SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Anto Čabraja

PARALELNI ALGORITMI ZA PROBLEM GRUPIRANJA PODATAKA

Diplomski rad

Voditelj rada:
prof. dr. sc. Goranka Nogo

Zagreb, srpanj 2014.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

Uvod

0.1 Problem grupiranja podataka

0.2 Primjena

0.3 Pregled rada

Poglavlje 1

Modeliranje problema grupiranja

1.1 Osnovni pojmovi

Kako bi u daljnjem razmatranju bilo jednostavnije objašnjavati strukture i same implementacije algoritama potrebno je problem grupiranja reprezentirati osnovnim pojmovima. U nastavku ćemo formalno definirati sve komponente od kojih se problem grupiranja sastoji.

Definicija 1.1.1. ***Uzorak*** *je apstraktna struktura podataka koja reprezentira stvarne podatke s kojima raspolaže algoritam za grupiranje.*

Definicija 1.1.2. ***Svojstvo*** *je vrijednost ili struktura koja predstavlja jednu značajku danog podatka unutar uzorka.*

Definicija 1.1.3. ***Udaljenost*** *između uzoraka definiramo kao funkciju* $f : D- > \mathbb{R}$*, gdje je* D *skup svojstava danih uzoraka*

Definicija 1.1.4. *Za uzorke kažemo da su* ***blizu*** *jedan drugome ako je njihova udaljenost manja od unaprijed zadane veličine*

Definicija 1.1.5. ***Klaster*** *je skup uzoraka koji su u prostoru podataka blizu. Ako su uzorci identični onda je njihova udaljenost uvijek* 0

Definicija 1.1.6. ***Jednistveno grupiranje*** *je postupak grupiranja kada svaki uzorak pripada jednom i samo jednom klasteru.*

Definicija 1.1.7. ***Nejasno ili nejedinstveno grupiranje*** *je postupak grupiranja gdje jedan uzorak može biti u više klastera.*

Napomena 1.1.8. *U radu ćemo promatrati* ***jedinstveno grupiranje*** *tako da će sve daljnje definicje i modeliranja predpostavljati da želimo dobiti disjunktne klastere.*

1.2 Matematičko modeliranje problema

Definicija grupiranja podataka nije jedinstvena. U literati se na različite načine pokušava opisati ovaj postupak. Neki od pokušaja opisne definicije su:

1. *Grupiranje podataka je postupak otkrivanja homogenih*[1] *grupa uzoraka unutar skupa svih danih uzoraka.*

2. *Grupiranje podataka je postupak određivanja koji su uzorci slični te ih svrstati u isti klaster.*

Za modelirali problem neće nam biti dovoljne opisne definicije. U ovom slučaju opisne definicje mogu poslužiti samo kao intuicija o ćemu se zapravo radi kada govorimo o grupiranju. U nastavku ćemo pomoću definiranih pojmova u poglavlju 1.1 matematički opisati problem grupiranja podataka.

Neka je $\mathbf{U} = \{U_1, U_2, \cdots, U_n\}$ skup od n uzoraka,te neka je $U_i = (s_1, s_2, \cdots, s_d)$ reprezentiran d-dimenzionalnim vektorom gdje s_i predstavlja jedno svojstvo. Ovako definiran $\mathbf{U}$ moguće je reprezentirati kao matricu $\mathbf{S}_{d\times n}$.Svaki stupac te matrice predstavlja jedan uzorak iz danog skupa $\mathbf{U}$.

$$\mathbf{S}_{d\times n} = \begin{pmatrix} s_{1,1} & s_{1,2} & \cdots & \cdots & s_{1,n} \\ s_{2,1} & s_{2,2} & \cdots & \cdots & s_{2,n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ s_{d,1} & s_{d,2} & \cdots & \cdots & s_{d,n} \end{pmatrix} \tag{1.1}$$

Iz definicje 1.1.6 te iz navedenog formalnog zapisa dajemo formalnu definiciju problema grupiranja.

Definicija 1.2.1. ***Skup od k klastera*** $\mathbf{C} = \{C_1, C_2, ..., C_k\}$ *je skup sa sljedećim svojstvima:*

- $C_i \neq \emptyset$
- $C_i \cap C_j = \emptyset$, $\forall i, j \; i \neq j$
- $\cup_{i=1}^{k} C_i = \mathbf{U}$

Napomena 1.2.2. *U terminima matrice* $\mathbf{S}$ *to znači da se svaki* C_i *zapravo sastoji od stupaca matrice* $\mathbf{S}$*.*

Definicija 1.2.3. ***Problem grupiranja*** *u skup od k klastera* $\mathbf{C}$ *je ekvivalentan problemu da* $\forall c, c' \in C_i$ *udaljenost od c do c′ je manja od udaljenosti c do bilo kojeg drugog* $c'' \in C_j$ $j \neq i$

