoj nalazi dvadeset uzoraka. Pri spremanju koordinata u bazu vodilo se računa o tome kojoj ulici i kojem nesigurnom objektu pripada zemljopisna koordinata, jer se u jednoj ulici može nalaziti i nekoliko desetaka nesigurnih objekata. Osijek ima oko sto tisuća stanovnika i na kraju je dobiveno približno 900 nesigurnih objekata. Za velike gradove taj broj bi bio i nekoliko desetaka tisuća, ovisno o veličini grada. Pribavljanjem zemljopisnih koordinata preostalo je u bazu spremati prethodne zadatke uslužnog sustava kako bi se mogla izračunati funkcija gustoće vjerojatnosti. Na osnovi dobivenih koordinata i funkcije gustoće vjerojatnosti napravljene su računalne simulacije djelovanja uslužnog sustava u Osijeku. Simulacije su obavljane za različite događaje koji utječu na stanje uslužnog sustava. Grad je dinamičan sustav i svakodnevno

postoje nova događanja na različitim mjestima u gradu. Na događanjima se pojavljuje veća koncentracija ljudi, a time se mijenjaju zadaci uslužnog sustava u odnosu na uobičajeno stanje. Napravljene su simulacije u kojima se zbog izvanrednih događaja okupljaju veće skupine ljudi. Zbog povećane koncentracije ljudi na jednom mjestu i promjene zahtjeva za uslužnim sustavom predviđanja su potrebna kako bi uslužni sustav mogao djelovati prema novim zahtjevima i prilagoditi se situaciji. Pomoću predviđanja sustav zna koji zahtjevi ga očekuju i može se pripremiti na njih. Bez predviđanja sustav ne bi mogao pravovremeno odgovoriti na zahtjeve i došlo bi do problema u radu uslužnog sustava.

6.2. Pretvorba zemljopisnih koordinata u Kartezijeve koordinate

Zemljopisne koordinate kućnih brojeva u gradu Osijeku predstavljene su u sfernom koordinatnom sustavu koji koristi tri parametra za predstavljanje pojedine koordinate u 3D prostoru. U slučaju zemljopisnih koordinata parametri su radijalna udaljenost r od ishodišta do točke u prostoru, zemljopisna širina koja se označava s φ i zemljopisna dužina koja se označava s λ. Za predstavljanje zemljopisnih koordinata pomoću sfernog sustava podrazumijeva se da je središte sustava u središtu Zemlje, radijalna udaljenost r jednaka je polumjeru Zemlje i iznosi približno 6371 kilometar. Na sljedećoj slici prikazana je grafička prezentacija zemljopisnih koordinata pomoću sfernog koordinatnog sustava.



Slika 6.2 Prikaz zemljopisnih koordinata u sfernom koordinatnom sustavu

Pretvorba iz sfernog koordinatnog sustava u Kartezijev koordinatni sustav može se napraviti pomoću sljedećih jednadžbi:

$$z = r cos\lambda \tag{6.1}$$

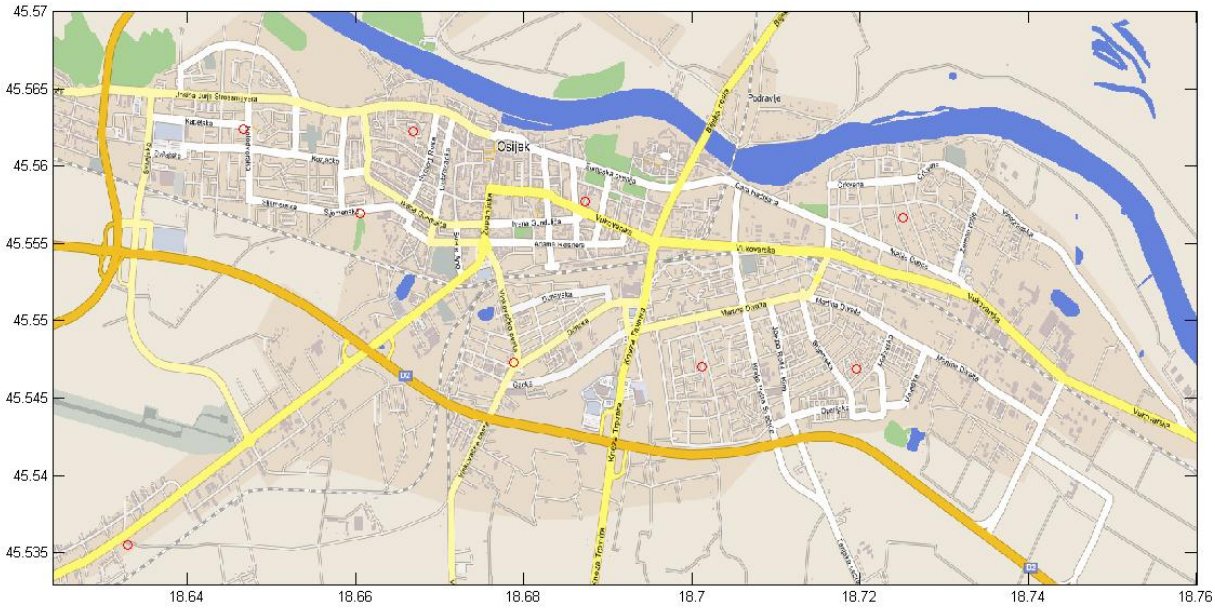
$$x = r cos\varphi cos\lambda \tag{6.2}$$

$$y = r cos\varphi sin\lambda \tag{6.3}$$

Za potrebe razvrstavanja z koordinata nije bila potrebna tako da ona nije korištena. Pretvorbom su iz zemljopisne širine i dužine dobivene koordinate u Kartezijevom koordinatnom sustavu. Pretvorba je bila potrebna kako bi se mogle obavljati matematičke operacije pri procesu razvrstavanja, jer postupci za odbacivanje grozdova nisu prilagođeni za sferni koordinatni sustav.

6.3. Pokusi pri uobičajenom radu uslužnog sustava

U ovom poglavlju napravljeni su pokusi u kojima uslužni sustav obavlja uobičajene zadatke koji su podjednako raspoređeni po čitavom gradu, ali u ovisnosti o broju stanovnika. To znači da je broj zadataka jednolično raspoređen po čitavome gradu i ne postoje dijelovi grada u kojima se nalazi znatno veći broj zadataka. Pokusima je napravljeno predviđanje stanja uslužnog sustava, kad iz svih dijelova grada dolazi približno jednak broj zadataka. Ne postoje veća okupljanja koja imaju veću koncentraciju zadataka na jednom mjestu. Kako bi se pronašlo što više mogućih rješenja pokusi su napravljeni za različit broj grozdova u kojima se može koncentrirati uslužni sustav. Rezultati su međusobno uspoređeni i provjereno je koje rješenje daje bolje rezultate. U prva dva pokusa objekti se razvrstavaju u devet i šesnaest grozdova i rezultati su prikazani na slikama 6.3 i 6.4.



