Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Dejan Grbec

GRUČENJE S TOPOLOŠKIMI OMEJITVAMI

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: doc. dr. Janez Demšar

Ljubljana 2012

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00266/2012
Datum: 11.04.2012

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **DEJAN GRBEC**

Naslov: **GRUČENJE S TOPOLOŠKIMI OMEJITVAMI**
CLUSTERING WITH TOPOLOGICAL CONSTRAINTS

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Gručenje (clustering) je ena od osnovnih tehnik v odkrivanju znanj iz podatkov (data mining). V osnovnih različicah temelji zgolj na razdaljah med točkami oz. njihovem atributnem opisu, zanemarja pa znane relacije med njimi. Primer takšnega problema je gručenje držav glede na njihove značilnosti, ki ne upošteva fizičnega razporeda držav.

Razvijte, implementirajte in preskusite algoritem za hierarhično gručenje, ki bo omejen tako, da bodo objekti v vsaki gruči sosedni v tem smislu, da bodo predstavljali sklenjen podgraf v Delaunayevi triangulaciji.

Mentor:

prof. dr. Janez Demšar

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Dejan Grbec, z vpisno številko **63080213**, sem avtor diplomskega dela z naslovom:

Gručenje s topološkimi omejitvami

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Janeza Demšarja,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 14. september 2012 Podpis avtorja:

ZAHVALA

Zahvaljujem se vsem, ki so mi na kakršen koli način pomagali pri izdelavi diplomske naloge. Še posebej bi se pa zahvalil mentorju doc. dr. Janezu Demšarju za pomoč in podporo.

Dejan Grbec, Ljubljana, september 2012.

Kazalo vsebine

Kazalo slik

Povzetek

Pri analizi podatkov imamo pogosto opraviti z objekti, ki se nahajajo nekje v prostoru. Podatki so lahko vezani na mesta, države ali druge prostorske objekte, ki imajo znane prostorske koordinate. Te podatke gručimo (angl. *clustering*) po izbranih atributih, pri tem pa moramo upoštevati tudi njihovo sosednost v prostoru. Cilj diplomske naloge je bilo razviti postopke, ki pri gručenju podatkov upoštevajo sosednost objektov v prostoru. Za določanje sosednosti objektov v prostoru smo uporabili Voronojev diagram (angl. *Voronoi diagram*), s katerim pri gručenju podatkov zahtevamo, da predstavljajo elementi gruč povezane dele diagrama.

Pri tem smo razvili aplikacijo, ki omogoča hierarhično gručenje podatkov z različnimi vrstami povezav (angl. *linkage type*) in različnimi tipi razdalj (angl. *distance metrics*). Aplikacija omogoča grafični prikaz Voronojevega diagrama, pri čemer so elementi diagrama obarvani v barve gruč, kar nam omogoča boljšo vizualno predstavo podobnosti/različnosti elementov med seboj. Pri gručenju podatkov izračunavamo tudi ocene delitve s pomočjo validacijske metode Silhueta (angl. *Silhouette*), ki nam pomaga izbrati število gruč podatkov, kjer je delitev najbolj optimalna.

Ključne besede: Hierarhično gručenje, topologija, podatki v prostoru, Silhueta.

Abstract

In data analysis we often have to deal with object located somewhere in space. Data can be bound to cities, countries or other space object that have known spatial coordinates. This object can be clustered by selected attributes considering their adjacency. The object of diploma thesis was to develop procedures that consider objects neighboring relation. To determine the adjacency of object in space, we used the Voronoi diagram where the clusters of data should preset related parts of diagram.

We have developed the application that provides hierarchical data clustering with different linkage types and different distance metrics. Application provides a graphical representation of Voronoi diagram, where the diagram elements are colored in cluster colors which enables us a better visual similarity presentation of items among themselves. During clustering we evaluate data division with validation method Silhouette, which give us the number of clusters where division is optimal.

Key words: Hierarhical clustering, topology, space bounded data, Silhouette.

Poglavje 1

Uvod

Gručenje podatkov velja za eno najpomembnejših metod podatkovnega rudarjenja (angl. *data mining*) ter je običajna praksa za statistično analizo podatkov, ki se uporablja na številnih področjih [6], kot so: strojno učenje, razpoznavanje vzorcev, analiza slik, priklic informacij, bioinformatika itn.

Glavna naloga gručenja podatkov je združevanje objektov v skupine, pri čemer si morajo biti elementi v gruči med seboj čimbolj podobni ter čimbolj različni od elementov iz drugih gruč. Povezovanje na osnovi grozdenja temelji na oddaljenosti elementov, kar pomeni, da so si bližnji elementi bolj podobni. V tem primeru ne gledamo na oddaljenost objektov v prostoru, pač pa oddaljenost točk v poljubnem številu dimenzij, kjer koordinate predstavljajo standardizirane vrednosti atributov. Pri hierarhičnem gručenju poznamo celo družino metod, ki se razlikujejo po načinu izračuna razdalje. Poleg izbire tipa razdalje, se mora uporabnik odločiti tudi za tip povezave. Pri postopku gručenja vsebujejo gruče podatkov več elementov, zato obstaja več načinov izračuna razdalje med grozdi.

Cilj diplomske naloge je bil razvoj algoritma gručenja podatkov s posebno omejitvijo. Ta omejitev je topološke narave, zato od tod obstaja zahteva, da si morajo biti elementi v gručah tudi prostorsko sosednji. V nadaljevanju

bomo podrobno opisali algoritem gručenja, ki smo ga priredili in implementirali v naši aplikaciji.

Danes obstaja veliko število algoritmov združevanja podatkov v gruče, ki jih lahko uporabimo za reševanje določenega primera. Vendar vsi algoritmi niso primerni za vsak tip podatkov, zato je dobro poznati njihove razlike in lastnosti. Algoritmi gručenja se delijo v sledeče skupine [12]:

- Modeli na osnovi povezave: V to kategorijo spada hierarhično gručenje [13], kjer je podobnost med elementi definirana z razdaljo med točkami. Bližja elementa sta si tudi bolj podobna. V to skupino spada naš algoritem hierarhičnega topološkega gručenja podatkov.

- Modeli na osnovi centroida: V to kategorijo spada metoda voditeljev (angl. *K-means*). Vsaka gruča ima določen centroid, na podlagi katerih se izračuna meje Voronojevih celic, kjer vsaka celica določa pripadnost elementov določeni gruči. Algoritem deluje tako, da pričnemo z določenim številom voditeljev, ki so naključno izbrani. Nato ponavljajoče razvrščamo skupine tako, da vsak primer priredimo najbližnjemu voditelju, dokler se lega voditeljev spreminja.

- Porazdelitveni modeli: Gruče so določene s pomočjo statističnih distribucij, kot je multivariantna normalna porazdelitev, ki jo uporablja algoritem EM (angl. *expectation maximization algorithm*). Gruče lahko enostavno definiramo kot objekte, ki z največjo verjetnostjo spadajo v isto distribucijo. Lepa lastnost tega pristopa je možnost pridobivanja umetnih podatkovnih zbirk s pomočjo vzorčenja naključnih predmetov iz distribucij.

- Modeli gostote: Gruče podatkov se formirajo na mestih, kjer je gostota objektov večja od okolice. Objektom, ki so poseljeni po redkih območjih, pravimo šum in jih tretiramo kot točke za ločevanje gruč. Primer algoritma, ki spada v to skupino, je DBSCAN (angl. *Density-based spatial clustering of applications with noise*).

- Podprostorski modeli: Naloga podprostorskih modelov gručanja je odkriti vse gruče v večih podprostorih. To pomeni, da lahko en objekt, ki je član večih podprostorov, vsebovan v različnih gručah. Prostori so si lahko osno vzporedni ali afinitetni (*sorodni, podobni*).

- Modeli na osnovi skupin: Algoritmi, ki spadajo v to skupino, ponavadi niso natančni in nam zagotavljajo samo informacije o skupinah.

- Modeli na osnovi grafov [5]: Povezave med točkami grafa omogočajo medsebojno povezljivost gruč. Gruče na grafu se imenujejo skupnosti. Cilj gručenja elementov na grafih je ta, da so elementi v posamezni gruči povezani na nek vnaprej določen način.

Kot smo videli, obstaja veliko različnih algoritmom gručenja podatkov, ki jih lahko spremenimo tako, da jim postavimo neke omejitve, kar je bil cilj diplomske naloge. Odločili smo se, da bomo razvili hierarhično gručenje podatkov z omejitvami, ki spada v kategorijo modelov na osnovi povezave. Omejitve so v našem primeru vezane na topologijo objektov in zahtevamo, da so si elementi v gručah tudi prostorsko sosednji. Taki elementi morajo vsebovati prostorske koordinate, s pomočjo katerih narišemo Voronojev diagram. Vsak element prostora pridobi lastno Voronojevo celico, na podlagi katerih se določi sosednost elementov. Elementa sta sosedna, če se njuni Voronojevi celici dotikata. V splošnem nam topologija objektov predstavlja graf, ki vnaprej definira, kako se bodo elementi med seboj združevali. V nadaljevanju bomo predstavili osnovne metode hierarhičnega gručenja, delovanje algoritma, razvoj aplikcije in razliko med osnovnim in topološkim gručenjem.

Poglavje 2

Pregled osnovnih metod

V nadaljevanju bomo podrobno predstavili nekaj osnovnih metod hierarhičnega gručenja podatkov, ki smo jih implementirali v naši aplikaciji. Čeprav je algoritem za hierarhično gručenje podatkov enostaven, obstaja veliko prilagoditev, ki nam pri enakih podatkih vračajo različne rezultate. To je odvisno od tipa povezave, načina meritve razdalj ter dodatnih omejitev, ki smo jih uvedli. Te omejitve so v našem primeru vezane na topologijo objektov. Predstavili bomo tudi metodo prikaza hierarhičnega gručenja, ki se imenuje dendrogram.

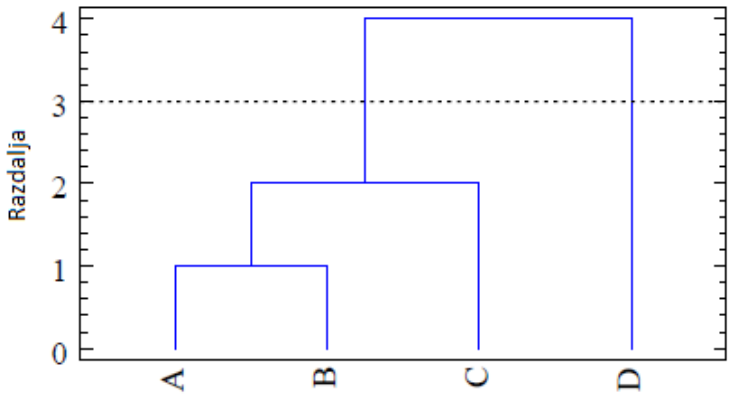
2.1 Prikaz gručenja z dendrogramom

Hierarhično gručenje se običajno vizualno prikazuje z dendrogramom [9], ki je prikazan na sliki 2.1. Vsaka združitev je predstavljena s horizontalno premico. Koordinata x predstavlja objekte, ki jih združujemo, koordinata y pa razdaljo (podobnost) med objekti. Dendrogram nam prikazuje tudi zgodovino združevanja elementov, ki so nas pripeljala do neke delitve objektov na gruče. Na sliki 2.1 vidimo, da je združevanje potekalo od spodaj navzgor v sledečem vrstnem redu:

1. A in B,

2. (A,B) in C,

3. (A,B,C) in D.



Slika 2.1: Primer dendrograma.

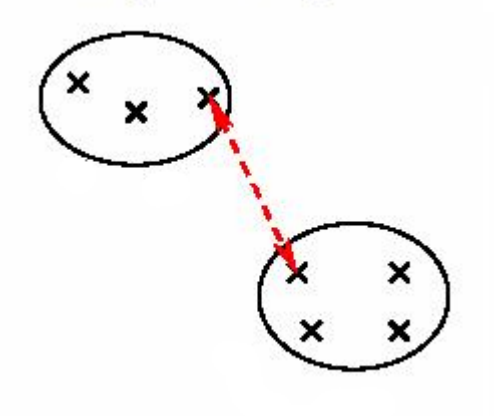
2.2 Metode združevanj

Pri hierarhičnem gručenju obstaja veliko različnih metod združevanj. Dejstvo je, da pri postopku gručenja obstajajo skupine pri katerih se moramo odločiti kako bomo primerjali podobnost gruč. V naslednjih odstavkih bomo opisali metodo najbližnjega soseda (angl. *single linkage*), metodo najbolj oddaljenega soseda (angl. *complete linkage*), povprečno metodo (angl. *average linkage*) in Wardovo metodo.

Metoda najbližjega soseda - single linkage: Pri metodi najbližjega soseda razdaljo med dvema gručama definiramo kot razdaljo med najbližjima elementoma v posameznih gručah. Razdaljo med gručama lahko zapišemo s sledečim izrazom: [7]

$$D(X, Y) = min_{x \in X, y \in Y}\ d(x, y) \tag{2.1}$$

kjer sta X in Y gruči podatkov in je d(x,y) razdalja med elementoma x in y. Na sliki 2.2 lahko vidimo grafični prikaz razdalje najbližnjega soseda.



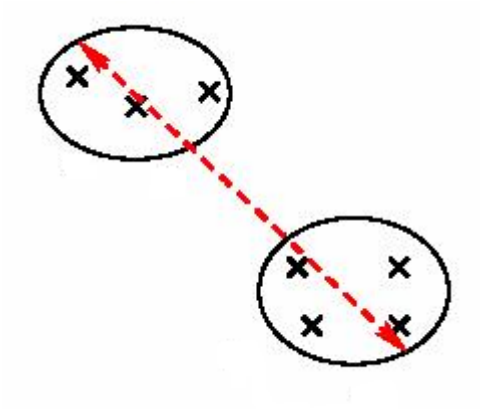
Slika 2.2: Ponazoritev metode najbližjega soseda.