[1] podaci koji se ne mogu smisleno separirati

Zapravo problem grupiranja je pronalazak najpogodnije particije za $\mathbf{C}$ u skupu svih mogućih particija. Prema napomeni 1.2.2 lako se zaključi da se problem grupiranja svodi na problem raspodjele n stupaca matrice $\mathbf{S}$ u k skupova. Uvedena matematička notacija za problem grupiranja omogućuje da postavimo model za rješavanje, koji je u kasnijem razmatranju pogodan za modeliranje i implementaciju.
Neka je $C = \{\mathbf{C}^1, \mathbf{C}^2, ..., \mathbf{C}^{N(n,k)}\}$ skup svih mogućih rješenja danog problema grupiranja, gdje je

$$N(n,k) = \frac{1}{k!}\sum_{i=1}^{k}(-1)^i\binom{k}{i}(k-i)^i \tag{1.2}$$

broj mogućih rješenja za raspodjelu n uzoraka u k klastera. Rješenje problema svodi se na optimizacijski problem

$$\underset{\mathbf{C}\in C}{optimiziraj}\ f(\mathbf{S}_{d\times n}, \mathbf{C}) \tag{1.3}$$

gdje je f funkcija dobrote rješenjea $\mathbf{C}$, Funkcija dobrote je funkcija koja mjeri kvalitetu rješenja za dani problem. Ovisno o tome kako je zadana može se gledati problem maksimizacije ili minimizacije. Gotovo uvijek njome se određuju jedan od dva važna parametra:

- koliko su blizu uzorci u danom klasteru
- koliko su blizu dva disjunktna klastera

Ako promatramo udaljenosti uzoraka unutar klastera, onda nam se problem optimizacije 1.3 svodi na problem minimizacije funkcije f, jer želimo postići što manju udaljenost uzoraka unutar jednog klastera. Međutim ako želimo postići da su nam klasteri međusobno disjunktni i da granica disjunkcije bude čvrsto definirana moramo promatrati udaljenosti među klasterima. U ovom slučaju potrebno je maksimizirati problem to jest tražiti za koju particju će vrijednost funkcije f biti najveća.
Konačno, sada znamo uzorke reprezentirati kao n-dimenzionalne vektore, također znamo definirati klaster kao skup vektora, te smo postavili model za određivanje kvalitete određenog klastera. Preostali posao je pronaći konkretnu funkciju f koja će na adekvatan način reprezentirati udaljenost između uzoraka.Vrlo je važno definirari od koji se komponenti uzorak sastoji. Osnovna podjela uzoraka je na *numeričke* i *kategoričke*. Numerički uzorak je onaj uzorak čije su sve vrijednosti numeričkog tipa, dok je kategorički onaj uzorak čije vrijednosti poprimaju vrijednosti nekih kategorija. Više o mogućnostima izgleda svojstava uzorka biti će rečeno u cijelini 1.4. U ovom trenutku za definiranje funkcije dobrote potrebno je samo imati u vidu da podaci ne moraju biti jednostavni, ali i dalje se od funkcije dobrote očekuje da na adekvatan način odredi udaljenost između dva uzorka.

Uzorak s numeričkim značajkama

Ukoliko su nam uzorci takvi da ih možemo predstaviti kao vektor numeričkih podataka tada ih možemo smjestiti u vektoriski prostor, te na njima upotrijebiti neku od standardnih vektorskih normi. Za problem grupiranja podataka u praksi najčešće se koristi Euklidska ili neka od p-normi [liter]. Euklidska norma daleko je najpopularnija i glavni je predstavnik normi koje mjere različitost među uzorcima. Drugim rječima za Euklidsku normu vrijedi da su uzorci više različiti što je vrijednost norme veća.
Neka su S_1 i S_2 dva uzorka sa n značajki, tada udaljenost d između dva uzorka u Eklidskoj normi računamo kao:

$$d(S_1, S_2) = \sqrt{\sum_{i=1}^{n} (s_{i,1} - s_{i,2})^2} \tag{1.4}$$

gdje su $s_{i,1}$ i $s_{i,2}$ značajke u uzorcima S_1 i S_2. U kreiranju rješenja sa numeričkim podacima uvijek ćemo koristit Euklidsku normu s malim izmjenama ovisno o vrsti problema. Međutim u praksi se vrlo često pojavljuje i noram koja koristi svojstva kovarijacijske matrice uzoraka[liter].

$$d(S_1, S_2) = (S_1 - S_2)^T \Sigma^{-1} (S_1 - S_2) \tag{1.5}$$

S_1 i S_2 uzorci, Σ kovarijacijska matrica uzoraka. Za ovo normu također vrijedi da su podaci više različiti što je vrijednost d veća. Također postoji veza između Euklidske norme i norme s kovarijacijskom matricom što je detaljno objašnjeno u [liter]. Samo ćemo reći da ako je Σ dijagonalna matrica onda se pripadna udaljenost zove normalizirana Euklidska udaljenost.
Osim normi koje mjere razlike među podacima, postoje i norme koje mjere sličnot. Norme koje mjere sličnost često su baziranje na određivanju kuta između dva uzorka u vektorskom prostoru. U [liter] objašnjene su neke od popularnijih normi ovog tipa.

Uzorak s kategoričkim značajkama

Individualna usporedba dva uzorka s kategoričkim značajkama vrlo često nema važnost i jedina povratna informacija je koliko značajki u uzorcima ima jednaku vrijednost. Ako ipak želimo postići da usporedbom dvije kategorije možemo zaključiti koliko su značajke u uzorku slične, to jest blizu, moramo svakoj značajki pridružiti težine i definirati operaciju razlike. Ako bolje pogledamo tim postupkom zapravo smo kategorije pretvorili u numeričke vrijednosti sa posebno definiranom funkcijom razlike. Promotrimo sljdeći primjer.