Slika 6.3 Razvrstavanje objekata u devet grozdova pri uobičajenom radu uslužnog sustava

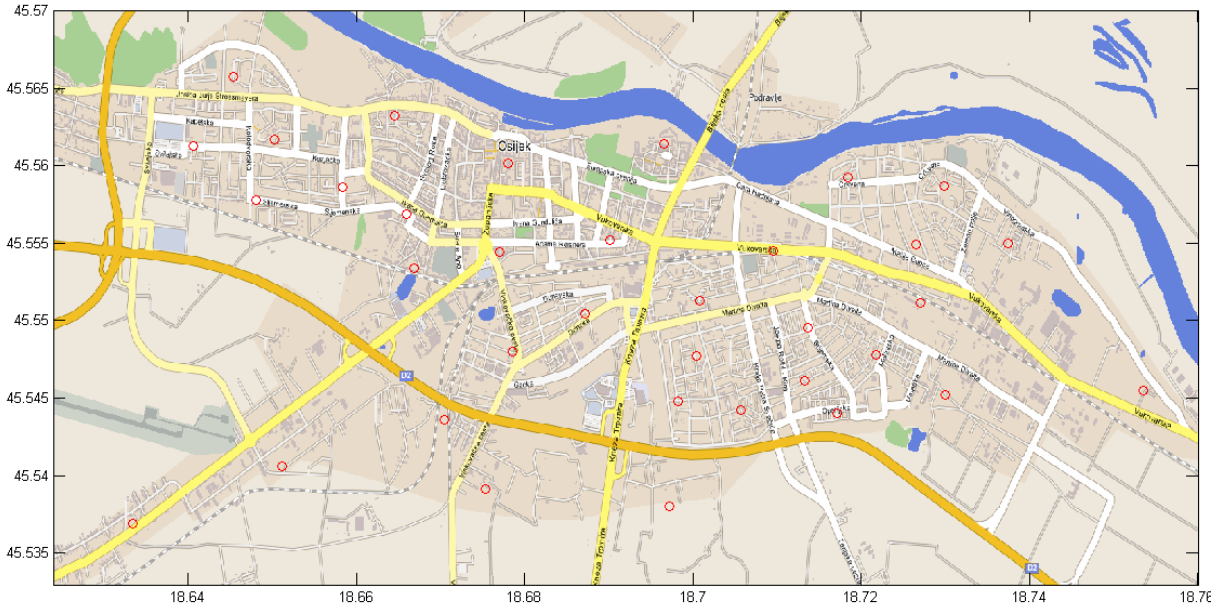
Slika 6.4 Razvrstavanje objekata u šesnaest grozdova pri uobičajenom radu uslužnog sustava

Na slikama 6.3 i 6.4 vidljiva su središta grozdova označena crvenim kružnicama. Osim same pozicije grozda u bazu se sprema i broj predviđenih zadataka u blizini grozda, kako bi se tamo mogao rasporediti predviđeni broj djelatnika. U ovim pokusima se u svakom dijelu grada pojavljuje približno jednak broj zadataka, a utjecaj broja zadataka biti će obrađen u

narednim pokusima. Na taj način napravljeno je razvrstavanje objekata prema zemljopisnim značajkama grada Osijeka. U ovom pokusu napravljena je podjela na devet i šesnaest grozdova. Na slikama 6.3 i 6.4 je vidljivo koji dijelovi grada predstavljaju središta grozdova i u njima uslužni sustav treba smjestiti odgovarajući broj djelatnika. Broj djelatnika za svaki grozd može se saznati iz predviđenog broja zadataka za pojedini grozd spremljenih u bazi podataka. Vidljivo je kako su središta grozdova smještena u najgušće naseljenim dijelovima grada što je i očekivano, jer se na početku pokusa pretpostavilo da svi dijelovi grada imaju iste zahtjeve za uslužnim sustavom u ovisnosti o broju stanovnika. Posljedično tome središta grozdova nalaze se u onim dijelovima grada u kojima se nalazi najveći broj zadataka.

U sljedećim pokusima korišteni su isti podaci kao i u prethodna dva pokusa, ali je ovaj put broj grozdova povećan na 25 i 36. Rezultati su prikazani na slikama 6.5 i 6.6.

Slika 6.5 Razvrstavanje objekata u 25 grozdova pri uobičajenom radu uslužnog sustava



Slika 6.6 Razvrstavanje objekata u 36 grozdova pri uobičajenom radu uslužnog sustava

Na slikama 6.5 i 6.6 crvenim kružnicama označeno je 25 i 36 grozdova u kojima uslužni sustav treba rasporediti svoje djelatnike. Budući da se središta grozdova nalaze na mjestima sa najmanjom ukupnom udaljenosti od nesigurnih objekata koji ga okružuju, raspoređivanjem u središtima grozdova znatno se smanjuje put djelatnika uslužnog sustava do zadataka koje moraju obaviti. Vidljivo je kako se povećanjem broja grozdova njihova središta približavaju, iz čega se može zaključiti da nije potrebno više od šesnaest grozdova.

U ovim pokusima mjerene su udaljenosti od središta grozdova do zadataka uslužnog sustava, te udaljenosti od proizvoljnih mjesta u gradu do tih istih zadataka. Proizvoljna mjesta u gradu su ona mjesta u kojima bi se uslužni sustav smjestio bez korištenja razvrstavanja i saznanja koje mu ono nudi. To su mjesta u kojima bi se slučajno smjestio djelatnik uslužnog sustava dok čeka novi zadatak. Rezultati pokusa su pokazali kako je korištenjem razvrstavanja udaljenost od središta grozdova do mogućih zadataka između 35% do 50% manja nego udaljenost od proizvoljnih mjesta u gradu do mogućih zadataka. Ovisno o kojem se uslužnom sustavu radi slijedi da se troškovi prijevoza i vrijeme reakcije mogu značajno smanjiti zbog smanjenja prijeđene udaljenosti. Isto tako ako se radi o uslužnom sustavu koji se nalazi na jednome mjestu, a korisnici dolaze po njegove usluge, slijedi da je on do 50% bliži svojim korisnicima i da će njegove usluge biti dostupnije i pristupačnije. Razvrstavanjem se značajno može poboljšati djelovanje uslužnog sustava, jer ono nudi najbolji mogući raspored dijelova uslužnog sustava koji su ovisni o okolnim značajkama. Pri planiranju razvoja uslužnog

sustava razvrstavanje može pomoći pri odaberu najboljeg rasporeda uslužnog sustava, a tijekom rada na osnovi postojećih podataka razvrstavanje se koristi za predviđanje budućeg stanja uslužnog sustava. U idućem poglavlju biti će pokazano kako se mogu iskoristiti postojeći podaci o uslužnom sustavu te na osnovu njih predvidjeti buduće ponašanje u izvanrednim situacijama i pravovremeno djelovati na značajne promjene u sustavu.

6.4. Pokusi s izvanrednim događajima u uslužnom sustavu

U ovom poglavlju napravljeni su pokusi za različite događaje u kojima se značajno mijenjaju pozicije i broj zadataka koje mora obaviti uslužni sustav. Pokusi su napravljeni za izvanredne događaje kao što su veća okupljanja ljudi na pojedinim mjestima u gradu. U tim situacijama na pojedinim mjestima značajno se mijenja broj zadataka koje uslužni sustav treba obaviti, a samim time mijenjaju se i središta grozdova koja su ovisna o tim zadacima. Svrha razvrstavanja je predvidjeti buduće stanje uslužnog sustava pri ovakvim situacijama kako bi uslužni sustav mogao djelovati što učinkovitije. U prošlom poglavlju zadaci uslužnog sustava bili su ravnomjerno raspoređeni, no sada se u pojedinim dijelovima grada značajno povećava broj zadataka, dok u drugim dijelovima grada broj zadataka stagnira ili opada. Kao primjer uzmimo uslužni sustav prijevoza putnika.

Ako su zadaci ravnomjerno raspoređeni, uslužni sustav treba se postaviti u središtima grozdova dobivenim u prethodnom poglavlju. No u izvanrednim situacijama broj zadataka u pojedinim dijelovima grada značajno se povećava i središta grozdova se pomiču prema tim dijelovima grada. Uslužni sustav mora to predvidjeti i djelovati prema novim zahtjevima. Osim pomicanja središta grozdova bitan je i veći broj usluga koje treba obaviti u promatranim dijelovima grada. U tim grozdovima potrebno je postaviti više djelatnika uslužnog sustava zbog povećanog opterećenja. Osim spremanja pozicije središta grozdova u bazu, sprema se i broj zadataka koje treba obaviti u svakom pojedinom grozdu. Ti se podaci koriste kako bi uslužni sustav mogao predvidjeti buduće stanje i pripremiti se za veća opterećenja u pojedinim grozdovima. U grozdovima s većim opterećenjima treba postaviti više djelatnika, a u grozdovima s manjim opterećenjima manji broj djelatnika. Na taj način djelatnici su optimalno iskorišteni i time se postiže smanjenje troškova uslužnog sustava te veće zadovoljstvo korisnika zbog bržeg vremena reakcije.