Slaba lastnost metode najbližjega soseda je tako imenovano veriženje. V tem primeru so se gruče prisiljene združevati zaradi bližnjih elementov, čeprav lahko vsaka gruča vsebuje veliko elementov, ki so si precej oddaljeni.

Metoda najbolj oddaljenega soseda - complete linkage:
Pri metodi najbolj oddaljenega soseda razdaljo med dvema gručama določata najbolj oddaljena elementa iz teh dveh gruč. Razdaljo med gručama lahko zapišemo s sledečim izrazom: [7]

$$D(X, Y) = max_{x \in X, y \in Y}\ d(x, y) \tag{2.2}$$

Na sliki 2.3 lahko vidimo grafični prikaz razdalje najbolj oddaljenega soseda.



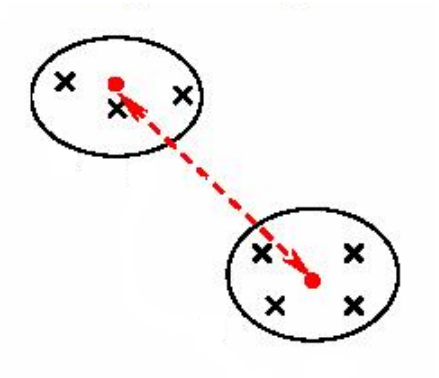
Slika 2.3: Ponazoritev metode najbolj oddaljenega soseda.

Težava metode najbolj oddaljenega soseda je ravno obratna metodi najbližjega soseda. Lahko se zgodi, da ne bomo združili dveh zelo bližnjih gruč, ker vsaka vsebuje elemente, ki so si zelo oddaljeni.

Povprečna metoda - average linkage:
Pri povprečni metodi je razdalja med dvema gručama določena s povprečjem razdalj vseh elementov iz vsake gruče. Razdaljo med gručama lahko zapišemo s sledečim izrazom: [9]

$$D(X,Y) = \frac{1}{N_X N_Y} \sum_{x \in X} \sum_{y \in Y} d(x,y) \tag{2.3}$$

Povprečna metoda predstavlja naravni kompromis med metodo najbližjega soseda in metodo najbolj oddaljenega soseda. Na sliki 2.4 je prikazana povprečna metoda.



Slika 2.4: Ponazoritev povprečne metode.

Wardova metoda - Ward's linkage:
Wardova metoda izračuna razdaljo med gručami tako, da preveri, koliko se bo ob združitvi povečala vsota kvadratov razdalj med elementi gruč. Razdalja $\Delta(X, Y)$ se imenuje strošek združitve (angl. *merging cost*), ki ga lahko zapišemo z naslednjim izrazom: [7]

$$\Delta(X, Y) = \frac{N_X N_Y}{N_X + N_Y} ||\vec{m}_X - \vec{m}_Y||^2 \tag{2.4}$$

kjer je N_X število elementov v gruči X in $\vec{m}_X$ center gruče X.

Če pogledamo enačbo 2.4, lahko vidimo, da na strošek združitve Δ vpliva tudi število elementov v posamezni gruči. Če imamo dve gruči, ki sta enako oddaljeni od tretje gruče, bo Wardov algoritem raje izbral tisto, ki vsebuje manjše število elementov. Pri hierarhičnem gručenju so sprva vsote kvadratov enake 0, ker je vsaka točka svoja gruča. Ko pričnemo združevati, se vsota kvadratov prične povečevati. Vsota kvadratov je definirana s sledečim izrazom:

$$\sum_{x \in X} ||x - \mu||^2 \tag{2.5}$$

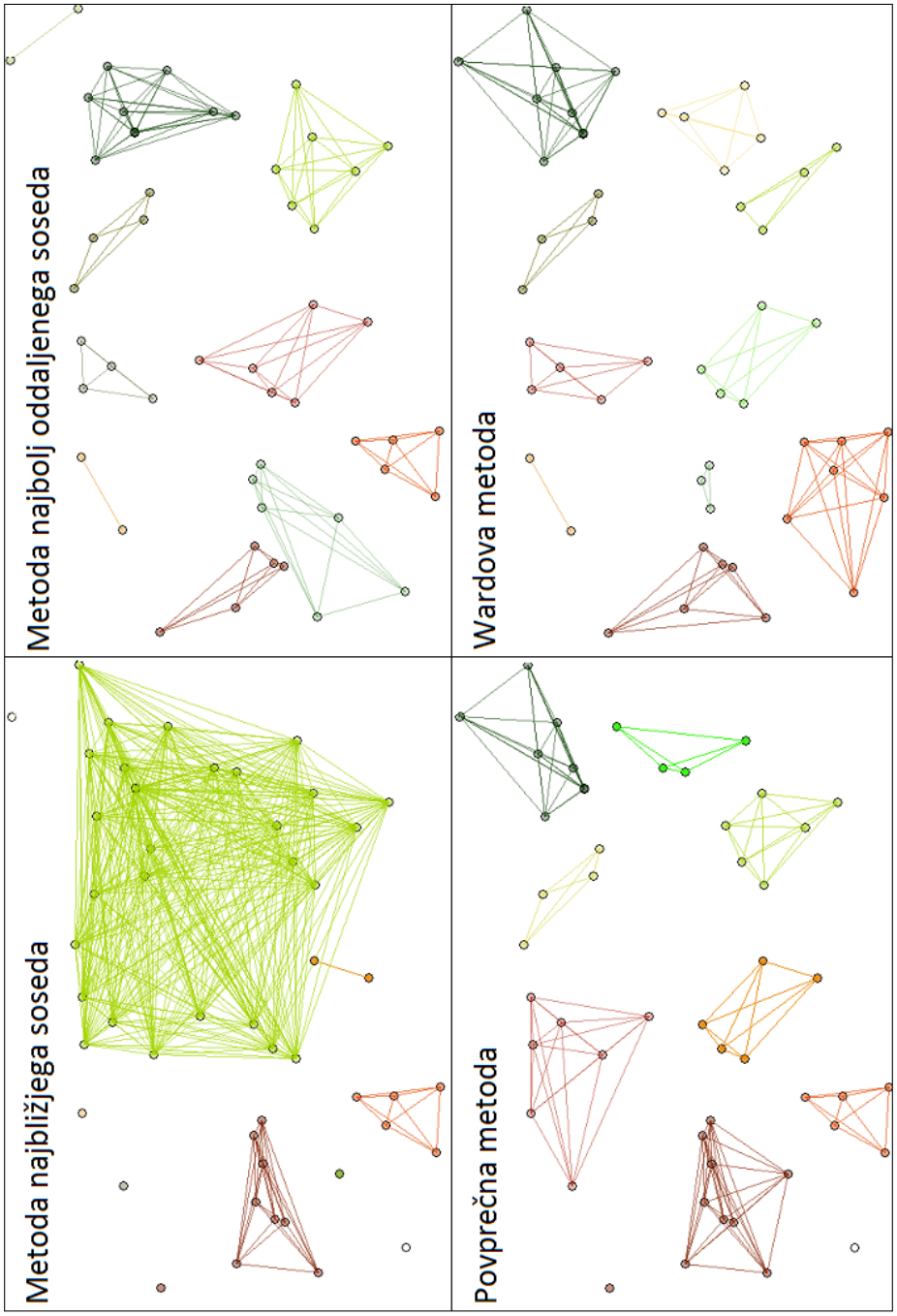
kjer je x element gruče X in μ povprečje (središče) točk v gruči X. Pri Wardovi metodi želimo, da je rast vsote kvadratov čim manjša.

Primerjava metod združevanj:
V tem podpoglavju bomo primerjali metode združevanj, ki smo jih obdelali v prejšnjih podpoglavjih. Uporaba različnih metod združevanj nas pri enakih podatkih pripelje do različnih rezultatov. To je vidno na sliki 2.5, kjer so prikazane vse štiri metode. Na vsaki sliki je 50 naključno razporejenih točk, ki so porazdeljene v 10 gruč.

Značilnost metode najbližjega soseda je ta, da se v poznih fazah združevanja na eni strani pojavlajo gruče z velikim številom elementov, na drugi pa gruče z majhnim številom elementov. To se dogaja zaradi prevlade večjih gruč, ki imajo na obrobju veliko število elementov. Zaradi tega imajo večje gruče, v primerjavi z manjšimi, večjo verjetnost, da se bodo združile.

Metoda najbolj oddaljenega soseda združuje elemente precej drugače od metode najbližjega soseda. Tukaj ne prihaja do prevlade gruč, pač pa so gruče dokaj enakomerno velike. Pri metodi najbolj oddaljenega soseda lahko pride do težave, ko sta gruči zelo blizu, vendar ju ne moremo združiti, ker vsebujejo zelo oddaljene zunanje elemente. Ta primer lahko vidimo na sliki 2.5 v spodnjem levem kotu, kjer se nahajati gruči temno rdeče in svetlo modre barve.

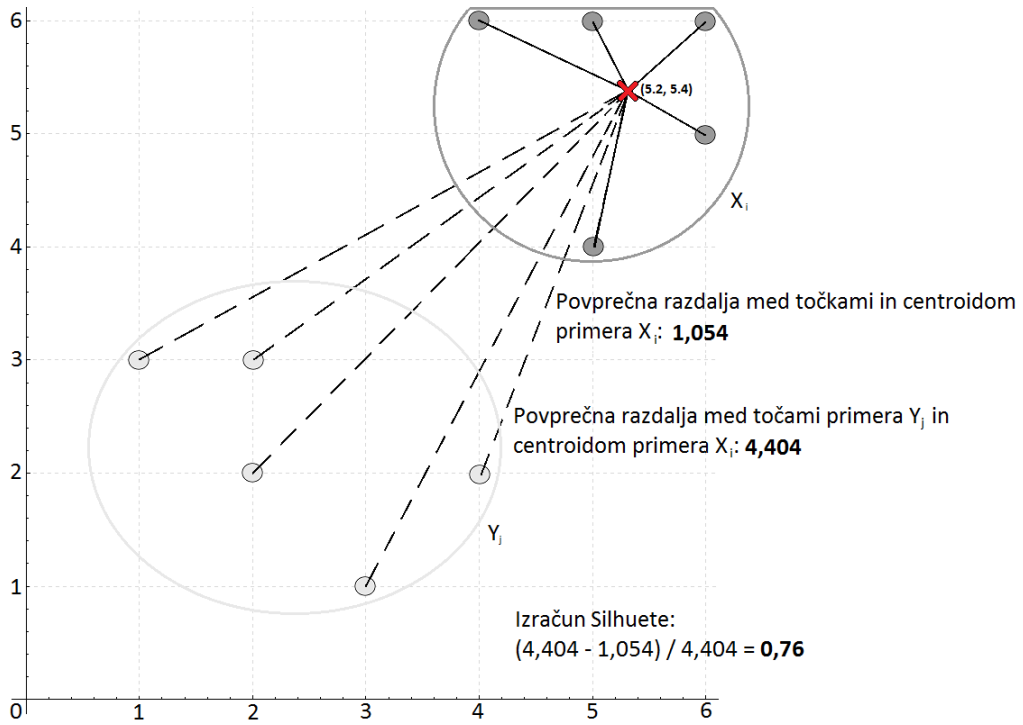
Ostaneta nam povprečna in Wardova metoda, ki sta tudi najbolj priljubljeni. Omenjeni metodi nimata podobnih težav, kot jih imata metodi najbližjega in najbolj oddaljenega soseda. Vse gruče so enakomerno oddaljene med seboj. Če primerjamo gruče na sliki 2.5 med povprečno in Wardovo metodo, lahko vidimo, da Wardova metoda ne vsebuje gruč z enim samim elementom za razliko od povprečne metode. To je tudi glavna razlika med omenjenima metodama, saj Wardova metoda pri razdalji upošteva tudi število elementov. Wardova metoda je zaradi tega bolj nagnjena k združevanju gruč z manj elementi.



Slika 2.5: Primerjava metod združevanj na 40. koraku gručenja.

2.3 Določanje števila gruč

Pri hierarhičnem gručenju podatkov dobimo n delitev elementov v gruče, kjer je n enako številu elementov. Pri takem številu delitev ne vemo, katera delitev je najbolj optimalna, zato lahko uporabimo validacijske metode. V naši aplikaciji smo implementirali validacijo gruč imenovano Silhueta (angl. *Silhouette*) [23].



Slika 2.6: Primer izračuna Silhuete na dveh primerih.

Silhueto izračunamo pri vsaki združitvi dveh najbolj podobnih gruč s pomočjo naslednje enačbe: [22]

$$s_i = \frac{b_i - a_i}{max\{a_i, b_i\}} \tag{2.6}$$

kjer je a_i povprečna razdalja primera X_i do vseh ostalih primerov v njegovi skupini in b_i povprečna razdalja med a_i in vsemi primeri iz Y_j, kjer $X_i \notin Y_j$. Iz enačbe 2.6 lahko ugotovimo, da je $-1 \leq s(i) \leq 1$. Večja številka

označuje boljšo delitev. Želimo da je a_i čim manjši, kar pomeni, da so si primeri v množici X_i zelo blizu, medtem pa mora biti b_i čim večji, kar pomeni, da je primer Y_j dovolj oddaljen od primera X_i. Izračun Silhuete na dveh primerih smo ponazorili na sliki 2.6.