Primjer 1.2.4. *Neka nam je na raspolaganju skup podataka o cvijeću, te neka nam je svaki cvijet zadan kao vektor s tri kategoričke komponente. Ounačimo sa S i S' dva cvijeta iz*

skupa zadanih uzoraka.

$$S = (crven, amerika, iglicasto)$$
$$S' = (plav, europa, iglicasto)$$

Lako vidimo da jedini zaključak kojeg iz ovih podataka možemo dobiti jest da se na prve dvije komponente razlikuju, dok je na trećoj vrijednost kategorije ista. Ako bi naprimjer htjeli kategorijama pridružiti vrijednost morali bi definirati što znači da su amerika i europa različite i koliko nam je to bitno svojstvo. Važnost svojstva ima ključnu ulogu i vrlo je teško empirijski procijeniti kakve numeričke podatke dodjeliti kategorijama.

U praksi nismo uvijek u mogućnosti predvidjeti sve moguće kategoričke vrijednosti, pa samim time ne možemo svakoj kategoriji pridružiti numeričku vrijednost. Često korištena i popularna metoda za određivanje pripadnosti određenoj klasi, kada se radi o kategoričkim vrijednostima, je traženje udjela pojavljivanja određene kategoričke vrijednosti u promatranom klasteru.U tom slučaju ne promatramo udaljenost dva uzorka nego koliko u klasteru ima kategorički vrijednosti sličnih onima koje obilježavaju novi uzorak. Drugim rječima promatramo sličnost uzorka sa klasterom. Neka je $\mathbf{C} = \{C_1, C_2, ...C_n\}$ skup od n klastera. Neka je $c = (c_1, c_2, ..., c_d)$ uzorak sa d značajki istog tipa kao i uzorci iz zadanog skupa klastera. Definiramo funkciju sličnosti s kao

$$s(c, C_x) = \sum_{i=1}^{d} \frac{find(c_i, C_x)}{|C_x|} \tag{1.6}$$

gdje je $C_x \in \mathbf{C}$, funkcija $find$ vraća broj pojavljivanje značajke c_i na i-tom mjestu u svakom uzorku iz skupu C_x, a $|C_x|$ označava broj uzoraka u skupu C_x. Važno je naglasiti da se promatra samo i-to mjesto u svim značajkama. Postoji mogućnost da se vrijednost značajke, to jest kategorija, na i-tom i j-tom mjestu u uzorku podudaraju. Na primjer ako imamo uzorak koji predstavlja automobile, te su dvije od značajki danog uzorka boja interijera i boja automobila. Očito dvije navedene značajke mogu poprimiti istu vrijednost, ali zapravo su potpuno disjunktne i kao takve ih treba promatrati. Konačno c pridružimo onom klasteru za koji funkcija s vrati najveću vrijednost.
Kada promatramo određeni skup podataka i kreiramo vektore uzoraka, u svakom trenutku znamo veličinu tog uzorka. Prilikom modeliranja problema sami određujemo skup značajki koji koristimo. Saznanje o broju značajki i njihovoj važnosti možemo upotrijebiti kako bi svakoj značajki odredili težinu. Naime, kako nisu sve značajke jednako bitne želimo postići da se utjecaj nekih značajki restringira, odnosno naglasi. S obzirom na navedeni zahtjev modificiramo funkciju s iz 1.6 te definiramo novu funkciju s_w.

$$s_w(c, C_x) = \sum_{i=0}^{d} w_i \frac{find(c_i, C_x)}{|C_x|} \tag{1.7}$$

Kao što vidimo svakoj značajki unutar uzorka pridružili smo težinu w_i. Važno svojstvo funkcije s_w je da omogućuje potpuno isključivanje nekih od značajki. Navedeno svojstvo u praksi često služi za empirijsko otkivanje nebitnih značajki. Otkrivanje i uklanjanje značajki čije postojanje ne pridonosi poboljšanju rješenja uvelike smanjuje prostornu složenost što će detaljno biti obrađeno u 1.4.

Uzorak s hibridnim značajkama

Priroda problema za koje pokušavamo rješiti problem grupiranja često je složene strukture te je nemoguće podatke reprezentirati uzorcima za značajkama samo jednog tipa. U takvom slučaju podatak se reprezentira za uzorkom čije značajke mogu biti numeričkog i kategoričkog tipa. Ovako definirana reprezentacija podataka povećava složenost i modeliranje funkcije dobrote, odnosno odluka o tome koliko su dva uzorka slična. Označimo sa $S = (s_1, s_2, ..., s_n)$ uzorak od n značajki numeričkog i kategoričkog tipa. Bez smanjenja općenitosti možemo predpostaviti da su na prvih l komponenti numeričke značajke, a na ostalih $n - l$ kategoričke. Funkciju udaljenosti d_h između dva uzorka S_1 i S_2 s hibridnim značajkama definiramo kao

$$d_h(S_1, S_2) = d(S_1(: l), S_2(: l)) + \frac{1}{s_w(S_1(l+1:), C_{S_2})} \tag{1.8}$$

gdje je d funkcija definirana u 1.4, s_w definirana u 1.7, a C_{S_2} označava klaster u kojemu se nalazi S_2. Važno je napomenuti da S_1 ne mora nužno biti u klasteru dok S_2 mora. Dakle ako dodajemo novi uzorak i želimo mu pronaći klaster to ćemo učiniti tako da ga stavimo kao prvi argument funkcije d_h. Kao što vidimo vrijednost funkcije s_w može biti jednaka nuli pa nam drugi sumand u definiciji funkcije d_h nije definiran. U praksi se eksperimentalno odredi kolika je važnost da uzorak nema niti jednu katekoričku značajku istu kao značajke unutar promatranog klastera, te se funkcija d_h dodefinira s obzirom na slučaj kada je $s_w = 0$. Neka je λ eksperimentalno određena vrijednost tada se potpunda definicija funkcije d_h može zapisati

$$d_h(S_1, S_2) = \begin{cases} d(S_1(: l), S_2(: l)) + \frac{1}{s_w(S_1(l+1:), C_{S_2})} & s_w \neq 0 \\ \\ d(S_1(: l), S_2(: l)) + \lambda & s_w = 0 \end{cases} \tag{1.9}$$