6.4.1. Pokusi s povećanim opterećenjem na autobusnoj i željezničkoj stanici

Pokusi su napravljeni kao primjer u slučajevima kad većina putnika dolazi u grad preko autobusne i željezničke stanice. Pretpostavka je da će određeni broj putnika zahtijevati usluge sustava prijevoza i tako povećati zahtjeve za uslužnim sustavom na tome području. Sukladno tome najveći broj zadataka dolazit će iz susjednih ulica. Za potrebe pokusa simuliran je skup postojećih podataka u kojima je povećan broj zadataka u ulicama blizu autobusne i željezničke stanice. Nakon toga podaci su spremljeni u bazu podataka pa razvrstavanje koristi simulirane podatke iz baze i razvrstavanjem tih podataka nastoji predvidjeti stanje uslužnog sustava u nastaloj situaciji. U istom pokusu promatrana je situacija kad se poveća broj zadataka u području KBC-a Osijek. Određeni dio putnika dolazi na preglede ili u posjete pa se i tamo povećava broj zahtjeva u odnosu na ostale dijelove grada. Stoga je u skupu simuliranih zadaka povećan broj zahtjeva u blizini KBC-a Osijek, dok je u ostalim dijelovima grada broj zadataka manji. Na osnovni predviđanja budućeg stanja uslužni sustav raspoređuje svoje djelatnike ovisno o broju zadataka u pojedinim dijelovima grada. Pokusi su provedeni za dva različita slučaja. U prvom slučaju grad je podijeljen na devet grozdova kao što je prikazano na slici 6.7, te na šesnaest grozdova kao što je prikazano na slici 6.8.

Slika 6.7 Razvrstavanje objekata na devet grozdova u slučaju povećanog opterećenja na autobusnoj i željezničkoj stanici

Slika 6.8 Razvrstavanje objekata na šesnaest grozdova u slučaju povećanog opterećenja na autobusnoj i željezničkoj stanici

Na slikama 6.7 i 6.8 grozdovi su prikazani crvenim kružnicama. Grozdovi u kojima je veći broj zadataka prikazani su većim i podebljanim crvenim kružnicama, dok su grozdovi s manjim brojem zadataka prikazani manjim crvenim kružnicama. Na obje slike je vidljivo kako postoje dva grozda s povećanim brojem zadataka za uslužni sustav. Prvi grozd nalazi se u blizini autobusne i željezničke stanice, a drugi grozd se nalazi pored KBC-a Osijek. Dobiveni rezultati su očekivani, jer je pokus proveden za slučaj u kojem se pojavljuje veći broj zadataka na navedenim pozicijama. Osim same pozicije grozdova razvrstavanje objekata uslužnog sustava kao rezultat daje i broj zadataka u pojedinom grozdu. Na osnovi broja zadataka može se predvidjeti broj djelatnika koje je potrebno smjestiti u području svakog grozda. Tako uslužni sustav može pravovremeno djelovati i imati optimalni broj djelatnika u svakom grozdu. Iz rezultata pokusa može se zaključiti kako je najveći broj djelatnika potrebno raspodijeliti u ulicu Cara Hadrijana pored koje se nalazi KBC Osijek, te na križanju ulica Grgura Čevapovića i Bartola Kašića pored kojeg se nalaze autobusna i željeznička stanica. U sve preostale grozdove u gradu potrebno je rasporediti manji broj djelatnika, koji se također može izračunati iz predviđenog broja zadataka spremljenog u bazi podataka.

6.4.2. Pokusi s povećanim opterećenjem na tržnicama

Pokusi su napravljeni kao primjer u slučajevima kad većina građana odlazi na tržnicu kupiti namirnice, to jest za glavne dane na tržnicama. Pretpostavka je da će određeni broj građana zahtijevati usluge sustava prijevoza kako bi prevezli kupljene stvari i tako povećati zahtjeve za uslužnim sustavom na području tržnica u gradu. Za potrebe pokusa simuliran je skup postojećih podataka u kojima je povećan broj zadataka u ulicama koje se nalaze u blizini glavnih tržnica u Osijeku. Nakon toga podaci su spremljeni u bazu podataka pa razvrstavanje koristi simulirane podatke iz baze i razvrstavanjem tih podataka nastoji predvidjeti stanje uslužnog sustava u nastaloj situaciji. Razvrstavanjem postojećih podataka iz baze nastoji predvidjeti buduće stanje uslužnog sustava kako bi uslužni sustav mogao rasporediti svoje djelatnike u ovisnosti o dobivenim rezultatima. Pokusi su provedeni za dva različita slučaja. U prvom slučaju grad je podijeljen na devet grozdova kao što je prikazano na slici 6.9, te na šesnaest grozdova kao što je prikazano na slici 6.10.

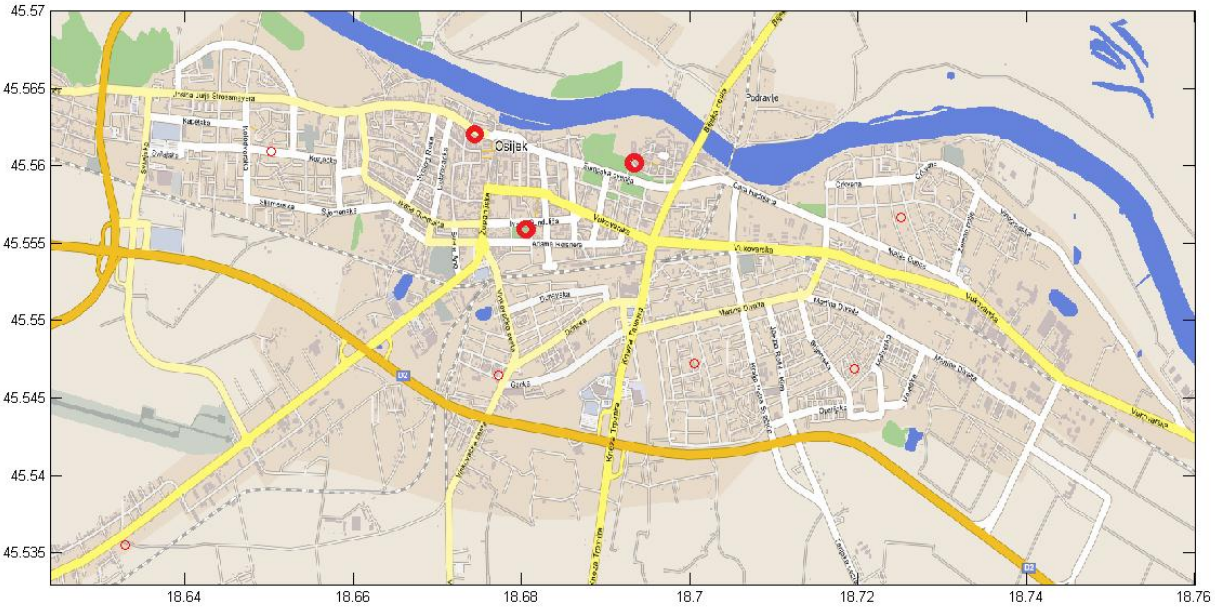
Slika 6.9 Razvrstavanje objekata na devet grozdova u slučaju povećanog opterećenja u blizini gradskih tržnica

Slika 6.10 Razvrstavanje objekata na šesnaest grozdova u slučaju povećanog opterećenja u blizini gradskih tržnica