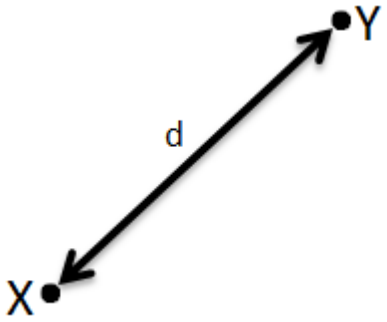
2.4 Definicije razdalj

Razdalje med gručami lahko merimo na več načinov. V naši aplikaciji smo implementirali evklidsko in manhattansko razdaljo. Poskusno smo implementirali tudi Hammingovo razdaljo, vendar smo kasneje ugotovili, da ni primerna za našo vrsto podatkov. V nadaljevanju bomo opisali uporabljene vrste meritev.

Evklidska razdalja:
Običajna razdalja v geometriji je evklidska razdalja. Izračunamo jo lahko s pomočjo naslednje enačbe: [12]

$$d(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2.7}$$

kjer je n število dimenzij ter x_i i-ta koordinata točke X. Evklidska razdalja v dvorazsežnem prostoru je prikazana na sliki 2.7.



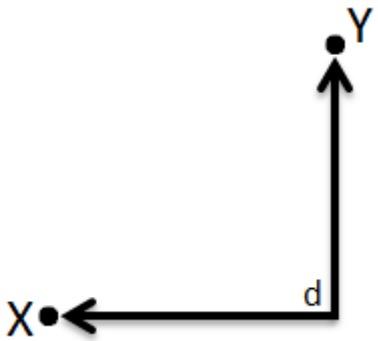
Slika 2.7: Evklidska razdalja.

Poleg evklidske razdalje, ki jo definira izraz 2.7, obstaja tudi kvadrirana evklidska razdalja. Ta razdalja uporablja enako enačbo, ki je brez kvadratnega korena. Rezultat tega je hitrejše delovanje gručenja, vendar moramo biti previdni. Pri hierarhičnem gručenju se rezultat lahko spremeni v primerjavi z običajno evklidsko razdaljo.

Manhattanska razdalja:
Manhattanska razdalja se imenuje po znanem delu mesta New Yorka. Najkrajša pot, ki jo lahko prepotujemo med dvema točkama, poteka po mreži ulic. Manhattanska razdalja je torej vsota sprememb istoležnih koordinat prostora. To lahko zapišemo z naslednjo enačbo: [12]

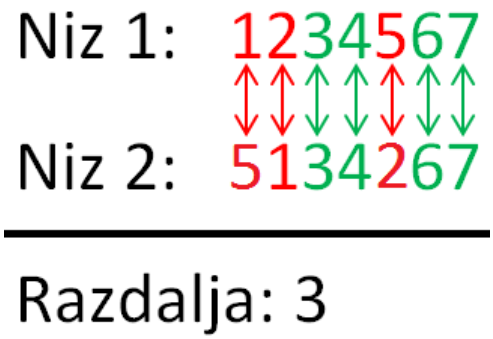
$$d(X, Y) = \sum_{i=1}^{n} |x_i - y_i| \tag{2.8}$$

Manhattanska razdalja v dvorazsežnem prostoru je prikazana na sliki 2.8.



Slika 2.8: Manhattanska razdalja.

Hammingova razdalja:
Hammingova razdalja se ne opira na razdaljo v geometričnem prostoru tako kot evklidska in manhattanska, pač pa je določena kot število razlik simbolov na istoležnih mestih dveh enako dolgih nizov. To je ponazorjeno na sliki 2.9.



Slika 2.9: Hammingova razdalja.

2.5 Metode določanja topologije objektov

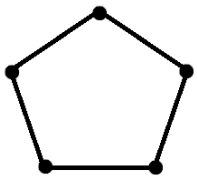
Cilj diplomske naloge je bil razvoj postopkov gručenja z upoštevanjem topologije objektov v prostoru. Algoritem hierarhičnega gručenja smo spremenili tako, da se pri združevanju gruč preverja tudi prostorsko sosednost objektov. Če si gruči nista sosednji v tem smislu, da sta soseda vsaj dva od njunih elementov, ju ne smemo združiti.

Prostorsko sosednost lahko določamo na veliko načinov. Sosednost je lahko določena kar iz samih lastnosti objektov. Objektom, kot so države sveta, lahko sosednost določimo z državno mejo, ki je splošno znana. Ta način je sicer logičen za omenjen primer, vendar ni uporaben za drugačne podatke. Želeli smo razviti aplikacijo, ki deluje generično na katerih koli podatkih, vse kar moramo poznati, so koordinate prostora. Za to smo uporabili Voronojev diagram, ki ga bomo predstavili v naslednjem odstavku.

Voronojev diagram:

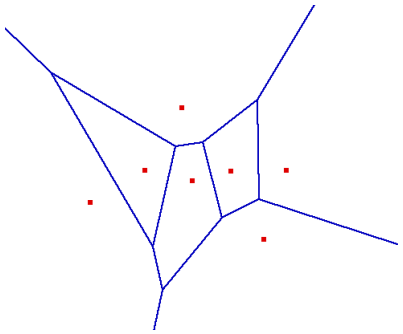
Voronojev diagram [19] je posebna vrsta dekompozicije danega prostora, ki je določena s pomočjo oddaljenosti objektov med seboj. Ti objekti se imenujejo seme (angl. *seed*). Voronojevim diagramom pravimo tudi Voronojev mozaik, Dirichletov mozaik ali Voronojeva dekompozicija. Vsaka Voronojeva celica vsebuje natanko eno seme, ki predstavlja objekt v danem prostoru.

V nadaljevanju bomo opisali generiranje Voronojevega diagrama. Imamo končni nabor točk $\{p_1, p_2, p_3, ..., p_n\}$ v evklidski ravnini. Vsaka točko p_k oklepa ustrezna Voronojeva celica, ki je konveksen poligon, kot ga predstavlja slika 2.10.



Slika 2.10: Primer konveksnega poligona.

Vsako Voronojevo celico dobimo tako, da narišemo pravokotno premico med sosednjima točkama, ki je enako oddaljena od obeh točk. Presečišča takih premic tvorijo točke poligona posamezne Voronojeve celice. Primer Voronojevega diagrama je prikazan na sliki 2.11.

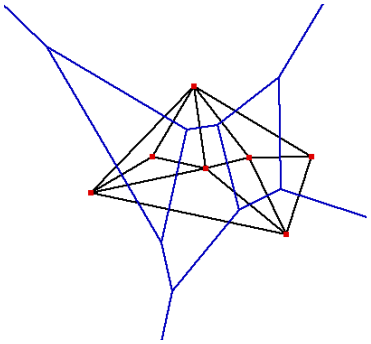


Slika 2.11: Voronojev diagram.

Za Voronojev diagram obstaja dualni graf, ki se imenuje Delaunayeva triangulacija [20] (slika 2.12). Delaunayeva triangulacija mora izpolnjevati

naslednji pogoj: če vsem trikotnikom očrtamo krog, potem znotraj nobenega kroga ne sme biti vsebovana druga točka. Med Delaunayevo triangulacijo in Voronojevim diagramom obstaja sledeča korelacija: če Delaunayeva triangulacija vsebuje povezavo med dvema poljubnima točkama, potem mora Voronojev diagram vsebovati pravokotno premico med njima, od katere sta obe točki enako oddaljeni in tvori poligon Voronojeve celice obeh točk. Velja tudi obratno: če imata poligona v Voronojevem diagramu skupno mejo, sta pripadajoči točki Delaunayeve triangulacije povezani.

Povezave med točkami v Delaunayevi triangulaciji in dotikajoče Voronojeve celice nam definirajo sosednost točk, ki smo jo potrebovali.



Slika 2.12: Delaunayeva triangulacija na vrhu Voronojevega diagrama.

Poglavje 3

Hierarhično topološko gručenje

3.1 Vrsti hierarhičnega gručenja

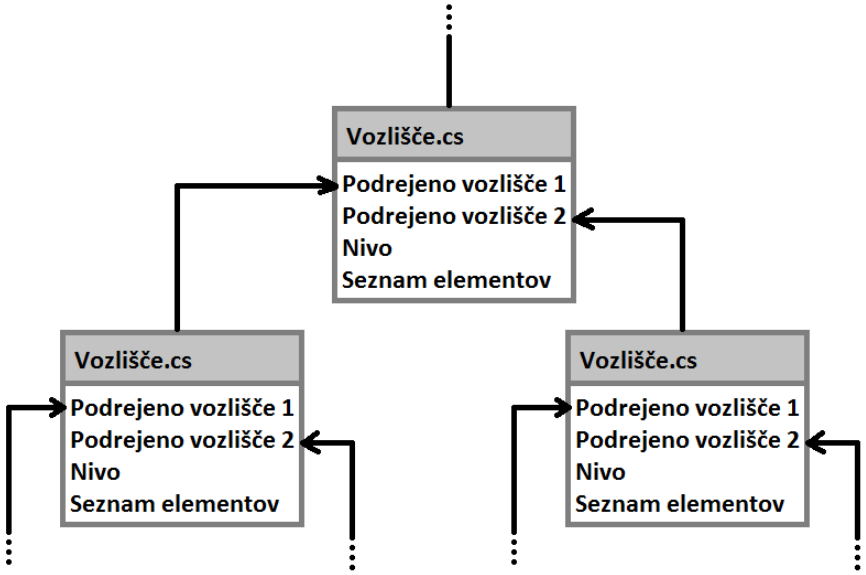
V splošnem poznamo dve vrsti hierarhičnega gručenja podatkov. Hierarhično gručenje lahko poteka od spodaj navzgor (angl. *bottom-up*) ali od zgoraj navzdol (angl. *top-down*) [1]. Prvi način obravnava vsak objekt kot samostojno gručo, ki jih v nadaljevanju po parih združuje po merilu podobnosti, dokler niso vsi elementi združeni v eno samo gručo podatkov. Tako gručenje se imenuje hierarhično gručenje na osnovi združevanja (angl. *hierarhical agglomerative clustering*). Ta način je tudi bolj priljubljen, saj je enostavnejši in časovno manj zahteven, zato smo ga tudi izbrali za razvoj algoritma gručenja podatkov z omejitvami.

Drugi način hierarhičnega gručenja poteka od zgoraj navzdol, ki pa deluje ravno obratno od prejšnjega. Pričnemo z vsemi elementi v eni gruči, ki jo rekurzivno delimo na manjše gruče po merilu različnosti, dokler elementi ne postanejo samostojne gruče. Tako gručenje se imenuje hierarhično gručenje na osnovi delitve (angl. *hierarhical divisive clustering*). Pri tem načinu gručenja je časovno najbolj zahtevno iskanje skupin najbolj različnih elementov znotraj ene gruče.

3.2 Algoritem gručenja

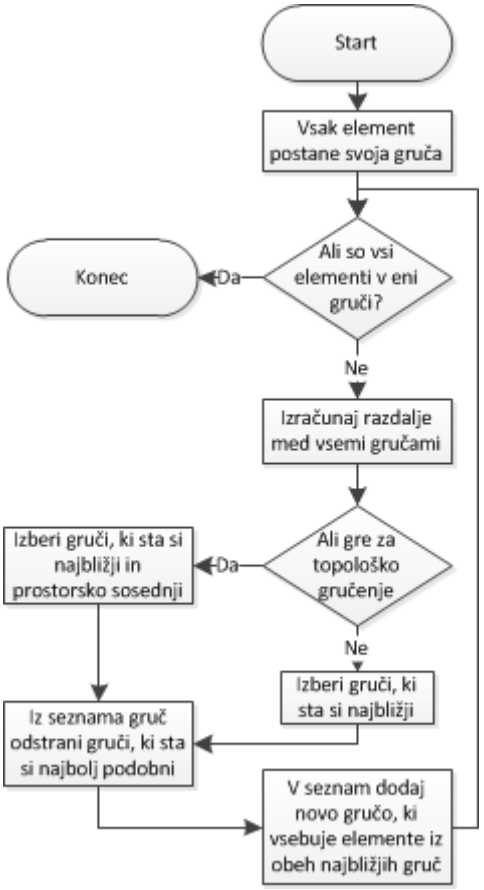
V tem poglavju bomo opisali osnovno delovanje algoritma hierarhičnega gručenja podatkov [8]. Pred pričetkom moramo podatke ustrezno pripraviti. Izbrane atribute moramo normalizirati, kar pomeni, da jih iz prvotnih vrednosti pretvorimo v vrednosti od 0 do 1. Nič predstavlja najmanjšo vrednost atributa vseh elementov, ena pa največjo vrednost. To nam zagotavlja, da imajo vsi atributi enako težo pri gručenju podatkov. Če si tega ne želimo, potem apliciramo ustrezne uteži na posamezne vrednosti atributov.

Gručenje podatkov se zaključi, ko so vsi elementi združeni v eni gruči podatkov. Zato je zelo pomembno razviti algoritem, ki deluje na podatkovni strukturi, ki nam omogoča shranjevanje celotne zgodovine gručenja. Najbolj primerna podatkovna struktura za naš primer je binarno drevo, kjer vsaka združitev predstavlja vozlišče drevesa. Vozlišče drevesa smo ponazorili z objektom Node (slov. *Vozlišče*), ki je prikazan na sliki 3.1. Vsako vozlišče vsebuje povezavi na dva podrejena vozlišča, nivo gručenja, ki ga predstavlja vozlišče in seznam elementov, ki je sestavljen iz seznama elementov obeh podrejenih seznamov.



Slika 3.1: Prikaz objekta, ki predstavlja vozlišče.