Ovako definitrana funkcija d_h je korektna. Naime, za sve parove uzoraka uvijek možemo odrediti vrijednost funkcije d_h jer su sve komponente dane funkcije dobo definirane. Funkcija d je zapravo standardna Euklidska norma pa je kao takva definirana za sve vektore. Funkcija s_w je kompozicija funkcije $find$, funkcije zbrajanja, množenja i kardinalnog broja. Sve funkcije su dobro definirane[liter] pa je i kompozicija dobro definiran. Primjerimo da se ovako definirana funkcija d_h, sa pravilno postavljenim paremetrom λ može koristit za sve slučajeve uzoraka: numeričke, kategoričke ili hibridne.

1.3 Metode razvoja algoritama za grupiranje

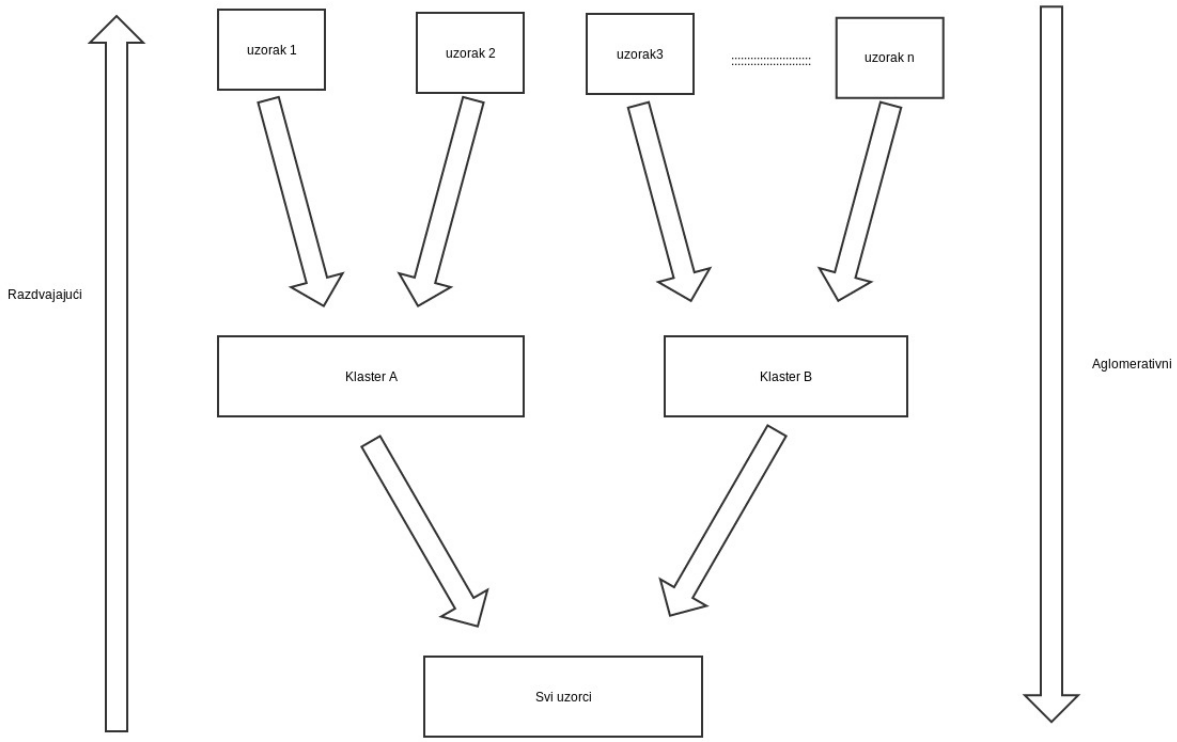
U postupku modeliranja rješenja za problem grupiranja koristi se više metoda. Metoda za rješavanje ima puno i znastvenici intenzivno rade na pronalasku novih i objašnjenju kvalitete postojećih metoda. Danas je ovo područje izuzetno cjenjeno i svako novo saznanje može dovesti do revolucionarnog napretka u rješavanju izuzetno teških problema. Ipak, dvije metode su se pokazale kao općenitije i mogu poslužiti kao predložak za rješavanje većeg broja problema. Konkretno radi se o *hijerarhijskom* i *particijskom* pristupu rješavanju problema. Među ostalim metodama koje se danas ističu kao jako korisne pokazale su se : evolucijske metode, metode particioniranja grafa, metode bazirane na neuronskim mrežama, te metode s gradinjentnim spustom[liter]. U ovom radu obradit ćemo navedene najpoznatije metode, a od ostalih metoda obradit ćemo evolucijske metode kojima je posvećena cjelina 2.

Hijerarijsko grupiranje

U hijerarhijskom grupiranju na početku procesa podaci nisu unaprijed smješteni u klastere nego postoje dva načina inicijalne raspodjele podataka.