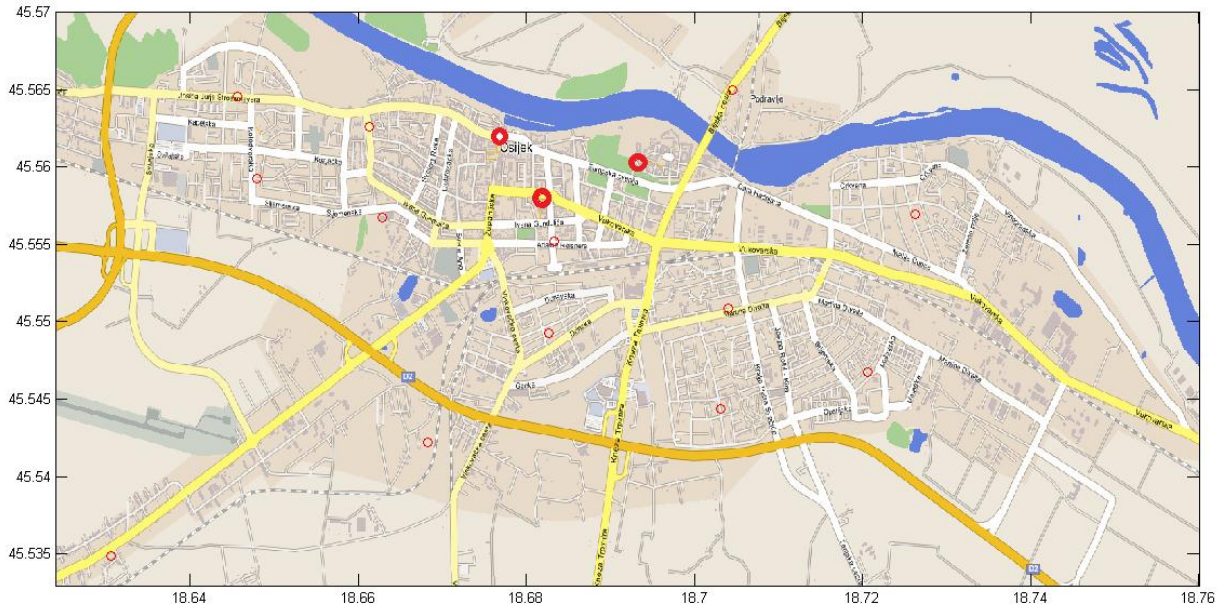
Na slikama 6.9 i 6.10 grozdovi su prikazani crvenim kružnicama. Grozdovi u kojima je veći broj zadataka prikazani su većim i podebljanim crvenim kružnicama, dok su grozdovi s manjim brojem zadataka prikazani manjim crvenim kružnicama. Na obje slike je vidljivo kako postoje četiri grozda s većim brojem zadataka za uslužni sustav. Prvi grozd s povećanim brojem zadataka nalazi se u blizini tržnice na Trgu Ljudevita Gaja, drugi grozd u blizini tržnice na Trgu bana Josipa Jelačića, treći grozd pored ulice Ljudevita Posavskog i četvrti pored Bosutskog naselja. Dobiveni rezultati su očekivani, jer su pokusi provedeni za slučajeve u kojima veći broj građana odlazi na tržnice. I u ovom slučaju razvrstavanje objekata uslužnog sustava kao rezultat daje i broj zadataka u pojedinom grozdu, pa se može predvidjeti broj djelatnika koje treba smjestiti u području svakog grozda. Uslužni sustav može pravovremeno djelovati i rasporediti djelatnike prema predviđenom stanju uslužnog sustava. Iz rezultata pokusa može se zaključiti kako je najveći broj djelatnika potrebno raspodijeliti u četiri ranije spomenuta grozda, dok je manji broj djelatnika potrebno rasporediti po preostalim grozdovima u gradu. Budući da su podaci simulirani nije bitan broj zadataka u pojedinom grozdu već je bitan samo omjer iz kojeg se može saznati u kojim grozdovima su veća opterećenja.

6.4.3. Pokusi s povećanim opterećenjem oko središta zabave

Pokusi su napravljeni kao primjer kad građani, većinom mlađa populacija, odlaze u izlaske i okupljaju se oko središta zabave i drugih mjesta predviđenih za zabavu u gradu. Pretpostavka je da će dio njih zahtijevati usluge sustava prijevoza kako bi došli ili se vratili s mjesta oko središta zabave. To će znatno povećati zahtjeve za uslužnim sustavom na području glavnih mjesta zabave u gradu. Za potrebe pokusa simuliran je skup postojećih podataka u kojima je povećan broj zadataka u ulicama koje se nalaze u blizini glavnih središta zabave u Osijeku. Podaci su spremljeni u bazu podataka pa razvrstavanje koristi simulirane podatke iz baze i razvrstavanjem tih podataka nastoji predvidjeti stanje uslužnog sustava u nastaloj situaciji kako bi uslužni sustav mogao rasporediti svoje djelatnike u ovisnosti o dobivenim rezultatima. Pokusi su provedeni za dva različita slučaja. U prvom slučaju grad je podijeljen na devet grozdova kao što je prikazano na slici 6.11, i šesnaest grozdova kao što je prikazano na slici 6.12.



Slika 6.11 Razvrstavanje objekata na devet grozdova u slučaju povećanog opterećenja oko središta zabave



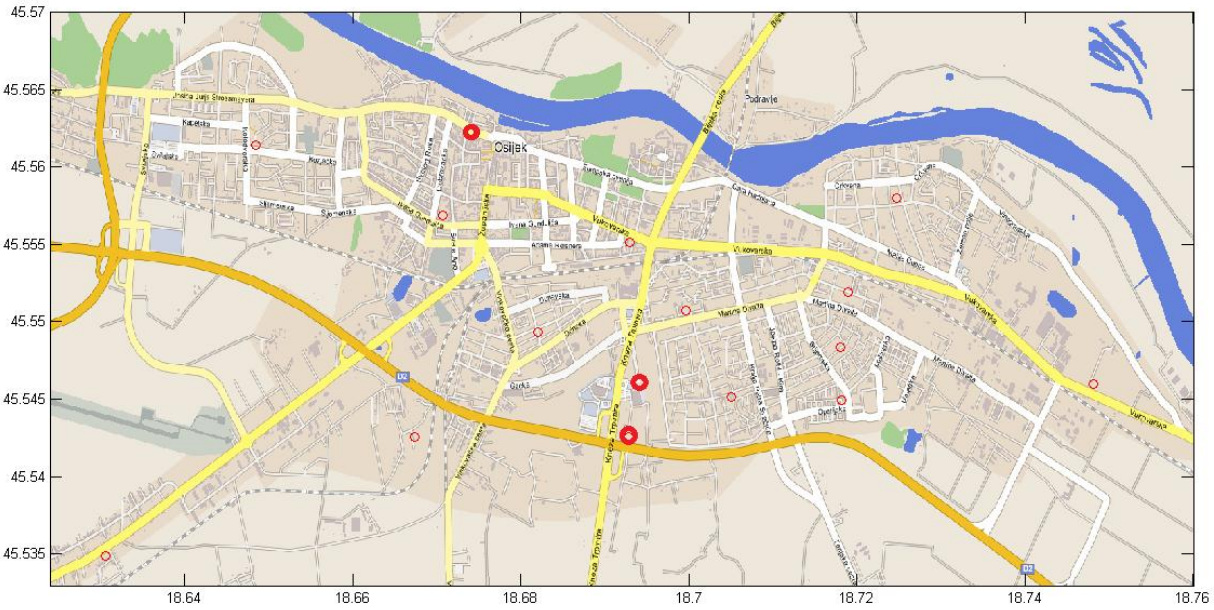
Slika 6.12 Razvrstavanje objekata na šesnaest grozdova u slučaju povećanog opterećenja oko središta zabave

Na slikama 6.11 i 6.12 grozdovi su prikazani crvenim kružnicama. Grozdovi u kojima je veći broj zadataka prikazani su većim i podebljanim crvenim kružnicama, dok su grozdovi s manjim brojem zadataka prikazani manjim crvenim kružnicama. Na obje slike je vidljivo kako postoje tri grozda s većim brojem zadataka za uslužni sustav. Prvi grozd s povećanim brojem zadataka nalazi se u ulici Stjepana Radića, drugi grozd u blizini Promenade, a treći grozd nalazi se u Tvrđi. Dobiveni rezultati su očekivani, jer je treći pokus proveden za slučaj kada većina mladih ide u zabavu. I u ovom slučaju razvrstavanje objekata uslužnog sustava kao rezultat daje broj zadataka u pojedinom grozdu, pa se može predvidjeti očekivani broj potrebnih djelatnika u području svakog grozda. Uslužni sustav može pravovremeno djelovati i rasporediti svoje djelatnike prema predviđanjima. Iz rezultata pokusa može se zaključiti kako je najveći broj djelatnika potrebno raspodijeliti u tri ranije spomenuta grozda, dok je manji broj djelatnika potrebno rasporediti po preostalim grozdovima u gradu.