Na sliki 3.2 je prikazan diagram algoritma gručenja podatkov. S pričetkom gručenja vsak element prestavimo v objekt Vozlišče, ki predstavlja gručo z enim elementom. Algoritem se zaključi, ko končamo z enim samim objektom Vozlišče, ki vsebuje vse elemente, kateri so sprva bili v svojih gručah. Dokler ta pogoj ni izpolnjen, med vsemi gručami izračunamo razdalje in izberemo tisti dve, ki sta si najbližji. Če gručimo topološko, mora biti zadoščeno tudi pogoju prostorske sosednosti. Nato iz seznama gruč odstranimo najbližja elementa, ki ju prestavimo v nov objekt Vozlišče. Gruči postaneta podrejeni, seznam elementov pa vsebuje elemente iz obeh gruč. Ta nov objekt dodamo v seznam gruč in nadaljujemo izvajanje z enakim postopkom.



Slika 3.2: Diagram poteka gručenja podatkov.

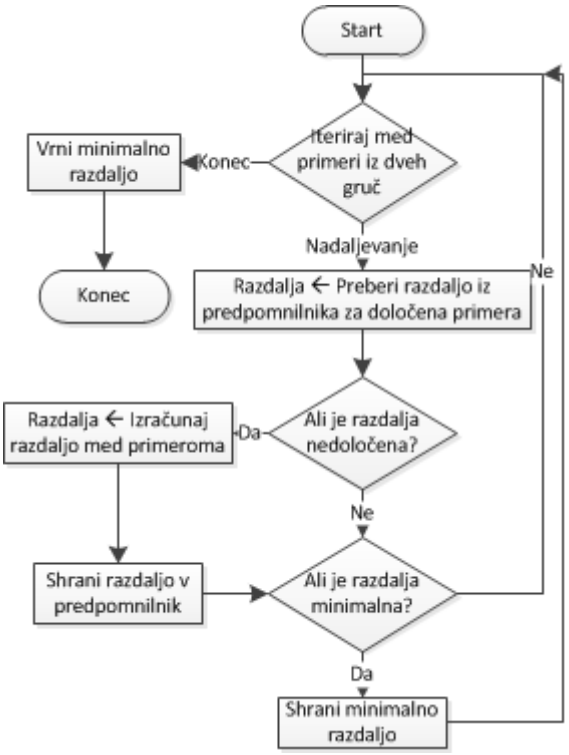
Časovna zahtevnost naivnega algoritma gručenja je $O(n^3)$, kar pomeni, da algoritem ni primeren za večje zbirke podatkov. Optimizirana metoda najbližjega soseda ima časovno zahtevnost enako $O(n\ log\ n)$, kar je ekvivalentno iskanju minimalnega vpetega drevesa. Ta časovna zahtevnost je sprejemljiva, vendar metoda ni najbolj priljubljena zaradi svojih lastnosti. Časovna zahtevnost hierarhičnega gručenja z delitvijo, ki smo ga omenili v poglavju 3.1, je $O(2^n)$, kar je še počasneje.

3.3 Optimizacije metod združevanj

Pri implementaciji metod združevanj bi predvsem predstavili optimizacijsko metodo, ki pohitri delovanje algoritma gručenja podatkov. Dejstvo je, da se pri gručenju podatkov zelo pogosto izračunavajo razdalje med primeri, ki so se izračunale že v preteklosti, zato je dobro premisliti, če bi te razdalje predpomnili v neko podatkovno strukturo, ki omogoča zelo hitro branje in vstavljanje elementov. Za ta namen smo razvili slovar z dvema ključema, ki predstavljata primera, njuna vrednost pa predstavlja razdaljo med njima. Na sliki 3.3 je prikazan diagram poteka, ki prikazuje izračun evklidske razdalje z optimizacijo.

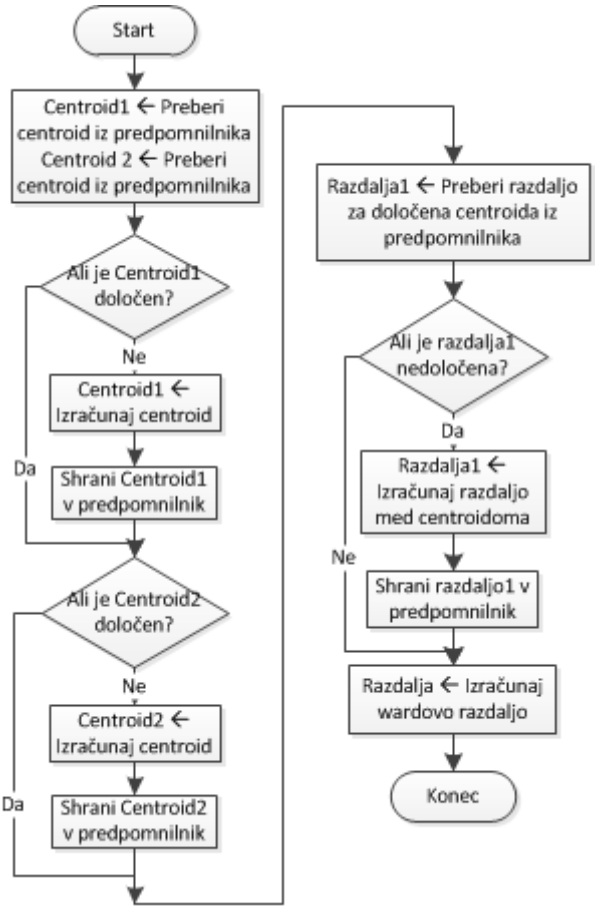
Pred izračunom preverimo ali obstaja zapis za določena primera. Če obstaja, ga uporabimo, v nasprotnem primeru pa izračunamo razdaljo med primeroma in jo shranimo v predpomnilnik. Izkazalo se je, da je pohitritev zelo občutna, saj je delovanje približno dvakrat hitrejše. Tako vrsto pohitritve smo uporabili pri metodi najbližjega soseda, metodi najbolj oddaljenega soseda in povprečni metodi.

Optimizirali smo tudi delovanje Wardove metode. Zaradi drugačnega delovanja Wardove metode je optimizacija malce drugačna od ostalih treh, kjer smo predpomnili samo razdalje med primeri. Če pogledamo



Slika 3.3: Diagram poteka izračuna evklidske razdalje z optimizacijo.

enačbo 2.4, lahko vidimo, da je v del enačbe vključena razdalja med središčema gruč (*centroid*). Zato smo se odločili za dvonivojsko predpomnenje, kjer v prvi vrsti predpomnimo centroide posameznih gruč, nato pa še razdalje med centroidi. Na sliki 3.4 je prikazan diagram poteka izračuna Wardove razdalje z optimizacijo. Optimizacija se je izkazala za zelo uspešno, saj smo pohitrili delovanje gručenja na osmih atributih za faktor 18.



Slika 3.4: Diagram poteka izračuna Wardove razdalje z optimizacijo.

Poglavje 4

Razvoj in uporaba aplikacije

V tem poglavju bomo predstavili razvoj in uporabo aplikacije, ki omogoča hierarhično topološko gručenje podatkov, njene konstrukte, uporabljene tehnologije in funkcionalnosti.

4.1 Uporabljena orodja in tehnologije

4.1.1 Visual Studio 2010

Aplikacijo smo razvili v integriranem razvojnem okolju (angl. *Integrated development enviroment - IDE*) Visual Studio 2010 [15], ki ga je razvilo podjetje Microsoft. Uporabljamo ga lahko za razvoj ukaznih aplikacij, aplikacij z uporabniškim vmesnikom, spletnih strani, spletnih aplikacij in spletnih servisov s pomočjo strojne ali nadzorovane kode, ki je podprta skozi vrsto platform. Visual Studio podpira različne programske jezike, kot so: C/C++, VB.NET, C# in F#. Podprti so tudi drugi jeziki (Python, Ruby, M,...), ki jih lahko namestimo ločeno. Visual Studio vključuje naslednja orodja in funkcionalnosti:

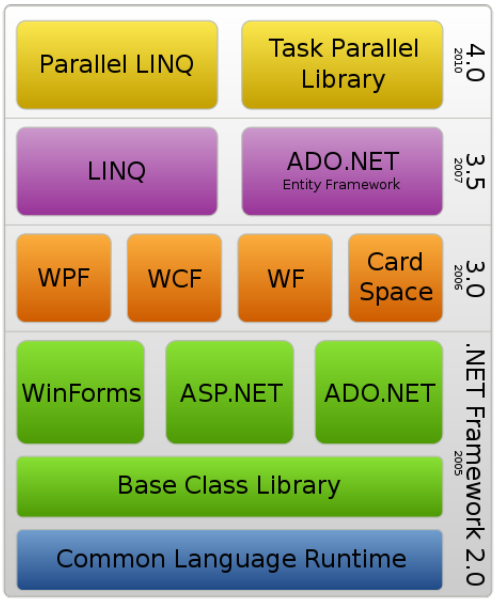
- Urejevalnik programske kode s pomočjo IntelliSensa, ki omogoča samodopolnjevanje kode,
- razhroščevalnik napak na ravni kode in strojne opreme,

- orodja za oblikovanje in izgradnjo uporabniškega vmesnika,
- orodja za razvoj spletnih aplikacij,
- orodje za načrtovanje razredov,
- orodje za razvoj shem podatkovnih baz itn.

Visual Studio omogoča razvijalcem, da si razširijo funkcionalnosti razvojnega okolja s pomočjo makrojev, vtičnikov in paketov. Makroji predstavljajo ponavljajoča opravila in akcije, ki so programirana vnaprej in omogočajo samodejno izvajanje. Z makroji ne moremo ustvariti novih ukazov ali orodij z vmesnikom. Pišemo jih v programskem jeziku Visual Basic, kjer koda ni vnaprej prevedena. Vtičniki imajo dostop do objektnega modela razvojnega okolja in komunicirajo z njegovimi orodji. Uporabljamo jih za dodajanje novih funkcionalnosti in orodij z vmesnikom. Vtičniki so povezani z razvojnim okoljem preko vmesnika COM (angl. *Component Object Model*) in jih razvijamo s katerimkoli skladnim programskim jezikom. Tretja razširitev so paketi, ki zagotavljajo največjo mero razširljivosti. Pakete razvijamo s pomočjo razvojnega kompleta Visual Studio SDK, s katerim lahko usvarimo razvijalska orodja, ter integriramo druge programske jezike.

4.1.2 Ogrodje .NET

Ogrodje .NET [16] je razvojno ogrodje razvito s strani Microsofta, ki primarno deluje na operacijskem sistemu Microsoft Windows. Vključuje veliko knjižnico z osnovnimi ukazi ter interoperabilnost med različnimi programskimi jeziki. Programi, ki so napisani z ogrodjem .NET, se izvajajo v programskem okolju CLR (angl. *Common Language Runtime*). CLR je aplikacijski navidezni stroj, ki zagotavlja varnost, upravljanje z pomnilnikom in obravnavanje izjem. Knjižnica razredov in CLR skupaj predstavljajo ogrodje .NET.



Slika 4.1: Novosti v verzijah ogrodja .NET.

4.1.3 Razširljiv označevalni jezik - XML

XML [14] je angleška okrajšava za *razširljiv označevalni jezik* (angl. *Extensible markup language*). XML opredeljuje pravila za kodiranje dokumenta v formatu, ki je berljiv tako uporabniku kot računalniku. Je zelo podoben jeziku HTML (angl. *Hyper text markup language*), vendar je bil razvit za

drugačne namene. Jezik HTML je bil razvit za prikazovanje podatkov s poudarkom na izgledu, medtem ko je naloga XML jezika prenos in shranjevanje podatkov s poudarkom na informaciji, kaj podatki pomenijo. XML je razdeljen na tri dele:

- Podatkovni (podatki znoraj oznak (angl. *tags*)),
- deklarativni (pomen oznak) in
- predstavitveni (oblikovanje izpisa podatkov).

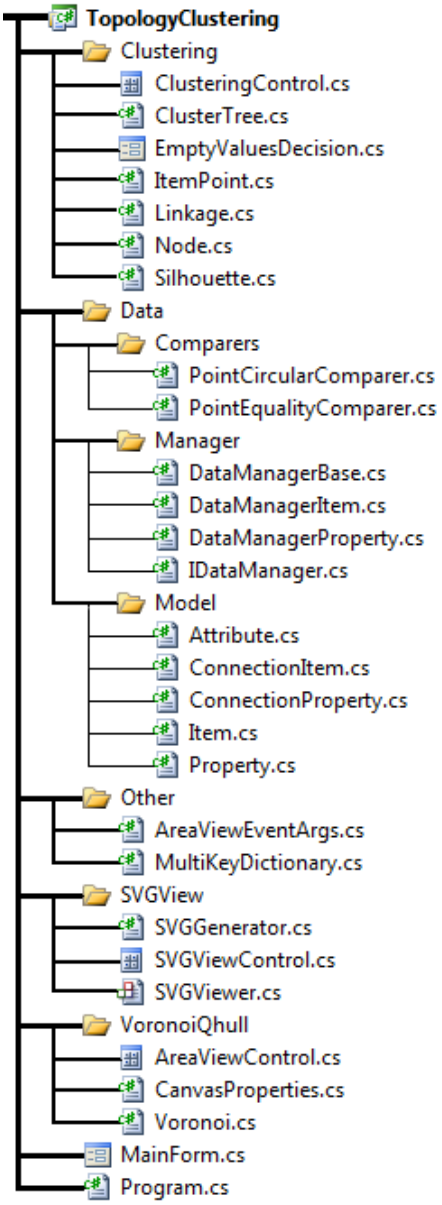
Na sliki 4.2 je primer zapisa podatkov v jeziku XML.

```
<?xml version="1.0" encoding="utf-8"?>
<items>
  <item>
    <id>1</id>
    <title>Afghanistan</title>
    <x>33 00 N</x>
    <y>65 00 E</y>
    <svgId>af</svgId>
  </item>
  <item>
    <id>2</id>
    <title>Albania</title>
    <x>41 00 N</x>
    <y>20 00 E</y>
    <svgId>al</svgId>
  </item>
  <item>
    <id>3</id>
    <title>Algeria</title>
    <x>28 00 N</x>
    <y>3 00 E</y>
    <svgId>dz</svgId>
  </item>
</items>
```

Slika 4.2: Zapis podatkov v jeziku XML.