1. Svaki uzorak je klaster
2. Svi uzorci su u jednom klasteru

Postupak grupiranja može se odvijati u dva smjera ovisno o inicijalnoj raspodjeli podataka. Tako u sličaju 1. kada su uzorci svaki za sebe u posebnom klasteru postupak grupiranja nastavlja se spajanjem uzoraka sa bliskim svojstvima u jedan klaster. U svakom sljedećemo koraku ponavlja se postupak te se klasteri sa sličnim svojstvima spajaju u jedan novi klaster. Postupak prestaje kada odlučimo da nam je funkcija dobrote postigla željenu točnost ili smo postigli željeni broj klastera. Primjetimo ako postupak izvršimo bez uvjeta zaustavljanja dobit ćemo jedan klaster i zapravo podaci će biti u 2. inicijalno stanje. Ako krećemo iz 2. inicijalnog stanja to jest na početku su nam svi uzorci smješteni u jedan klaster onda u svakom sljedećem koraku radimo separaciju uzoraka. Saparacija uzoraka se provodi nad uzorcima koji se najviše razlikuju te se tako dobije više klastera. Postupak također nastavljamo dok ne dobijemo željenu točnost ili željeni broj klastera. Prva navedena metoda kada na početku imamo veliki broj klastera, točnije svaki uzorak je klaster, se naziva *aglomerativni postupak*. Druga metoda se naziva *razdvajajući postupak*. Iz opisa hijerarhijske metode vidimo kako u navedenim postupcima gradimo hijerarhiju te je se navedeni postupak može prikazati kao stablo gdje su čvorovi klasteri, a veze povezuju klastere koji su nastali ili su spojeni iz čvorova djece. Tako kreiran graf kao struktura stabla naziva se *dendogram* i na slici 1.1 prikazan je postupak grupiranja pomoću navedenog grafa.



Slika 1.1: Hijerarhijski proces grupiranja

Hijerarhijsko grupiranje ima dva važna svojstva:

- Broj klastera ne mora biti unaprijed poznat
- Klasteri su neovisni o početnim uvjetima

Što se tiče vremenske i prostorne složenosti, ova metoda nije pogodna za velike skupove podataka. Vremenska složenost je $O(n^2 log(n))$, dok je prostorna $O(n^2)$. Također postupak grupiranja je statički, to jest ako je neki uzorak pripao nekom klasteru ne postoji mogućnost da prijeđe u neki drugi klaster na istom nivou hijerarhije.

Particijsko grupiranje

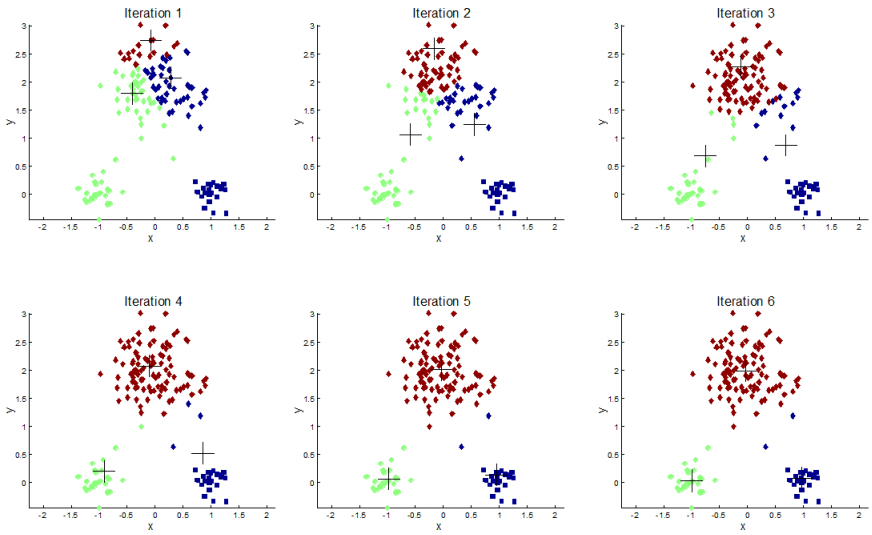
Za postupak particijskog grupiranja uzorke inicijalno podjelimo u skup klastera. Inicijalna podjela u prvom koraku u praksi se najčešće izvodi na slučajan način. Međutim, ako su nam unaprijed poznata neka saznanja o uzorcima moguće je već u prvom koraku usmjeriti uzorke u prave klastere. Inicijalna podjela se izvodi tako da se na slučajan način izabere k uzoraka i oni predstavljaju klastere, nakon toga se svi ostali uzorci rasporede u k klastera

prema udaljenosti od izabranih predstavnika. Nakon inicijalne podjele za svaki klaster ponovo se odredi predstavnik klastera. Predstavnik je uzorak iz klastera koji predstavlja centar dotičnog klastera i s obzirom na njega se u klaster ubacuju novi uzorci. Uvodi se funkcja udaljenost uzoraka kako smo to definirali u cjelini 1.2 koja služi kao ocjena sličnosi. Postupak izvodimo iterativno tako da pogledamo sve uzorke i pridružimo ih predstavniku klastera kojem su najviše slični, to jest kojem su najbliže. Nakon postupka pridruživanja ponovo se bira novi predstavnik klastera i za sve podatke se pogleda sličnost s novim predstavnikom klastera. Particijsko grupiranje može se generalno prikazati pseudokodom:

Algorithm 1: Particijsko klasteriranje

Na slučajan način izabrati k predstavnika m_i, $i \in \{1, 2, ..., k\}$;
while *nije zadovoljen uvijet zaustavljanja* **do**
  for *svaki uzorak* Z_i *iz skupa uzoraka* **do**
    izračunaj $veza(m_j, Z_i)$ za svaki centroid i svaki uzorak pridruži centroidu sa najmanjom vrijednosti *veza*.;
  pronađi nove predstavnike klastera m_j kao srednju vrijednost svih elemenata klastera;

Funkcija veza je funkicja koja u pozadini koristi neku od definiranih funkcija u 1.2. Svaki uzorak će biti u klasteru od čijeg je predstavnika najmanje udaljen. Računanje novih predstavnika uzimanjem u obzir srednje vrijednosti ovisi o tipu problema i može se mjenjati s obzirom na zahtjeve.