6.4.4. Pokusi s povećanim opterećenjem zbog okupljanja građana

Pokusi su napravljen kao primjer iznimnih situacija kad se veći broj građana okuplja na jednom mjestu. Iznimne situacije mogu biti koncerti, sportska natjecanja ili različiti skupovi. Pretpostavka je da će zbog većeg broja ljudi biti povećani zahtjevi za uslugama sustava prijevoza. Zahtjevi će se povećati na područjima glavnih mjesta za okupljanje u gradu. Razvrstavanje i u ovom slučaju koristi prethodne podatke za slične situacije u prošlosti. Za potrebe pokusa simuliran je skup postojećih podataka u kojima je povećan broj zadataka u blizini glavnih okupljališta u Osijeku. Podaci su spremljeni u bazu podataka pa razvrstavanje koristi simulirane podatke iz baze i razvrstavanjem tih podataka nastoji predvidjeti stanje uslužnog sustava u nastaloj situaciji kako bi uslužni sustav mogao rasporediti svoje djelatnike u ovisnosti o dobivenim rezultatima. Pokusi su proveden za dva različita slučaja. U prvom slučaju grad je podijeljen na devet grozdova kao što je prikazano na slici 6.13, i šesnaest grozdova kao što je prikazano na slici 6.14.

Slika 6.13 Razvrstavanje objekata na devet grozdova u slučaju povećanog opterećenja zbog okupljanja građana



Slika 6.14 Razvrstavanje objekata na šesnaest grozdova u slučaju povećanog opterećenja zbog okupljanja građana

Na slikama 6.13 i 6.14 grozdovi su prikazani crvenim kružnicama. Grozdovi u kojima je veći broj zadataka prikazani su većim i podebljanim crvenim kružnicama, dok su grozdovi s manjim brojem zadataka prikazani manjim crvenim kružnicama. Sa slike 6.13 je vidljivo kako postoje dva grozda s povećanim brojem zadataka, dok su na slici 6.14 tri grozda s povećanim brojem zadataka za uslužni sustav. Prvi grozd nalazi se u ulici blizini Trga Ante Starčevića, a druga dva grozda sa slike 6.14 u blizini dvorane Gradski Vrt. Dobiveni rezultati su očekivani, jer je treći pokus proveden za slučaj većih okupljanja građana na navedenim mjestima. I u ovom slučaju razvrstavanje objekata uslužnog sustava kao rezultat daje i broj zadataka u pojedinom grozdu, pa se može predvidjeti očekivani broj potrebnih djelatnika u području svakog grozda, i uslužni sustav može pravovremeno djelovati i rasporediti djelatnike. Iz rezultata pokusa može se zaključiti kako je najveći broj djelatnika potrebno raspodijeliti u tri ranije spomenuta grozda, dok je manji broj djelatnika potrebno rasporediti po preostalim grozdovima u gradu.

6.5. Analiza dobivenih rezultata

U prethodnim poglavljima pokusima su simulirane različite situacije koje mogu značajno promijeniti zahtjeve za uslužnim sustavom. Simulacijama se nastojalo predvidjeti nekoliko mogućih situacija koje se mogu dogoditi i značajno promijeniti stanje uslužnog sustava. U obzir je uzeto nekoliko mogućih događaja i opisano stanje uslužnog sustava u svakom od njih. Pokusi su pokazali kako uslužni sustav svaki događaj mora promatrati odvojeno i djelovati drugačije pri svakom događaju. U praktičnoj upotrebi broj objekata u bazi o prethodnim zadacima može biti jako velik i nemoguće je obraditi sve podatke bez upotrebe računala. Upotrebom razvrstavanja iz mnoštva podataka nastojalo se izvući korisne podatke za pojedine događaje, iz njih saznati novo znanje i predvidjeti buduće stanje uslužnog sustava.

Pokusima se pokazalo kako je moguće pronaći najbolje rješenja, to jest pozicije u kojima treba smjestiti djelatnike uslužnog sustava. Kao najbolje rješenje treba odabrati središta grozdova, jer ona imaju najmanju ukupnu udaljenost do svih mogućih zadataka u okolini središta grozdova. U svakom pokusu mjerena je udaljenost od središta grozda do mogućih zadataka u okolini, te udaljenost od najbliže slučajno odabrane pozicije do zadataka u okolini. Ako se ne koristi razvrstavanje nije moguće odabrati najbolju poziciju za smještanje djelatnika uslužnog sustava nego se ona odabire slučajnim odabirom ili trenutno najbližom slobodnom pozicijom u gradu.

Pokusi su ponovljeni pedeset puta i svaki put se mjerila ranije spomenuta ukupna udaljenost. Ukupna udaljenost ovisila je o slučajno odabranim pozicijama u gradu u koje se smještaju djelatnici uslužnog sustava bez korištenja razvrstavanja. Budući da se odabiru slučajnim odabirom one mogu biti znatno udaljene od središta grozdova i tada će se njihova udaljenost do mogućih zadataka biti velika. Isto tako slučajne pozicije mogu biti blizu središtima grozdova i tada će njihova udaljenost do mogućih zadataka biti manja. Zbog toga su pokusi ponovljeni pedeset puta kako bi se dobila srednja vrijednost udaljenosti od slučajnih pozicija u gradu do mogućih zadataka. Budući da se središta grozdova ne mijenjaju tijekom ponavljanja pokusa udaljenost od središta grozdova do mogućih zadataka uvijek ostaje ista. Pokusi su pokazali kako se korištenjem razvrstavanja ukupna udaljenost do mogućih zadataka u prosjeku smanjuje između 35% i 50% što znači da će razvrstavanje objekata uslužnog sustava smanjiti prijeđenu udaljenost i do 50% u odnosu na slučaj bez korištenja razvrstavanja. Zbog smanjenja udaljenosti može se uštedjeti na održavanju uslužnog sustava kao što su troškovi prijevoza. To je prva prednost postignuta korištenjem razvrstavanja u

odnosu kad se ono ne koristi. Druga prednost osim smanjenja udaljenosti je i smanjenje vremena reakcije koje je potrebno za dolazak na zadatak. Budući da uslužni sustav ima manju udaljenost do mogućeg zadatka biti će mu potrebno i manje vremena da ga obavi. Zbog toga će uštedjeti na vremenu i ranije biti spreman za novi zadatak. Budući da sada djelatnik uslužnog sustava brže obavlja zadatke zbog skraćenog vremena reakcije, on može obaviti više zadataka u jednom danu, što je dodatna ušteda za uslužni sustav.

Treća prednost je u tome što razvrstavanje osim središta grozdova daje i broj mogućih zadataka u pojedinom grozdu u određenom vremenskom razdoblju. Na taj način djelatnici se mogu optimalno rasporediti po grozdovima i dobiti odgovarajući broj zadataka. Nakon svakog obavljenog zadatka, ako već nije dobio novi zadatak, djelatnik se raspoređuje u najbliži grozd u kojem je potreban i tamo čeka novi zadatak. Budući da je uslužni sustav dinamičan i stalno se mijenjaju njegovi zahtjevi, što je pokazano ranijim pokusima, razvrstavanje je potrebno obaviti nakon pristizanja novih podataka. Kako bi se razvrstavanje što prije obavilo u ovoj disertaciji su razvijeni postupci koji omogućuju paralelne procese razvrstavanja i djelotvornije predviđanje stanja uslužnog sustava.

Svim navedenim poboljšanjima uslužni sustav može znatno smanjiti troškove održavanja, učinkovitije rasporediti djelatnike, brže obavljati zadatke, a djelatnici mogu obaviti više zadataka u jednom danu nego što je to bilo bez korištenja razvrstavanja. Kao posljedica svega uslužni sustav može smanjiti cijene svojih usluga i privući veći broj korisnika. U daljnjem razvoju primijenjene postupke treba pratiti u praktičnoj primjeni, vidjeti dobivena poboljšanja u odnosu na postojeće stanje, ukloniti moguće nedostatke ako ih ima, i napraviti poboljšanja proizišla iz praktične primjene.