4.2 Struktura projekta

Na sliki 4.3 je prikazana drevesna struktura projekta. Projekt se deli na 5 različnih delov, kot so: Clustering (slov. *Gručenje*), Data (slov. *Podatki*), Other (slov. *Ostalo*), SVGView in VoronoiQ-Hull.



Slika 4.3: Struktura projekta.

V mapi clustering so vključene vse funkcionalnosti v zvezi z gručenjem. To so metode združevanj, tipi meritev, izračun Silhuete, strukturni elementi, kot je razred Node (*vozlišče dendrograma*) in ItemPoint (*element z normaliziranimi atributi*), algoritem za gručenje, kot tudi grafični vmesnik za izbiro praznih vrednosti in vmesnik, ki omogoča izbiro parametrov gručenja in prikazuje seznam elementov združen v gruče.

Mapa data ima nalogo pridobivanja podatkov iz datotek tipa XML. Ko so podatki prebrani, se ohranijo v pomnilniku in ostanejo nespremenjeni na njihovem osnovnem modelu. Ko izberemo ustrezne atribute, se podatki prepišejo v drug model, za kar poskrbi funkcionalnost gručenja.

V mapi SVGView se nahaja grafični vmesnik, ki na zemljevidu SVG (angl. *Scalable vector graphics*, slov.

Skalabilna vektorska grafika) prikazuje gruče podatkov obarvane v izbrane barve. Barvanje površin je izvršeno s pomočjo stilov CSS (angl. *Cascading style sheets*), ki so uporabljeni predvsem v spletnem programiranju in s pomočjo drevesa gručenja, ki ga posreduje funkcionalnost gručenja.

Mapa VoronoiQHull je zadolžena za določanje sosednosti pri branju podatkov in risanje Voronojevega diagrama. Uporabili smo knjižnico Qhull, ki nam omogoča generiranje Voronojevih diagramov. Do izhoda knjižnice Qhull dostopamo preko ukazne vrstice sistema.

V mapi other se nahajo ostali podporni razredi, ki omogočajo pohitritev delovanja in prenašanje parametrov preko dogodkov (angl. *Events*).

4.3 Knjižnica Qhull

Za generiranje Voronojevih diagramov smo uporabili prosto dostopno knjižnico Qhull [18] (podmodul Qvoronoi). Knjižnica nam je omogočala pridobivanje določenih parametrov Voronojevega diagrama glede na vhodne točke. Težava, s katero smo se soočali pri uporabi Qvoronoi knjižnice, je ta, da nima neposredne podpore za delo z aplikacijami .NET. Na sistemih Windows deluje knjižnica v ukazni vrstici, kar nam je povzročalo nekaj težav. Težavo smo rešili z objektom Process (slo. *proces*), ki omogoča zagon ukazne vrstice z danim ukazom. Na sliki 4.4 je prikazana metoda, ki s pomočjo omenjenega objekta odpre ukazno vrstico z vhodnim ukazom in vrne njen izhod, ki ga je potrebno dekodirati. Ta metoda nam predstavlja komunikacijo med našo aplikacijo in knjižnico Qvoronoi.

Za generiranje Voronojevega diagrama smo potrebovali tri ukaze v sledečem zaporedju:

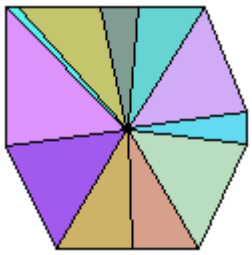
1. qvoronoi.exe p TI points.txt,
2. qvoronoi.exe FN TI points.txt in
3. qvoronoi.exe Fo TI points.txt.

```
private string GetOutput(String commandLine)
{
    Process p = new Process();
    p.StartInfo = new ProcessStartInfo("cmd", commandLine)
    {
        RedirectStandardOutput = true,
        UseShellExecute = false,
        CreateNoWindow = false
    };
    p.Start();
    string output = p.StandardOutput.ReadToEnd();
    p.WaitForExit();
    return output;
}
```

Slika 4.4: Metoda za komunikacijo med aplikacijo in knjižnico Qhull.

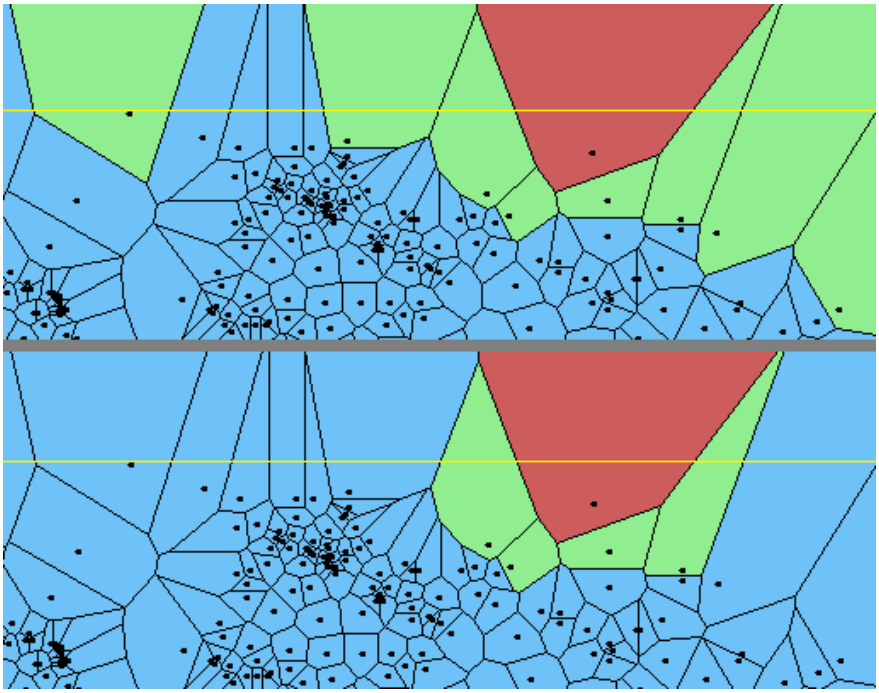
Vhodne točke pri vseh ukazih nam predstavlja datoteka *points.txt*. Prvi ukaz nam vrne seznam koordinat točk vseh poligonov Voronojevega diagrama. Seznam nam ne sporoča pripadnosti točk danemu poligonu, zato smo morali uporabili drugi ukaz. Drug ukaz nam vrne seznam točk, ki pripadajo določenemu poligonu. Pripadnost je določena s pomočjo zaporedne številke koordinate prvega ukaza. Seznam pripadnih točk je urejen po zaporedju vhodnih točk. Tretji ukaz smo uporabili za identifikacijo neomejenih področjih, ki so lastnost vsakega Voronojevega diagrama. Neomejenega področja ne moremo v celoti zapisati s točkami, ki bi tvorile poligon, zato moramo uporabiti premice z določenim koeficientom naklona. Ukaz nam vrne seznam parov točk, med katerima potuje premica, ki je določena s koeficientom naklona. Točki sta identificirani s pomočjo zaporedne številke iz vhoda. Neomejena področja smo želeli zapisati s točkami poligona, zato smo uporabili manjši trik. Odločili smo se, da manjkajoče točke neomejenih področij obstajajo na primerno veliki razdalji premic z določenim koeficientom. Tako smo poenostavili risanje voronojevega diagrama pri čemer nismo vplivali na določanje sosednosti. Na sliki 4.5 je prikazan omejen Voronojev diagram.

Ko smo vsakemu elementu določili točke poligonov, ki tvorijo Voronojev diagram, nam ni bilo težko določiti sosednosti med elementi. Če si elementa delita več kot eno točko poligona, potem lahko predpostavimo njuno sose-



Slika 4.5: Omejen Voronojev diagram.

dnost. Ta pogoj žal ni zadosten, saj nam sosednost pokvarijo elementi, ki so določeni z neomejenimi področji. Premice teh elementov se lahko združijo na veliki oddaljenost, kar lahko pripelje do tega, da elementi postanejo sosednji, čeprav so si med seboj preveč oddaljeni. To situacijo lahko vidimo na sliki 4.6. Zelena področja so sosede rdečega elementa.



Slika 4.6: Težava pri določanju sosednosti elementov z neomejenimi področji (zgornja slika).

Težavo smo rešili z dodanim pogojem, ki prepreči določanje soseda, če si elementa delita več kot eno točko poligona preko rumene premice, ki je narisana na sliki 4.6. Ta premica je del pravokotnika, ki omejuje vse točke Voronojevega diagrama in je določen z najbolj obrobnimi točkami Voronojevega diagrama. Na spodnjem delu slike 4.6 so prikazane sosede rdečega elementa, kjer upoštevamo dodaten pogoj, ki smo ga opisali. Tako delovanje je izpolnilo naša pričakovanja.

4.4 Implementacija Silhuete

V tem poglavju bomo opisali našo implementacijo validacijske metode Silhueta. To metodo se največ uporablja pri gručenju z metodo voditeljev, uporabimo jo pa lahko tudi pri hierarhičnem gručenju. Pri gručenju N elementu dobimo skozi celo hierarhijo združevanj N različnih kombinacij združitev gruč, zato smo se odločili, da Silhueto izračunamo na vsakem koraku gručenja. Taka metoda je bila časovno najbolj optimalna, saj se nam ni bilo potrebno ponovno sprehajati po drevesni strukturi gručenja. Informacijo o vrednosti Silhuete hrani objekt Vozlišče (angl. *Node*). Na sliki 4.7 je prikazan diagram podatka izračuna Silhuete med postopkom gručenja.

Slika 4.7: Izračun Silhuete med postopkom gručenja.

Če govorimo o najboljši oceni Silhuete, potem gre za povprečje ocen vseh združitev na določenem nivoju združevanj v drevesu (Dendrogamu). Pri hierarhičnem gručenju podatkov moramo biti pozorni na rezultat izračuna Silhuete. Vsako hierarhično gručenje podatkov se prične

z združevanjem posameznih elementov v nove gruče. Osredotočimo se na enačbo 2.6. Če združujemo primer X_i, ki vsebuje en sam primer, iz tega sledi $b_i = 0$. V tem primeru bo rezultat Silhuete enak 1, kar morda ni realen rezultat, saj se po vsej verjetno nahajamo v zgodnji fazi gručenja. Iz tega sledi, da so zgodnje združitve ocenjene najbolje, tega si pa ne želimo, saj so podatki porazdeljeni v velikem številu gruč. Za ta primer smo se odločili za manjši poseg pri izračunu Silhuete. Vse izračune Silhuete, ki so enaki ena, preprosto ignoriramo in jih nadomestimo z oceno 0. V tem primeru so ocene Silhuete boljše v pozni fazi združevanj. Popravek je izpolnil naša pričakovanja, zato se lažje orientiramo pri pregledu ocen delitev, ki smo jih izračunali s pomočjo Silhuete.

4.5 Opis funkcionalnosti

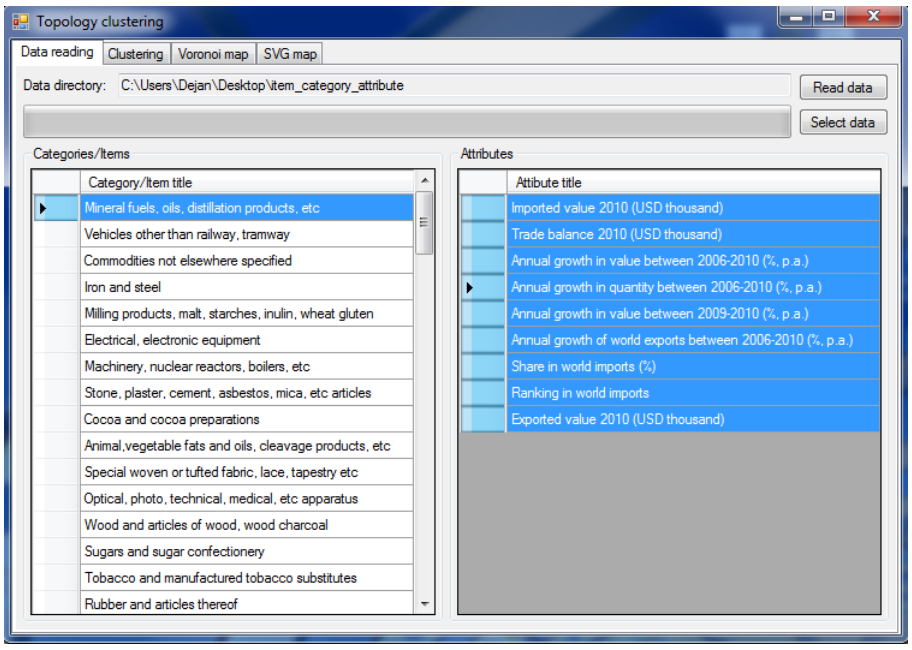
Ko smo pričeli z razvojem aplikacije, smo imel v mislih dva cilja. Prvič, aplikacija mora biti funkcionalno zadostna, zato smo poskušali vključiti vse potrebne funkcionalnosti. Drugi cilj je bil ta, da je aplikacija čimbolj večnamenska. Da smo izpolnili drugi cilj, smo definirali enoten format vhoda podatkov, ter prilagodili risanje Voronojevega diagrama.