Slika 1.2: Particijski proces grupiranja

Slika 1.2 Prikazuje nekoliko iteracija praticijske metode klasteriranja. Možemo primjetiti kako se u svakoj iteraciji promijenio predstavnik klastera te kako s obzirom na novog predstavnika uzorci mjenjaju klaster.
Kao što vidimo ovaj model grupiranja predviđa poznavanje broja klastera. Međutim složenost algoritma je znatno manja od prve navedene metode. Svaki prolazak treba n koraka, po jedan korak za svaki uzorak, te je stoga složenost $O(n)$. U svakom koraku uzorci mogu promjeniti klaster i tako iterativnim postupkom do detalja profiniti particije. Ova metoda danas je najpoplarnija i u praksi se najviše koristi. U kombinaciji sa nekom od ostalih metoda kao što su evolucijske metode može se varirati i broj klastera te tako uvjetno popraviti jedini nedostatak ove metode. Danas najpoznatiji algoritam za grupiranje, algoritam k-sredina, je predstavnik particijskog grupiranja. U ovom radu naglasak će biti upravo na ovoj metodi te ćemo za nju kreirati većinu algoritama i testiranja.Između ostalog obradit ćemo i navedeni algoritam k-sredina. Opis dvije najpoznatije metode daje širok pregled na probleme koji se javljaju prilikom pokušaja grupiranja podataka i veliki je izazov postaviti model koji će minimizirati sve nedostatke.

1.4 Upravljanje podacima

Podaci nad kojima treba provesti algoritam grupiranja gotovo uvijek su jako komplicirani. Također nije poželjno stvoriti uzorke tako da se iskoriste sve značajke koji opisuju dani podatak. Naime, kada želimo spremiti podatke da bi ih mogli kasnije upotrjebiti u algoritmu potrebno je odabrati značajke koje su zajedničke svim podacima. Vrlo je bitno da značajki ima dovoljno da se podaci mogu separirati, ali da ih ne bude prevelik broj jer nam prostorna i vremenska složenost raste. Dakle lako je zaključiti da moramo postići kompromis između dvije osnovne komponente. Kada bi spremili sve značajke danog podatka i korisitili tako definirane uzorke svaki uzorak bi nam bio reprezentiran sa velikim brojem značajki. Gotovo nikada ne bi uspijeli postići da svi podaci imaju jednak broj značajki pa bi nam uzorci bili različitih duljina. Ako su različiitih duljina nemoguće je uspoređiati takve uzorke niti donijeti smislene zaključke. S druge strane ako odlučimo restringirati značajke tako da su svi jednake duljine kao onaj s najkraćom duljinom pojavit će se problem ne adekvatnih značajki. Sve u svemu moramo posebnu pažnju posvetiti odabiru značajki i kreiranju uzorka. Kako bi postupak bio jasan provet ćemo ga na jednom primjeru te navesti sve detalje na koje treba obratiti pažnju.

Primjer 1.4.1. *Pretpostavimo da su nam dani komentari o automobilima. Svaki komentar se sastoji od rečenica koje govore o kvaliteti lima automobila, o opremi automobila, o snazi motora, kvaliteti interjera,te dostupnosti na tržištu. Zadatak je iz komentara izdvojiti rečenice i svaku rečenicu staviti u klaster ovisno o tome o čemu rečenica govori.*

Komentar:
Boja lima je idalne kvalitete, ne gubi intenzitet na suncu. Korozija je značajno zabilježena tek na modelima starijima od 10 godina. Izuzetno kvalitetan interijer sa opremom kao što je električni paket, sustav protiv proklizavanja, klima i ostala oprema. Automobil je dostupan u tri varijante kao sportback, limuzina, karavan. Cijena mu se na tržištu kreće oko 10000 eura.

Rješenje:
Prvi korak u konstrukciji uzorka je otkriti tip podatka kojeg će biti značajke. Konkretno za primjer kojeg promatramo prirodno je da značajke budu riječi. Kako je naš zadatak klasificirati rečenice promotrimo kao primjer prvu rečenicu.
Boja lima je idalne kvalitete, ne gubi intenzitet na suncu. Prvo možemo iz rečenice izbaciti veznike, zatim rječi kao što su *ni*, *ne*. Konačno dobijemo sljedeći vektor sa korjenima rječi $v_1 = (boja, lim, idealan, kvaliteta, gubi, intenzitet, sunce)$. Parsiranjem i čišćenjem druge rečenice dobijemo vektor $v_2 = (korozija, znacajno, zabil jezena, modelima, stari ji, 10, godina)$. Prva i druga rečenica govore o kvaliteti lima, imaju sedam značajki ali kao što možemo primjetiti niti jedna značajka od v_1 se ne podudara sa značajkama iz v_2. Svaki algoritam za grupiranje sa standardnim funkcijama koje mjere veze odnosno sličnost podataka neće prepoznati ove dvije rečenice kao uzorke iz istog klastera. Prvi korak u rješavanju nastalog problema je kako prepoznati značajke specifične za određenu rečenicu. Ovaj postupak nema egzaktan odgovor i u praksi se dizajneru rješenja prepušta da analizira podatke i izbaci rječi za koje procijeni da su previše specifične. Matematički ovaj problem se rješava računanjem frekvencija pojavljivanja rječi, te se izbacuju one s najmanjom frekvencijom. Pogledajmo ponovo vektore značajki. U v_1 možemo primjetiti kako su rječi *idealna*, *kvaliteta*, *gubi* općenite i nikako ne reprezentiraju određenu kategoriju. U v_2 vidimo da su sve riječi osim *korozija* općenite te nikako ne opisuju lim automobila. Kada izbacimo sve nepoželjnje rječi dobijemo sljedeća dva vektora