7. Zaključak

U ovoj disertaciji predložena su dva nova postupka za razvrstavanje nesigurnih objekata, koji su izvorni znanstveni doprinosi. U procesu razvrstavanja nesigurnih objekata najzahtjevnija računalna operacija je računanje očekivane udaljenosti. Pri računanju očekivane udaljenosti mora se računati udaljenost od svakog uzorka, kojima je predstavljena funkcije gustoće vjerojatnosti, do središta svakog grozda. To znatno usporava proces razvrstavanja nesigurnih objekata. Za izbjegavanje računanja očekivane udaljenosti razvijeni su algoritmi koji odbacuju grozdove kao kandidate za pojedini objekt bez računanja očekivane udaljenosti. Svi postojeći algoritmi navedeni u literaturi imaju određene nedostatke koji u pojedinim situacijama nemaju zadovoljavajuća svojstva. Također, nigdje u literaturi nije pronađen postupak koji omogućava razvrstavanje nesigurnih objekata pomoću paralelnih procesa.

Stoga su ovoj disertaciji predstavljena dva nova postupka. Prvi postupak omogućuje razvrstavanje prostornih podataka zasnovano na simetralnoj podjeli prostora, te usporedbi i odbacivanju grozdova. On znatno smanjuje broj računanja očekivanih udaljenosti [1] u odnosu na postupke temeljene na algoritmu MinMax, te ubrzava proces odbacivanja grozdova u odnosu na postupke temeljene na Voronojevim dijagramima. Rezultat je brži proces razvrstavanja u čitavom rasponu svih parametara bitnih za razvrstavanje. Razvrstavanje zasnovano na simetralnoj podjeli prostora između dvaju grozdova odbacuje jedan od grozdova kao kandidata za promatrani nesigurni objekt. Pomoću simetrala dvodimenzionalni prostor je podijeljen na dvije ravnine, a u svakoj ravnini nalazi se jedan grozd. Postupak koristi projekcije MBR-a nesigurnog objekta i grozdova na simetralu koja dijeli prostor na dvije ravnine. Uspoređuju se vrijednosti koordinata projekcija na simetralu i vrijednosti originalnih koordinata, i na osnovi usporedbe odbacuje se jedan od grozdova.

Prednost postupka zasnovanog na simetralnoj podjeli prostora je u tome što se odbačeni grozd ne uspoređuje s preostalim grozdovima i time se smanjuje složenost algoritma. Daljnju usporedbu s preostalim grozdovima nastavlja samo onaj grozd koji se nalazi sa iste strane simetrale kao promatrani objekt. Pokusima je dokazano da predloženi postupak učinkovitije odbacuje grozdove nego postojeći postupci u čitavom opsegu svih parametara. Provedeni su pokusi u kojima se mijenja jedan od parametara dok osnovni parametri zadržavaju osnovne

vrijednosti iz tablice 5.1. Predloženi postupak temeljen na simetralnoj podjeli prostora, nazvan SPP algoritam, imao je najbolje rezultate u svim pokusima.

U ovoj disertaciji predstavljen je i drugi postupak. On dijeli područja skupa objekata određivanjem prostornih odnosa objekata s ciljem povećanja mogućnosti paralelne obrade [2]. Postupak je skraćeno nazvan SDSA. Objašnjeno je na koji način se dijeli ukupno područje skupa objekata i koji uvjeti moraju biti zadovoljeni kako bi se osigurala ispravna podjela područja i razvrstavanje. Podjela na konačan broj manjih pravokutnih područja ovisna je o broju i rasporedu objekata i grozdova u prostoru. To je jedino ograničenje u broju pravokutnika na koje se dijeli ukupno područje skupa objekata. Svako pravokutno područje je nezavisno i razvrstavaju se samo objekti unutar područja. Pokazano je da nije nužno promatrati sve grozdove nego samo grozdove iz tog i susjednih područja, jer je sigurno da grozdovi iz vanjskih pravokutnih područja ne mogu biti bliži promatranim objektima. Svi grozdovi koji se nalaze izvan promatranih područja automatski su odbačeni za sve objekte unutar promatranog područja bez računanja očekivane udaljenosti. Postupak je kombiniran s postojećim metodama za razvrstavanje tako da se one primjene nad objektima u promatranom pravokutnom području. Pokusima je pokazano kako se kombinacijom sa SDSA algoritmom dobiju bolji rezultati nego kad se koriste samo osnovni algoritmi. Dobiveni rezultati su očekivani, jer se podjelom na samome početku odbace grozdovi iz vanjskih područja bez usporedbe s objektima.

Pravokutna područja međusobno su nezavisna i pokazane su mogućnosti za paralelno razvrstavanje nesigurnih objekata. Svako pravokutno područje može biti razvrstano kao poseban paralelan proces, i izvoditi se pojedinačno na različitim jezgrama računala, ili čak na različitom računalu. Paralelnim izvođenjem dobiju se znatno bolji rezultati, jer se razvrstavanje istovremeno obavlja na više jezgri računala. U ovoj disertaciji pokusima su dokazane prednosti paralelnog razvrstavanja na računalima s više jezgri. Pokusi su provedeni na računalu sa četiri jezgre pa je najveći broj paralelnih procesa bio četiri. Pokazano je kako se povećanjem broja jezgri smanjuje vrijeme izvođenja procesa razvrstavanja. Predloženim postupkom skraćeno je vrijeme izvođenja uz iste troškove, jer se razvrstavanje izvodi na istom računalu sa četiri jezgre kao i serijsko razvrstavanje.

Razvijeni postupci iskorišteni su za stvaranje modela predviđanja ponašanja uslužnog sustava razvrstavanjem postojećih podataka o objektima uslužnog sustava, što je treći znanstveni doprinos ove disertacije. Model je iskorišten za opisivanje trenutnog stanja

uslužnog sustava i predviđanje budućeg stanja. Za čuvanje podataka napravljena je baza podataka koja sadrži zemljopisne koordinate svih kućnih brojeva u Osijeku. Ulice su podijeljene na dijelove koji predstavljaju jedan nesigurni objekt s minimalnim područjem nesigurnosti. Nesigurni objekt je uzorkovan i svakom uzorku dodijeljena je vjerojatnost u funkciji gustoće vjerojatnosti. Broj uzoraka ovisio je o tome koliko kućnih brojeva ima u jednom nesigurnom objektu, jer je svaki kućni broj predstavljao jedan uzorak. Vjerojatnost da se objekt nalazi u tome uzorku predstavljao je broj zadataka koje je uslužni sustav obavio na tome uzorku. Napravljena je baza podataka u kojoj se nalaze vremena svih zadataka koje je uslužni sustav obavio na pojedinom uzorku. Iz baze podataka se za svaki uzorak može saznati broj i vrijeme zadataka, a iz njih se mogu izračunati vrijednosti funkcije gustoće vjerojatnosti. Na osnovu postojećih podataka iz baze podataka o pojedinom uzorku napravljena su predviđanja stanja uslužnog sustava, iz kojeg se može predvidjeti gdje će se u budućnosti ukazati potreba za uslužnim sustavom. Time se mogu znatno uštedjeti sredstva za održavanje uslužnog sustava i vrijeme izvođenja zadataka.

Razvrstavanjem podataka o uslužnom sustavu određena su središta grozdova, koja određuju mjesta na kojima je najveća koncentracija zadataka za uslužni sustav. Uslužni sustav koristi predviđanje i smješta svoje djelatnike u središtima grozdova kako bi što djelotvornije mogli djelovati prema potrebama uslužnog sustava. Budući da su potrebe za uslužnim sustavom ovisne o različitim događajima napravljeni su pokusi za različite događaje koji se mogu dogoditi i promijeniti zahtjeve za uslužnim sustavom. Za potrebe pokusa napravljena je skripta koja može prikupiti zemljopisne koordinate svih kućnih brojeva u Osijeku, jer su napravljeni pokusi za uslužne sustave u Osijeku. Napravljeni su pokusi i simulacije za uobičajeni rad uslužnog sustava, te pokusi za iznimne situacije gdje se na pojedinim mjestima okuplja veći broj građana. Pokusima je dokazano da je za poboljšanje djelovanja uslužnog sustava potrebno koristiti predviđanje budućeg stanja uslužnog sustava. Rezultati pokusa su pokazali da se u iznimnim situacijama bitno mijenjaju zahtjevi za uslužnim sustavom. Mijenja se broj zadataka u promatranom trenutku i mjesta gdje se moraju obaviti zadaci. Pomoću predviđanja uslužni sustav zna gdje će se dogoditi zadaci i na vrijeme rasporediti svoje djelatnike. Predloženim modelom smanjuju se troškovi uslužnog sustava, povećava broj zadataka koje uslužni sustav može obaviti u jedinici vremena i pouzdanost uslužnog sustava.