Aplikacija omogoča branje podatkov v dveh formatih. V prvem načinu imamo definirane relacije med elementi, drugi način pa predstavlja relacije med elementi in kategorijami, ki jih upoštevamo pri gručenju podatkov. Zato potrebujemo tri ali štiri datoteke:

- Atributi.xml,
- elementi.xml,
- povezave.xml in
- kategorije.xml (samo za drugi način).

Prva datoteka vsebuje definicije vseh atributov, po katerih bomo gručili podatke. Druga datoteka vsebuje elemente, ki imajo definirane prostorske koordinate. Tretja datoteka vsebuje vrednosti relacij med atributi in elementi. Če gre za način relacij med dvema elementoma, potem vsaka povezava vsebuje dva identifikatorja elementov in identifikator atributa. Četrto datoteko uporabljamo pri načinu relacij med elementi in kategorijami, kjer so vsebovane definicije kategorij. Te kategorije so v datoteki povezav referencirane skupaj z elementi in atributi preko identifikatorjev.

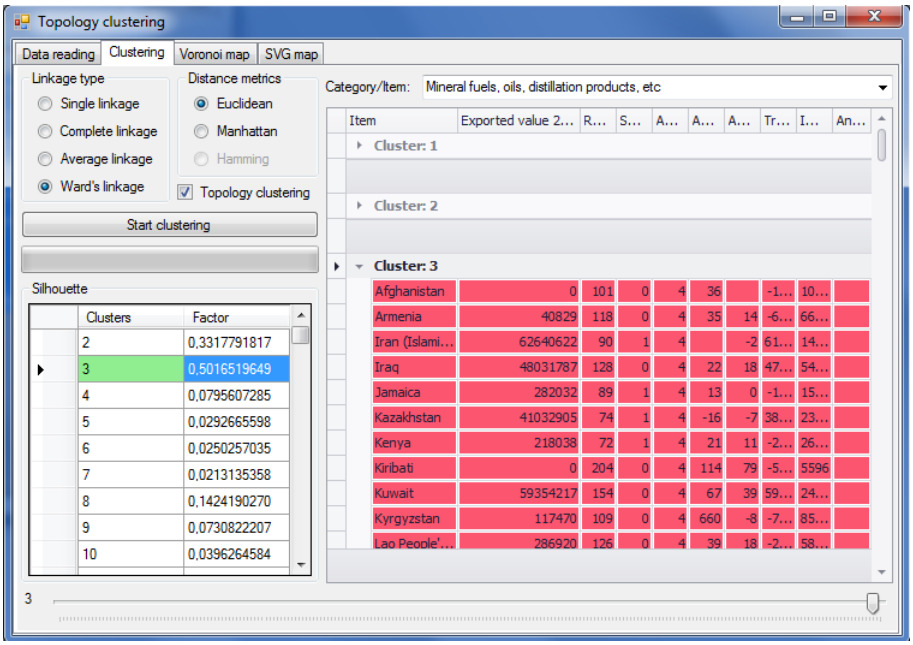
Na sliki 4.8 je prikazano okno branja podatkov. Ko uporabnik izbere mapo z datotekami, aplikacija preveri format datotek in prebere podatke v ustreznem formatu. Na levem seznamu se prikažejo kategorije ali elementi odvisno od formata podatkov. Na desnem seznamu se nahajajo vsi atributi, ki so na voljo. Ko uporabnik izbere ustrezne atribute ter elemente ali kategorije, pritisne na gumb izbire podatkov, ki ga popelje na okno gručenja podatkov.



Slika 4.8: Okno branja podatkov.

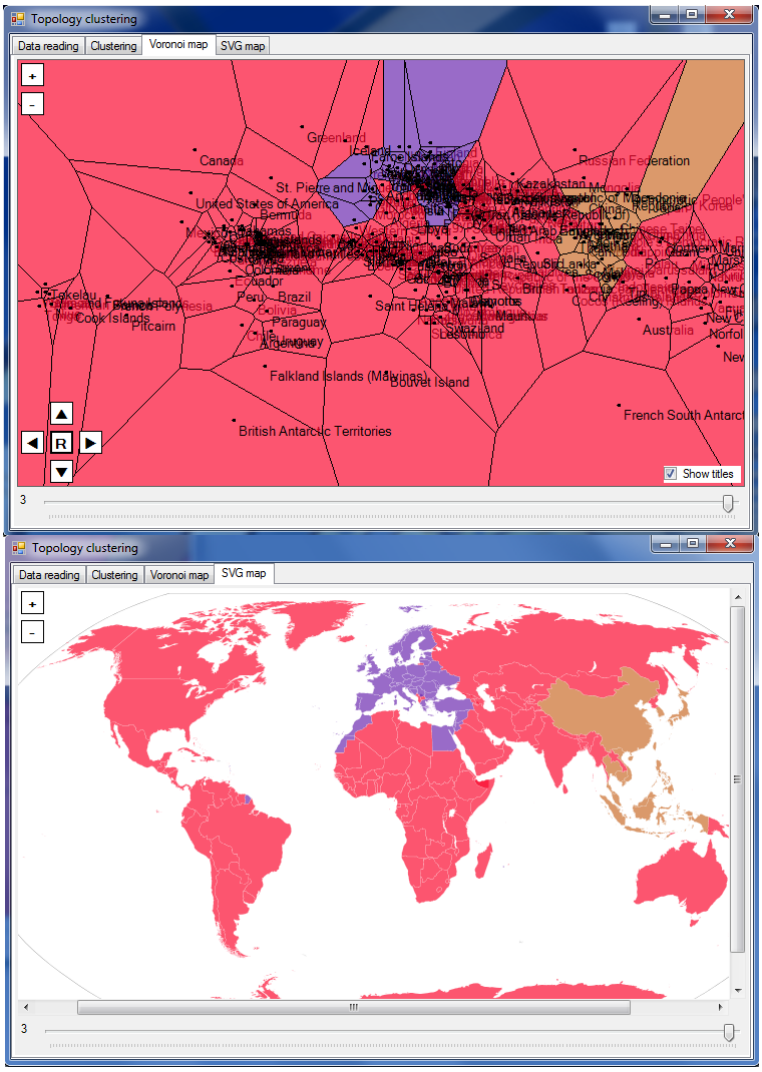
Če obstajajo prazne vrednosti atributov, se mora uporabnik odločiti, kako jih bo nadomestil. Na izbiro ima največjo vrednost, najmanjšo vrednost, povprečno vrednost ali ignoriranje atributa.

Okno za gručenje podatkov omogoča izbiro parametrov gručenja ter prikazuje rezultate gručenja. Na sliki 4.9 je prikazano okno gručenja podatkov. Zgoraj levo izbiramo metode združevanj, meritve razdalj ter upoštevanje topologije elementov. Izbiro med običajnim in topološkim gručenjem smo uporabili za primerjavo med načinoma, ki ga bomo predstavili v naslednjem poglavju. Ko se gručenje podatkov zaključi, se desni seznam napolni z vrednostmi atributov, ki so razvrščeni v gruče. Elementi gruč v seznamu so obarvani v enake barve kot gruče na Voronojevem diagramu in zemljevidu sveta. V levem seznamu se nahajajo izračuni vseh Silhuet za vse kombinacije gruč, ki so se formirale v poteku gručenja. Optimalna delitev je označena z zeleno barvo. Pod seznamom se nahaja drsnik, ki omogoča vpogled v hierarhijo gručenja podatkov. Drsnik se privzeto nastavi na optimalno delitev, ki smo jo izračunali s pomočjo Silhuete.



Slika 4.9: Okno gručenja podatkov.

Po zaključku gručenja podatkov se prikažeta zavihka pogleda Voronojevega diagrama in zemljevida, ki sta prikazana na sliki 4.10. Privzet pogled na Voronojevem diagramu je optimalna razporeditev točk, ki na zaslonu prikaže vse točke. Uporabnik ima omogočeno pomikanje pogleda v vse smeri, povečevanje z dvojnim klikom in prikazovanje nazivov elementov v prostoru.



Slika 4.10: Okno Voronojevega diagrama in zemljevida.

Elementi na Voronojevem diagramu in zemljevidu so obarvani enake barve kot gruče na seznamu na sliki 4.9. Spodaj se nahaja enak drsnik kot na oknu gručenja, ki omogoča vpogled v hierarhijo gručenja podatkov.

Poglavje 5

Primerjava običajnega in topološkega gručenja

Ko smo končali z razvojem aplikacije, nas je čakalo testiranje topološkega hierarhičnega gručenja na resničnih podatkih. Aplikacija omogoča gručenje v običajnem in topološkem načinu, kar smo izrabili za primerjavo obeh metod. Za primerjavo obeh metod gručenja smo uporabili podatke o ekonomskih kazalci držav sveta, ki smo našli na spletu [24]. Podatkovna zbirka je zelo velika, saj vsebuje 229 držav sveta, več kot 100 kategorij skupin izdelkov in 9 atributov, ki so našteti spodaj na seznamu. Podatke smo pridobili s pomočjo aplikacije, ki smo jo razvili za prenos datotek iz spleta. Aplikacija se je v ponavljajočem zaporedju navigirala po spletni strani in shranjevala datoteke posameznih držav sveta. Datoteke žal niso bile v ustreznem formatu, zato smo jih morali pretvoriti v naš format. V nadaljevanju bomo predstavili razliko med običajnim in topološkim gručenjem z različnimi metodami združevanj na izbranih kategorijah.

Kategorije:

- Kovine in jeklo,
- goriva, olja in destilacijski proizvodi,
- farmacevtski proizvodi,

- žita,
- steklo in stekleni izdelki,
- plastika in plastični izdelki,
- aluminij in aluminijasti izdelki,
- bombaž,
- itn.

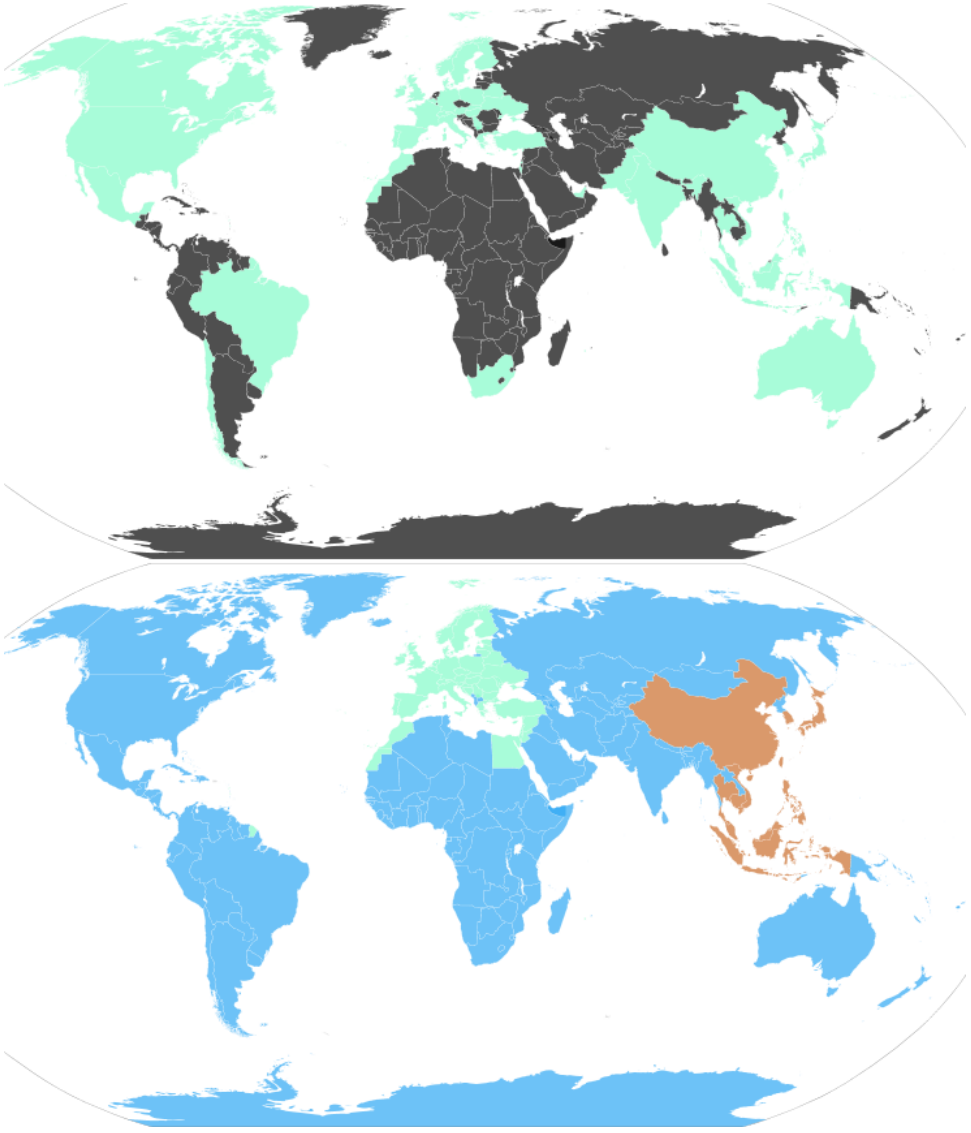
Atributi:

- Uvožena vrednost,
- izvožena vrednost,
- razlika med uvoženo in izvoženo vrednostjo,
- letna rast v vrednosti med letoma 2006 in 2010,
- letna rast v količini med letoma 2006 in 2010,
- letna rast v vrednosti med letoma 2009 in 2010,
- letna rast svetovnega izvoza vrednosti med letoma 2006 in 2010,
- delež svetovnega uvoza in
- razvrstitev glede na svetovni uvoz.