$$v_1 = (boja, lim, intenzitet, sunce)$$

$$v_2 = (korozija)$$

Međutim opet imamo problem da vektori uzoraka nisu iste duljine te nemaju zajedničkih podataka. U praki ovaj problem se rješava tako da se sve značajke napišu kao uređena n-torka te se kreiraju vektori uzoraka koji predstavljaju pojavljivanje određene značajke. U konkretnom slučaju radi se o binarnom vektoru gdje 1 predstavlja da se značajka pojavila u uzorku, a 0 da nije.

$$(boja, lim, intenzitet, sunce, korozija)$$

$$v_1 = (1, 1, 1, 1, 0)$$

$$v_2 = (0, 0, 0, 0, 1)$$

Navedenim postupkom dobili smo jednistveno zapisane sve rečenice kao vektor on pet komponenti. Dakle duljina je ista i skup značajki koje promaramo je isti za sve uzorke. Isti postupak provedemo za ostale rečenice, izvučemo iz njih najbitnije značajke te iz spojimo sa već uređenom petorkom i dobijemo traženu uređenu 11-torku koja predstavlja sve rečenice iz komentara.

($boja, lim, intenzitet, sunce, korozija, interjer, oprema, dostupan, varijante, cijena, trziste$)

Svaku rečenicu prikažemo kao uzorak

- Boja lima je idalne kvalitete, ne gubi intenzitet na suncu
  uzorak = $(1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0)$
- Korozija je značajno zabilježena tek na modelima starijima od 10 godina
  uzorak = $(0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0)$
- Izuzetno kvalitetan interijer sa opremom kao što je električni paket, sustav protiv proklizavanja, klima i ostala oprema.
  uzorak=$(0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0)$
- Automobil je dostupan u tri varijante kao sportback, limuzina, karavan. Cijena mu se na tržištu kreće oko 10000 eura.
  uzorak=$(0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1)$

Ovako kompaktno zapisani podaci zauzimaju puno manje memorije, lakši su za pohranu ali i za analizu. Težine koje smo dodali u funkciju sličnosti u ovom trenutku ima bitnu ulogu. Na primjer ako nam se pojavi riječ lim ili korozija sasvim je očito da se radi o kvaliteti lima te ako veću pažnu posvetimo tim značajkama veća je vjerojatnost da na kraju pripadaju ispravnom klasteru. Analogno ako promatramo riječ tržište, jedina prihvatljiva iterpretacija je da se ta riječ pojavljuje samo u rečenicama koje govore o kretanju na tržištu te i ovoj značajki treba pridjeliti dodatnu važnost. S druge strane riječ sunce može se pojaviti u raznim kontekstima, njegova je uloga bitna ali nikako ne smije biti krucijalan u odabiru klase te mu se važnost treba restringirati.

Poglavlje 2

Evolucijske metode

2.1 Meta-heuristike

Kada govorimo o teškim problemima, zapravo govorimo o problemima čiji skup rješenja eksponencijalo raste i čak ni današnja super brza računala ne mogu ispitati sva moguća rješenja kako bi pronašli ono najbolje. Problemi za koje ne možemo u realnom vremenu pronaći egzaktno rješenje nazivaju se NP-teški problemi. Ovako neformalna definicija ostavlja mnoga pitanja kao na primjer: što je to rješenje u realnom vremenu? U ovom radu nećemo promatrati složenost algoritama, kao ni konstrukciju modela za izračunavanje istih, ali zbog potpunosti i razumijevanja težine problema grupiranja i opravdanosti uvođenja heuristika definirat ćemo samo nužne pojmove.

Definicija 2.1.1. *Prihvatljivo vrijeme je vrijeme koje može proći a da rješenje još uvijek bude aktualno.*

Definicija 2.1.2. *Rješenje u realnom vremenu je rješenje čiji se rezultat može upotrijebiti u prihvatljivom vremenu od trenutka pokretanja algoritma koji traži rješenje.*

Definicija 2.1.3. *NP-teški problemi su problemi za koje ne možemo izračunati egzaktno rješenje u realnom vremenu.*