8. Literatura

[1] I. Lukić, M. Köhler, N. Slavek, *Improved Bisector Pruning for Uncertain Data Mining*, Proceedings of the 34th International Conference on Information Technology Interfaces, ITI, (2012).

[2] I. Lukić, M. Köhler, N. Slavek, *Segmentation of Data Set Area Method in Clustering of Uncertain Data*, Proceedings of the jubilee 35th International ICT Convention, MIPRO,(2012).

[3] A. Jain, R. Dubes, *Algorithms for Clustering Data*, Prentice Hall, New Jersey, (1998).

[4] M. Lorr, *Cluster Analysis for Social Scientists*, The Jossey-Bass Social and Behavioral Science Series. San Francisco, Washington, London: Jossey-Bass, (1983).

[5] J. Buhmann, *Data clustering and learning*, In Arbib, M., editor, The Handbook ofBrain Theory and Neural Networks, pages 308–312. Cambridge, MA: The MIT Press, (2003).

[6] A. Constantine, J. Gower, *Graphical representation of asymmetric matrices. Applied Statistics*, 27:297–304, (1978).

[7] J. Hartigan, *Representation of similarity matrices by trees*, Journal of the American Statistical Association, 62(320):1140–1158, (1967).

[8] S. Wilks, *Mathematical Statistics*. New York: John Wiley and Sons, (1962).

[9] R. Rummel, *Applied Factor Analysis*, Evanston, IL: Northwestern University Press, (1970).

[10] D. Wishart, *K-means clustering with outlier detection, mixed variables and missingvalues*, In Schwaiger, M., and Opitz, O., editors, Exploratory Data Analysis in EmpiricalResearch, pages 216–226. New York: Springer, (2002).

[11] C. Aggarwal, J. Wolf, P. Yu, C. Procopiuc, J. Park, *Fast algorithms for projected clustering*, In Proceedings of the 1999 ACM SIGMOD international conference on management of data, pages 61–72. Philadelphia: ACM Press, (1999).

[12] G. Gan, C. Ma, J. Wu, *Data ClusteringTheory, Algorithms,and Applications, Data Clustering: Theory, Algorithms, and Applications*, ASA-SIAM Series on Statistics and Applied Probability, SIAM, Philadelphia, ASA, Alexandria, VA, (2007).

[13] J. Mao, A. Jain, *A self-organizing network for hyperellipsoidal clustering (HEC)*, IEEE Transactions on Neural Networks, 7(1):16–29, (1996).

[14] A. Jain, M. Murty, P. Flynn, *Data clustering: A review*, ACM ComputingSurveys, 31(3):264–323, (1999).

[15] D. Morrison, *Measurement problems in cluster analysis*, Management Science (Series B, Managerial), 13(12):B775–B780, (1967).

[16] L. Orlóci, *An agglomerative method for classification of plant communities*, Journalof Ecology, 55:193–205, (1967).

[17] L. Legendre, P. Legendre, *Numerical Ecology*, New York: Elsevier Scientific, (1983).

[18] W. Williams, J. Lambert, *Multivariate methods in plant ecology: V. similarityanalyses and information-analysis*, Journal of Ecology, 54(2):427–445, (1966).

[19] B. Duran, P. Odell, Cluster Analysis: *A Survey, volume 100 of Lecture Notes inEconomics and Mathematical Systems*. Berlin, Heidelberg, New York: Springer-Verlag, (1974).

[20] L. McQuitty, *Elementary linkage analysis for isolating orthogonal and oblique typand typal relevancies*, Educational and Psychological Measurement, 17:207–222, (1957).

[21] F. Rohlf, *Single link clustering algorithms*, Handbook of Statistics, volume 2, pages 267–284. Amsterdam: North-Holland, (1982).

[22] G. Lance, W. Williams, *A general theory of classificatory sorting strategies I.Hierarchical systems*. The Computer Journal, 9(4):373–380, (1967).

[23] H. van Groenewoud, P. Ihm, *A cluster analysis based on graph theory*, Vegetatio, 29:115–120, (1974).

[24] B. Everitt, *Cluster analysis, 3rd edition*. New York, Toronto: Halsted Press, (1993).

[25] A. Edwards, L. Cavalli-Sforza, *A method for cluster analysis. Biometrics*, 21(2):362–375, (1965).

[26] P. Macnaughton-Smith, W. Williams, M. Dale, L. Mockett, *Dissimilarityanalysis: A new technique of hierarchical sub-division*, Nature, 202:1034–1035, (1964).

[27] H. Späth, *Cluster Analysis Algorithms*. West Sussex, UK: Ellis Horwood Limited, (1980).

[28] J. Macqueen, *Some methods for classification and analysis of multivariate observations*, In Proceedings of the 5th Berkeley symposium on mathematical statistics andprobability, volume 1, pages 281–297. Berkeley, CA: University of California Press, (1967).

[29] J. Hartigan, M. Wong, *Algorithm AS136: A k-means clustering algorithm*, Applied Statistics, 28(1):100–108, (1979).

[30] L. Bobrowski, J. Bezdek, *c-means clustering with the l_1 and l_∞ norms*, IEEETransactions on Systems, Man and Cybernetics, 21(3):545–554, (1991).

[31] Z. Huang, *Extensions to the k-means algorithm for clustering large data sets with categorical values*. Data Mining and Knowledge Discovery, 2(3):283–304, (1998).

[32] J. Pena, J. Lozano, P. Larranaga, *An empirical comparison of four initialization methods for the k-means algorithm*, Pattern Recognition Letters, 20(10):1027–1040, (1999).

[33] S. Khan, A. Ahmad, *Cluster center initialization algorithm for k-means clustering*, Pattern Recognition Letters, 25(11):1293–1302, (2004).

[34] A. Tarsitano, *A computational study of several relocation methods for k-means algorithms*, Pattern Recognition, 36(12):2955–2966, (2003).

[35] V. Faber, *Clustering and the continuous k-means algorithm*, Los Alamos Science, 22:138–144, (1994).

[36] S. Phillips, *Acceleration of k-means and related clustering algorithms*, In Mount, ALENEX: International workshop on algorithm engineering and experimentation, LNCS, volume 2409, pages 166–177. San Francicsco: Springer-Verlag, (2002).

[37] D. Pelleg, A. Moore, *x-means: Extending k-means with efficient estimation of the number of clusters*, In Proceedings of the seventeenth international conference on machine learning, pages 727–734. San Francisco: Morgan Kaufmann, (2000).

[38] C. Cheng, A. Fu, Y. Zhang, *Entropy-based subspace clustering for mining numerical data*, In Proceedings of the fifth ACM SIGKDD international conference on knowledge discovery and data mining, pages 84–93. New York: ACM Press, (1999).

[39] L. Kaufman, P. Rousseeuw, *Finding Groups in Data—An Introduction to Cluster Analysis*, Wiley Series in Probability and Mathematical Statistics. New York: John Wiley& Sons, Inc, (1990).

[40] M. Ester, H. Kriegel, J. Sander, X. Xu, *A density-based algorithm for discov-ering clusters in large spatial databases with noise*, In Second international conferenceon knowledge discovery and data mining, pages 226–231. Portland, OR: AAAI Press, (1996).