5.1 Fosilna goriva, nafta in destilacijski produkti

Odločili smo se, da bomo prikazali razliko med običajnim in topološkim hierarhičnim gručenjem na modelu element-kategorija-atribut s kategorijo fosilna goriva, nafta in destilacijski produkti. Uporabili smo Wardovo metodo združevanja in evklidsko metodo meritve razdalj. Manjkajoče podatke smo nadomestili s povprečno vrednostjo. Na sliki 5.1 sta prikazana zemljevida gruč podatkov za oba načina gručenja. Na zgornjem delu slike je prikazano

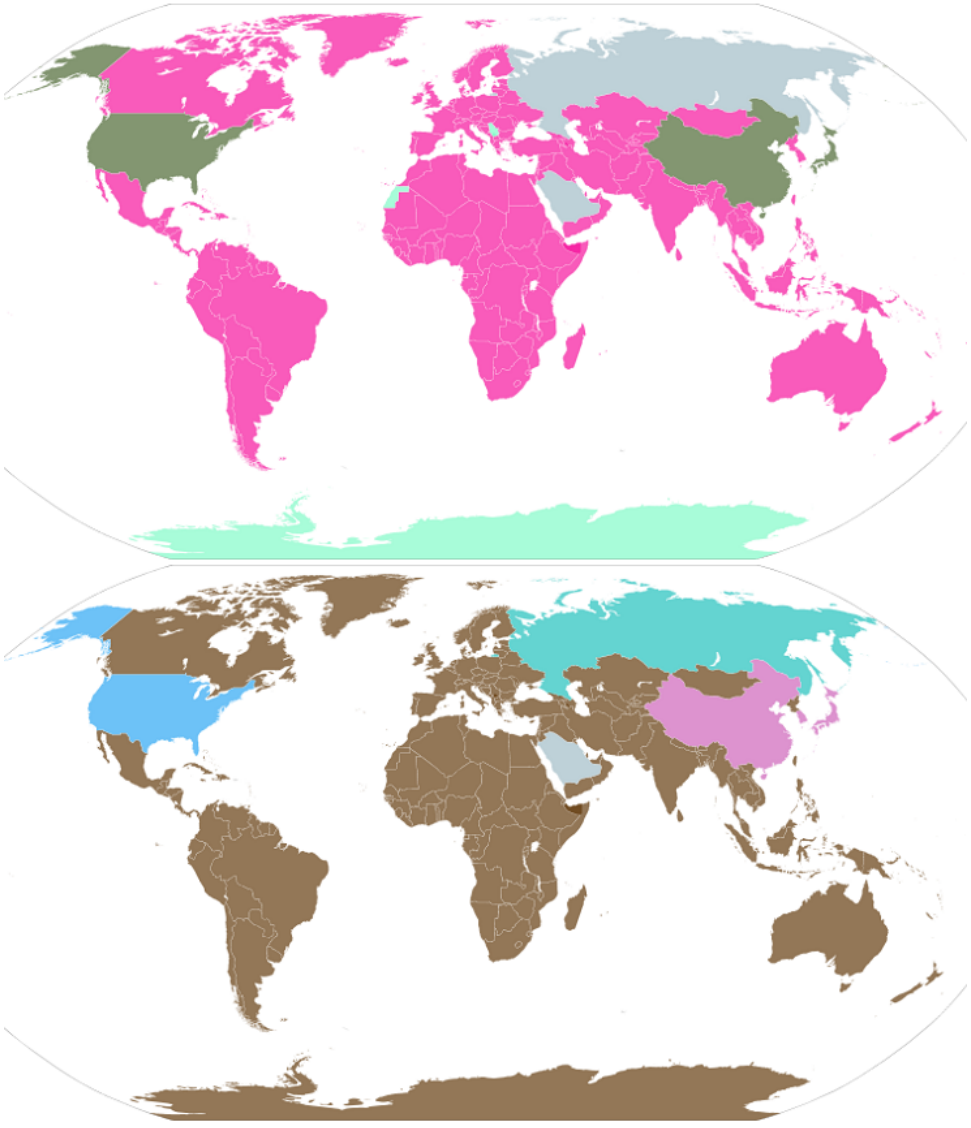
običajno gručenje podatkov z dvema gručama podatkov, spodaj pa topološko gručenje podatkov s tremi gručami podatkov, ki je bilo izbrano s pomočjo Silhuete.



Slika 5.1: Primerjava običajnega in topološkega gruč. z Wardovo metodo.

Osredotočimo se na običajno hierarhično gručenje, kjer lahko opazimo, da zelena barva predstavlja države, ki so gospodarsko bolj razvite, imajo razširjeno industrijo in predstavljajo večje finančne tokove. Iz tega lahko

predpostavimo, da države predvsem uvažajo nafto in omenjene surovine, kar tudi govorijo številke. V drugo skupino spadajo države, ki so v razvoju, imajo lastno proizvodnjo omenjenih surovin (Rusija in Savdska Arabija) ali pa je obseg porabe zanemarljiv morebiti zaradi velikosti države ali drugih dejavnikov. Največ naftnih surovin uvozijo Združene države Amerike, Kitajska in Japonska.



Slika 5.2: Primerjava običajnega in topološkega gruč. s povprečno metodo.

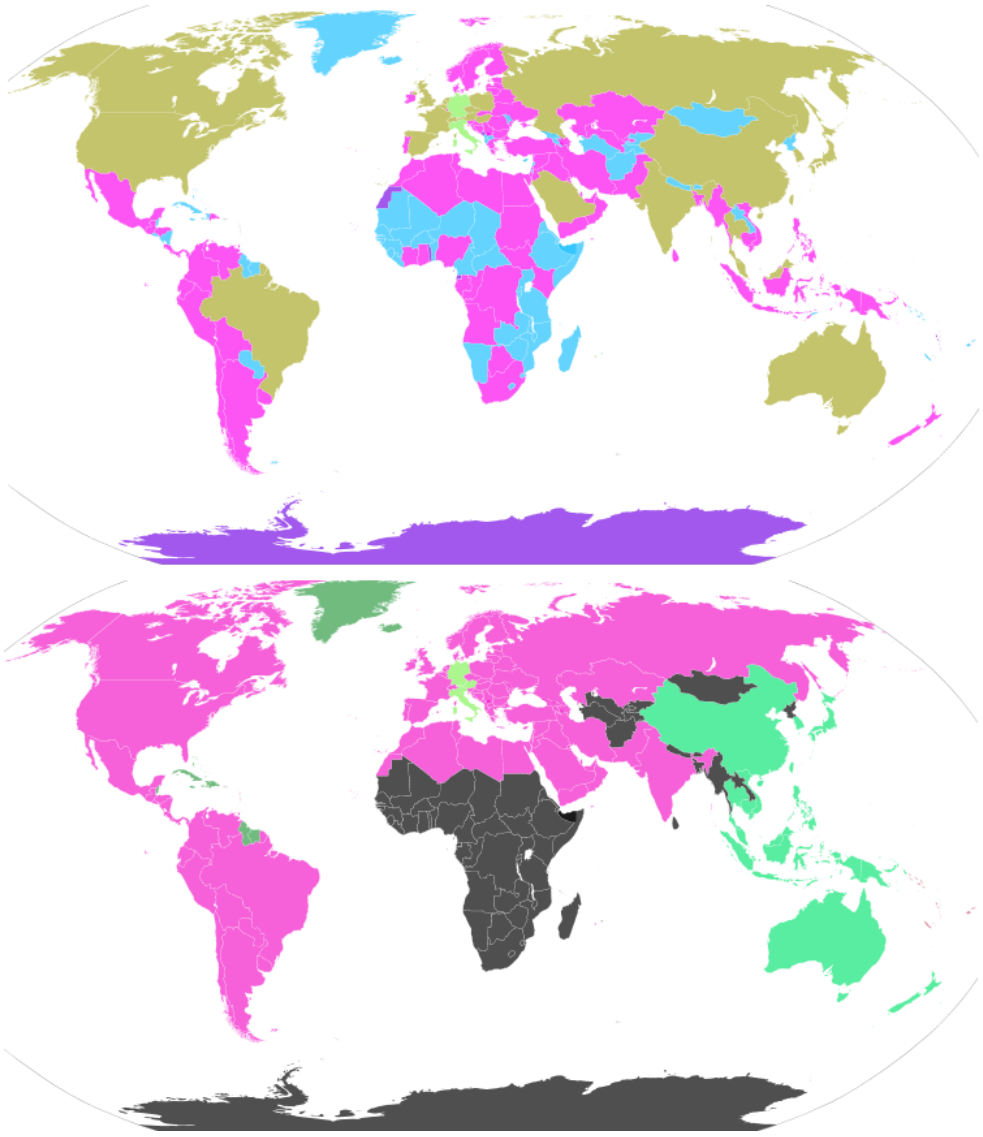
Pri topološkem hierarhičnem gručenju se gruče očitno razlikujejo od običajnega gručenja. Rjava in zelena barva predstavljata večje uvoznike nafte, modra pa ravno obratno. Na sliki vidimo, da se Azija in Evropa dokaj dobro ujemata pri obeh načinih gručenja, medtem pa države v Severni Ameriki, ki so imele večji uvoz nafte, sedaj spadajo v skupino z zmanjšanim uvozom. Iz tega primera lahko sklepamo, da topološko hierarhično gručenje podatkov z Wardovo metodo dobro izpostavi skupine zelo podobnih sosednjih elementov, medtem pa zanemari izjeme, ki nimajo podobnih sosed. Tak primer so Združene države Amerike, ki imajo največjo največjo porabo in uvoz nafte na svetu.

Preizkusili smo tudi ostale metode združevanj v navezi s topološkim gručenjem podatkov. Na sliki 5.2 je prikazana razlika med običajnim in topološkim gručenjem s povprečno metodo združevanja. Pri običajem gručenju sta najbolj zanimivi gruči obarvani v sivo in zeleno barvo. Sem spadajo države z največjim izvozom nafte (Savdska Arabija in Rusija) ter države z največjim uvozom nafte (ZDA, Kitajska in Japonska). Vse ostale države spadajo v sredino.

Gruče, ki so se formirale pri topološkem gručenju, so sicer drugačne, pravzaprav pa gre za izjeme (države z največjim uvozom in največjim izvozom), ki so se lokalizirale v svojo gručo zaradi topoloških omejitev. Zaradi prostorske nesosednosti sta na primer Rusija in Savdska Arabija razvrščeni v ločeni gruči. Na tem primeru lahko vidimo, kako velika razlika obstaja med Wardovo metodo in ostalimi metodami, ki temeljijo samo na osnovi povezave. Wardova metoda poskuša formirati gruče s čimbolj podobnim številom elementom in s tem nekoliko zanemari velike izjeme, medtem pa se ostale metode fokusirajo samo na razdaljo med primeroma, kar pripelje do nezmožnosti gručenja izstopajočih primerov s povprečno okolico. Na tem primeru lahko sklepamo, da topološko gručenje z metodo na osnovi povprečne povezave izpostavi lokalne ekstreme.

5.2 Trgovanje Slovenije z ostalim svetom

V tem podpoglavju bomo prikazali razliko med običajnim hierarhičnim gručenjem in topološkim hierarhičnem gručenjem z Wardovo metodo na modelu element-element-atribut. Na sliki 5.3 je prikazana razlika med obema načinoma gručenja na podatkih trgovanja Slovenije z ostalim svetom.



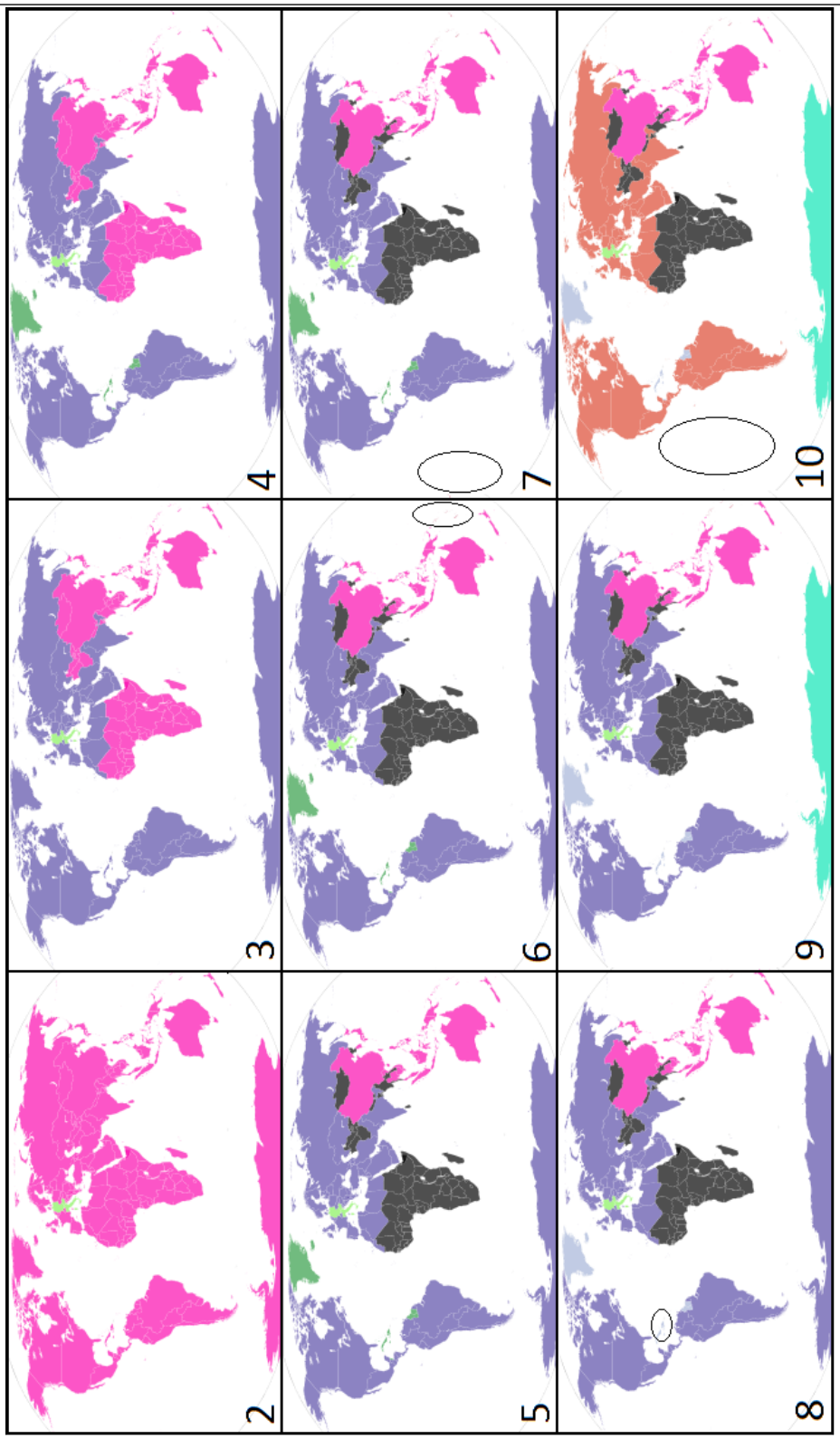
Slika 5.3: Primerjava običajnega in topološkega gruč. z Wardovo metodo.