[41] M. Dash, H. Liu, X. Xu, *'1+1>2': merging distance and density based clustering*, In Proceedings of the seventh international conference on database systems for advanced applications, pages 32–39. Hong Kong: IEEE Computer Society, (2001).

[42] P. Grabusts, A. Borisov, *Using grid-clustering methods in data classification*, In Proceedings of the International Conference on Parallel Computing in Electrical Engineering, 2002. PARELEC ’02. pages 425–426. Warsaw, Poland: IEEE ComputerSociety, (2002).

[43] L. Zadeh, *Fuzzy sets*, Information and Control, 8:338–353, (1965).

[44] J. Bezdek, *Fuzzy Mathematics in Pattern Classification*, PhD thesis, CornellUniversity, Ithaca, NY, (1974).

[45] L. Xiao, E. Hung, *An Efficient Distance Calculation Method for Uncertain Objects*, Computational Intelligence and Data Mining, CIDM 2007., p. 10–17, (2007).

[46] M. Chau, R. Cheng, B. Kao, J. Ng, *Uncertain data mining: An example in clustering location data*, In PAKDD Singapore 2006, p. 199–204, (2006).

[47] N.D. Nilesh , D. Suciu, *Efficient query evaluation on probabilistic databases*, In Proc. of VLDB Conference 2004, p. 864–875, (2004).

[48] R. Cheng, D. Kalashnikov, S. Prabhakar, *Querying imprecise data in moving object environments*, IEEE TKDE 2004, 16(9):1112–1127, (2004).

[49] O. Wolfson, P. Sistla, S. Chamberlain, Y. Yesha, *Updating and querying databases that track mobile units*, Distributed and Parallel Databases 1999, 7(3), (1999).

[50] M. Ester, H. P. Kriegel, J. Sander, X. Xu, *A density based algorithm for discovering clusters in large spatial databases with noise*, In Proc. of ACM SIGKDD Conference (1996).

[51] H. P. Kriegel, M. Pfeifle, *Density-based clustering of uncertain data*, In KDD 2005, p. 672–677, (2005).

[52] H. P. Kriegel, M. Pfeifle, *Hierarchical density-based clustering of uncertain data*, In Proc. of IEEE ICDM Conference, (2005).

[53] M. Ichino, H. Yaguchi, *Generalized minkowski metrics for mixed feature type data analysis*, IEEE TSMC 1994, 24(4):698V–708, (1994).

[54] L. Xiao, E. Hung, *An Efficient Distance Calculation Method for Uncertain Objects*, Computational Intelligence and Data Mining, CIDM 2007., p. 10–17, (2007).

[55] F. K. H. A. Dehne, H. Noltemeier, *Voronoi trees and clustering problems*, Inf. Syst. 1987, 12(2):171–175, (1987).

[56] B. Kao, S. D. Lee, D. W. Cheung, W. S. Ho, K. F. Chan, *Clustering Uncertain Data using Voronoi Diagrams*, Data Mining, 2008. ICDM '08. Eighth IEEE International Conference 2008, p. 333 – 342, (2008).

[57] B. Kao, S. D. Lee, F.K.F. Lee, D. W. W. S. H. Cheung, *Clustering Uncertain Data using Voronoi Diagrams and R-Tree Index*, Knowledge and Data Engineering, IEEE Transactions, Sept. 2010. On pages: 1219-1233, (2010).

[58] W. K. Ngai, B. Kao, C. K. Chui, R. Cheng, *Efficient clustering of uncertain data*, In ICDM 2006, p. 436–445, (2006).

[59] M. Chau, R. Cheng, B. Kao, *Uncertain Data Mining: A New Research Direction*, InProc. Workshop on the Sciences of the Artificial, Hualien, Taiwan (2005).

[60] R. Cheng, X. Xia, S. Prabhakar, R. Shah, J. Vitter, *Efficient indexing methods for probabilistic threshold queries over uncertain data*, In Proc. of VLDB Conference , (2004).

[61] M. Nanni, *Speeding-up hierarchical agglomerative clustering in presence of expensive metrics*, In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages378–387, (2005).

[62] C. Elkan, *Using the triangle inequality to accelerate k-means*, In Proc. of ICML Conference, (2003).

[63] Y. Manolopoulos , A . Nanopoulos , A . N . Papadopoulos , Y. T heodoridis , *R - Trees : Theory and Applications* . Springer, ISBN 1-85233-977-2, (2005).

[64] B. R. Bowring, *Transformation from spatial to geographical coordinates*, Survey Review XXIII, 181, (1976).

Sažetak

U ovoj disertaciji predložena su dva nova postupka za razvrstavanje nesigurnih objekata. Prvi postupak omogućuje razvrstavanje prostornih podataka zasnovano na simetralnoj podjeli prostora, te usporedbi i odbacivanju grozdova. Postupak znatno smanjuje broj računanja očekivanih udaljenosti i ubrzava proces odbacivanja grozdova u odnosu na postojeće postupke. Drugi postupak dijeli područja skupa objekata određivanjem prostornih odnosa objekata s ciljem povećanja mogućnosti paralelne obrade. Postupkom je omogućeno paralelno izvođenje procesa razvrstavanja. Postupak ne zahtijeva dodatna ulaganja u opremu, jer se može izvoditi na računalu s više jezgri. Razvijeni postupci iskorišteni su za stvaranje modela predviđanja ponašanja uslužnog sustava razvrstavanjem postojećih podataka o objektima uslužnog sustava. Pomoću modela mogu se predvidjeti zahtjevi za uslužnim sustavom i djelovati prema zahtjevima. Postiže se smanjenje troškova uslužnog sustava i povećava broj zadataka koje uslužni sustav može obaviti.

Ključne riječi: *nesigurni objekti, očekivana udaljenost, paralelno procesiranje, razvrstavanje, uslužni sustavi*.

Abstract

Two original procedures for clustering spatially uncertain data are proposed in this dissertation. The first procedure enables the clustering of spatial data based on bisector division of space, using comparison and cluster pruning. It significantly reduces the number of the expected distances calculations and speeds up the process of clusters pruning in comparison to existing procedures. Second procedure divides the data set area using spatial relations among objects to increase the possibility of parallel processing. The procedure enables parallel execution of the clustering process. The procedure does not require additional investments in equipment, because of use a computer with multiple cores. Presented procedures are used for creating a model for prediction of behaviour service-oriented system, using clustering of existing data about objects of service-oriented system. This model can predict the requirements for service-oriented system and prepare according to the requirements. Costs are reduced and increased the number of tasks that can be done by service-oriented system.

Keywords: *clustering, expected distance, parallel processing, service-oriented systems, uncertain objects*.

Životopis

Ivica Lukić rođen je 10.12.1981. godine u Konjicu. Nakon završetka osnovne škole 1997. godine u Đakovu upisuje se u opću gimnaziju A.G. Matoša gdje maturira s odličnim uspjehom. Tijekom srednje škole sudjeluje na natjecanjima iz matematike, fizike, hrvatskog i povijesti i osvaja prva mjesta u općini Đakovo, a na županijskim natjecanjima osvaja 2. i 3. mjesto iz povijesti i hrvatskog. Nakon završetka gimnazije 2001. godine upisuje se na Elektrotehnički fakultet u Osijeku gdje i diplomira 2006. godine. Po završetku studija zapošljava se u tvrtki Inel d.o.o. gdje radi nešto manje od godinu dana. Nakon toga zapošljava se kao asistent na Elektrotehničkom fakultetu u Osijeku, te upisuje poslijediplomski doktorski studij, na kojem mu je mentor doc. dr. sc. Ninoslav Slavek. Uže područje interesa i istraživanja su mu razvrstavanje objekata koji sadrže nesigurnost, podjelom područja interesa na dijelove i izvođenje na paralelnim sustavima. Drugo područje interesa je kvaliteta programske podrške gdje sudjeluje u istraživanju zajedno s mentorom doc.dr.sc. Ninoslavom Slavekom. Od stručnog djelovanja potrebno je izdvojiti izradu informacijskog sustava i display-a za Veleučilište Lavoslava Ružičke u Vukovaru te izradu informacijskog sustava za Sveučilište Josipa Jurja Strossmayera u Osijeku.