Na zgornjem delu slike je zemljevid običajnega gručenja. Gruče obarvane v temno in svetlo zeleno barvo predstavljajo države s katerimi Slovenija največ trguje, pri čemer je uvoz večji od izvoza. Gruče obarvane v vijolično in svetlo modro predstavljajo države s katerimi Slovenija ne trguje veliko, temno roza barva pa označuje države v katere Slovenija večino izvaža.

V spodnjem delu slike se nahaja zemljevid topološkega gručenja podatkov. Gruče črne in temno zelene barve (Grenlandija) predstavljajo države s katerimi Slovenija ne posluje veliko. Svetlo in srednje zeleni predstavljata države, s katerimi Slovenija veliko trguje, kjer je uvoz večji od izvoza in temno roza predstavlja države, v katere večino uvažamo.

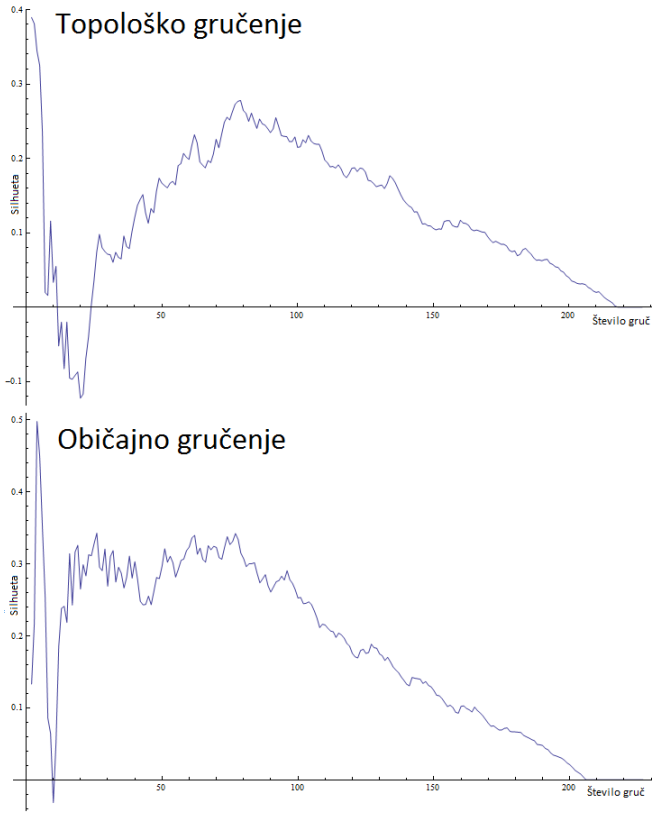
Najprej bomo primerjali države, s katerimi ne poslujemo veliko. Pri običajnem gručenju je v srednji in južni Afriki precej šuma, ki ga je topološko gručenje izničilo. Antarktika se je pridružila črni gruči, Grenlandija se je pa umestila v svoji gručo zaradi nesosednosti z Afriko. Zanimive so sosednje države (Italija, Avstrija in Nemčija), v katere največ uvažamo in največ izvažamo. Te so v enaki gruči v obeh primerih gručenja. Podobne lastnosti ima srednje zelena gruča (Kitajska in ostale azijske države), ki pa je ločena zaradi nesosednosti. V tem primeru je topološko gručenje zmanjšalo šum na območjih zmanjšanega trgovanja in poudarilo ekstreme, ki predstavljajo območja s povečanim trgovanjem.

Na sliki 5.4 smo prikazali zadnjih devet korakov gručenja. Pri delitvi na dve gruči opazimo močno vez pri podobnosti Slovenije s sosednjimi državami. Pri delitvi na tri gruče se od preostalega sveta odcepi Azija z Afriko. Ta del (roza) je ekonomsko neaktiven oz. predstavlja predvsem uvoz blaga. Pri naslednji delitvi (4) se od vijolične gruče odcepi Grenlandija z ostalimi državami, ki predstavljajo manjšo ekonomsko aktivnost, podobno kot Afrika, ki tudi na naslednjem koraku (5) odcepi od Azije, saj ima podobne karakteristike kot Grenlandija. V ostalih korakih (6,7,8,10) se predvsem delijo manjše otoške državice na Tihem oceanu in v Srednji Ameriki, ki so obkrožene na sliki.



Slika 5.4: Koraki topološkega gručenja.

Na sliki 5.5 je prikazana primerjava Silhuete na običajnem in topološkem gručenju. Vodoravna os prikazuje število gruč, horizontalna pa povprečno oceno trenutne delitve. Trend naraščanja ocene Silhuete v zgodnjih fazah združevanja do zadnje tretjine (od desne proti levi) je na obeh grafih podoben. V nadaljevanju se pri običajnem gručenju obdrži trenuten trend, medtem pa pri topološkem prične padati. Proti koncu pri obeh načinih ocena močno pade in ponovno naraste. V splošnem ima običajno gručenje ponavadi boljše povprečne ocene Silhuete. Razlog za to je po vsej verjetnosti topologija, ki vnaprej definira pravila, kako se bodo elementi povezovali med seboj. Vsaka gruča (ali element) ima določen nabor gruč, s katerimi se lahko združi, vendar v tem naboru po veliki verjetnosti ni gruče, s katero sta si najbolj podobni.



Slika 5.5: Primerjava Silhuete pri običajnem in topološkem gručenju.

Poglavje 6

Zaključek

Podatkovno rudarjenje je v današnjih časih nepogrešljivo za ustanove in podjetja, ki želijo biti čimbolj učinkovita in konkurenčna. Večinoma je uporabljeno za podporo odločanju. Gručenje podatkov je samo ena veja podatkovnega rudarjenja, ki ga uporabljamo za razdeljevanje objekov v skupine glede na kriterij podobnosti. V tem delu smo uvedli novo vrsto hierarhične gručenja podatkov z omejitvami, ki so vezane na topologijo elementov v prostoru.

V prvem delu diplomske naloge smo se srečali z vrstami gručenj podatkov. Nato smo prešli na osnovne gradnike hierarhičnega gručenja podatkov, kot so metode združevanj, metode meritev razdalj, grafični prikaz hierarhije gručenja, validacijska metoda Silhueta in metoda določanja sosednosti nove podvrste hierarhičnega gručenja. V drugem delu smo predstavili razvoj aplikacije, uporabljene tehnologije, strukturo projekta, opisali smo uporabljene algoritme in njihove optimizacijske metode. Predstavili smo tudi algoritem validacijske metode Silhueta in knjižico Qhull, ki nam je omogočala risanje Voronojevih diagramov, ki smo jih uporabili za določanje sosednosti elementov. V zadnjem delu smo predstavili delovanje aplikacije in njene funkcionalnosti, nato pa smo prikazali razliko med običajnim in topološkim gručenjem podatkov na resničnih podatkih. V prvem delu smo predstavili razlike med običajnim in topološkim gručenjem podatkov na modelu element-atribut-

kategorija z različnimi metodami združevanj na podatkih finančnih kazalcev trgovine z nafto in podobnimi izdelki. Nato smo preizkusili model element-element-atribut, kjer smo preiskali ekonomske kazalce trgovanja Slovenije z ostalim svetom.

Med izdelavo diplomske naloge in razvojem aplikacije se je prikazalo nekaj možnosti za izboljšavo aplikacije. Pri uporabi naših podatkov smo opazili težavo Voronojevega diagrama. Voronojev diagram je odličen za določanje sosednosti elementov, ki so prostorsko omejeni na ravno ploskev (npr. občine Slovenije). Mi smo zemljevid sveta preslikali na ravno ploskev in pri tem izgubili sosednost med Azijo in Ameriko. To je v določeni meri vplivalo na rezultate topološkega gručenja podatkov, česar si nismo želeli.

Vpeljava nove podvrste hierarhičnega gručenja podatkov, ki smo jo imenovali topološko gručenje podatkov, se je izkazala za dokaj uspešno, saj nam je na resničnih podatkih uspelo obrazložiti delovanje v primerjavi z običajnim gručenjem. Aplikacija, ki smo jo razvili, je izpolnila naša pričakovanja, njene funkcionalnosti pa so zadoščale našim potrebam. Uspelo nam je optimizirati kar nekaj algoritmov, ki so pohitrili delovanje aplikacije. Pomembna stvar je bil format vhoda podatkov, za katerega mislimo, da je dovolj vsestranski za katerekoli podatke, ki smo jih želeli podpreti. V splošnem smo zadovoljni s svojim rezultatom.

Literatura

[1] Machine Learning and Data Mining. Dostopno na:
http://www.slideshare.net/pierluca.lanzi/
machine-learning-and-data-mining-08-clustering-hierarchical

[2] Hierarhical Cluster Analysis. Dostopno na:
http://www.clustan.com/hierarchical_cluster_analysis.html

[3] Fortune's Voronoi algorithm. Dostopno na:
http://www.diku.dk/hjemmesider/studerende/duff/Fortune/

[4] Graph-based clustering. Dostopno na:
http://strehl.com/diss/node21.html

[5] Graph clustering, str. 27-29. Dostopno na:
http://dollar.biz.uiowa.edu/~street/graphClustering.pdf

[6] Cluster analysis: Basic concepts and algorithms, str. 488-495. Dostopno na:
http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

[7] Distances between Clustering, Hierarchical Clustering 36-350, Data Mining. Dostopno na:
http://www.stat.cmu.edu/~cshalizi/350/lectures/08/
lecture-08.pdf

[8] Hierarhical clustering, Online edition 2009 Cambridge. Dostopno na:
http://nlp.stanford.edu/IR-book/pdf/17hier.pdf

[9] Hierarchical clustering, David M. Ble, Princeton University. Dostopno na:
http://www.cs.princeton.edu/courses/archive/spr08/cos424/slides/clustering-2.pdf

[10] Cluster Analysis, Analyzing Multivariate Data, J Lattin, J D Carroll, P E Green. Dostopno na:
ftp://163.25.117.117/96emis/%C0%B3%A5%CE%A6h%C5%DC%B6q%A4%C0%AAR_%A6%BF%AB%DB%B6h/Multivariate/Cluster%20analysis/Clustering-Presentation.pdf

[11] Blaž Zupan, Janez Demšar, Vladislav Rajkovič, "Poslovna inteligenca (zapiski predavateljev, samo za interno uporabo)", str. 8-14, 9.januar 2012.

[12] Cluster analysis. Dostopno na:
http://en.wikipedia.org/wiki/Cluster_analysis

[13] Hierarhical clustering. Dostopno na:
http://en.wikipedia.org/wiki/Hierarchical_clustering

[14] Označevalni jezik XML. Dostopno na:
http://en.wikipedia.org/wiki/XML

[15] Razvojno okolje Visual Studio. Dostopno na:
http://en.wikipedia.org/wiki/Microsoft_Visual_Studio

[16] Ogrodje .NET. Dostopno na:
http://en.wikipedia.org/wiki/.NET_Framework

[17] Knjižnica MSDN. Dostopno na:
http://msdn.microsoft.com/en-us/ms348103.aspx

[18] Knjižnica Qhull. Dostopno na:
http://www.qhull.org/

[19] Voronojev diagram. Dostopno na:
`http://en.wikipedia.org/wiki/Voronoi_diagram`

[20] Delaunajeva triangulacija. Dostopno na:
`http://en.wikipedia.org/wiki/Delaunay_triangulation`

[21] Fortunov algoritev generiranja Voronojevih diagramov. Dostopno na:
`http://en.wikipedia.org/wiki/Fortune's_algorithm`

[22] Validacijska metoda Silhueta. Dostopno na:
`http://en.wikipedia.org/wiki/Silhouette_(clustering)`

[23] Validacijska metoda Silhueta. Dostopno na:
`http://blog.data-miners.com/2011/03/cluster-silhouettes.html`

[24] Podatki ekonomskih kazalcev držav sveta, Trade map. Dostopno na:
`http://www.trademap.org/index.aspx?ReturnUrl=%2fBilateral_TS.aspx`