Promotrimo sada problem grupiranja, za koji pokušavamo pronaći i analizirati dobar algoritam. Poblem je trivijalno rješiv iscrpnom pretragom. Iscrpna pretraga je proces ispitivanja svih mogućih rješenja danog problema. U našem slučaju to bi značilo ispitivanje svih mogućih kombinacija n uzoraka u k klastera. Predpostavimo da i mamo samo dva moguća klastera i n uzoraka. Za svaki uzorak možemo odlučiti na dva načina, tako da ga pridružimo prvom ili drugom klasteru. Isti postupak možemo procesti za svaki uzorak pa

ugrubo imamo sljedeću jednakost

$$\underbrace{2 \cdot 2 \cdot 2 \cdot \cdots \cdot 2}_{\text{n puta}} = 2^n$$

Iako je navedena ocjena za broj mogućih rješenja jako gruba ipak govori o redu veličina danog problema. Već za $k = 2$ problem grupiranja je težak problem jer imamo 2^n mogućnosti gdje smjestiti uzorke. Upravo iz navedenih razloga prilikom rješavanja problema grupiranja nikada se ne upotrebljava iscrpna pretraga osim možda u nekim slučajevima izuzetno malog skupa podataka. Osim iterativnih metoda navedenih u poglavlju 1, kada tražimo optimalno ili suboptimalno rješenje problemu je moguće pristupiti i na heuristički način. Heuristike ne daju nikakvu garanciju na kvalitetu rješenja, za razliku od iterativnih metoda koje u svakom koraku poboljšaju već postojeće rješenje. Međutim kod heurisika je prednost što lakše izbjegavaju lokalne minimume i na pseudo-slučajan način sa skokovima traže rješenje u prostoru svih mogućih rješenja.

Definicija 2.1.4. *Algoritmi koji pronalaze rješenje koje je zadovoljavajuće dobro, te imaju polinomnu složenost izvršavanja s obzirom na ulazni skup podataka nazivaju se heuristike.Heuristike ne daju nikakvu garanciju na kvalitetu rješenja*

Zadovoljavajuće dobro rješenje je rješenje koje zadovoljava unaprijed zadanu točnost. Iako u definiciji navodimo kako heuristike ne daju nikaku garanciju oko kvalitete rješenja u praksi su se pokazale kao vrlo korisne i stabline. Stablinost se odnosi upravo na rješenje, te u praksi gotovo uvijek daju rješenje koje je jako blizu optimalnog. Garancija oko kvalitete rješenja nije moguća upravo zbog pseudo-slučajnih elemenata koji su sastavni dio heurističkih algoritama. Naime, prilikom analize algoritma koriste si se tok algoritma kao glavni alat za dokaz korektnosti, dok kod heuristika nema glavnog toka u klasičnom smislu jer pseudo-slučajnim elementima izvodimo unaprijed nepredvidive skokove.

Definicija 2.1.5. *Meta-heurisike su skup algoritamskih koncepata koji služe za rješavanje odrđenog skupa sličnih problema. Za meta-heuristike su poznati neki rezultati korektnosti te su takvi algoritmi generalizirani za širu upotrebu.*

Za meta-heuristike možemo reći da su specijalizirane heuristike. Dva su osnovna pristupa rješavanju problema pomoću heurisika.

- Konstrukcijski algoritmi
- Algoritmi lokalne pretrage

Primjenom konstrukcijskih algoritama rješenje se gradi dio po dio, najčešće bez mogućnosti povratka sve dok se ne izgradi konačno rješenje. Kod algoritama lokalne pretrage izabere

se jedno rješenje i ono se iterativno poboljšava dok ne zadovolji uvjete. Ideja je da se baziramo oko nekog relativno dobrog rješenja i pretražujemo njemu slična kako bi našli najbolje ili dovoljno dobro rješenje. Heuristički postupak ukratko se može opisati u nekoliko koraka.

- Slučajno izaberi neko rješenje
- Ako je rješenje jako loše ponovo biraj
- Ako je rješenje skoro dobro poboljšaj
- Ako je rješenje dobro stani

Iz ovih nekoliko koraka vidimo kako se heuristike ponašaju na vrlo jednostavan način. Ako je neko rješenje jako loše ono ne može opstati jer će njegovo mjesto zauzeti neko bolje rješenje. Ako je rješenje preživilo prvu selekciju onda od njega pokušamo napraviti nešto bolje rješenje. Navedeni postupak se zapravo svakodnevno javlja u prirodi kao selekcijski postupak jedinki. Loše jedinke izumiru, dobe ostaju, razmnožavaju se i tako nastaju sve bolje i kvalitetnije jedinke. Upravo navedena inspiracija bit će vodilja kroz daljnji izbor algoritama za heuristike opisanih u sljedećoj cjelini.

2.2 Prirodom inspirirani algoritmi

Glavna pitanje prilikom kreiranja heurističkog rješenja je kako provesti postupak selekcije dobrih, kako stvoriti nova rješenja ako je neko rješenje dovoljno dobro, te konačno kako reprezentirati podatke. Rješenja na upravo navedene probleme možemo pronaći u prirodi. Kada govorimo o algoritmima inspiriranim prirodnim procesima mislimo na tri osnovna tipa algoritama. Konkretno radi se o evolucijskim, procesnim i algoritmima rojeva. Procesni algoritmi su algoritmi koji simuliraju procese u prirodi. Jedan od takvih procesa je kaljenje metala. Metal se zagrijava na određenu temparaturu te se postepeno hladi i tokom procesa hlađenja mjenja se struktura metala i tako se stvara poželjna struktura legure. Navedeni tip algoritama opisuje algoritme lokalne pretrage. Postoje još mnogi algoritmi ovog tipa [liter], ali upravo navedeni je najpoznatiji predstavnik. Što se algoritama rojeva tiče to su algoritmi koji simuliraju neki proces uglavnom mrava i pčela. Kako su obe navedene vrste poznate kao jako radno orjentiranje njihovi procesi se vrlo često odvijaju iterativno. Tako naprimjer postoji mravlji algoritam koji simulira put kojim mravi putuju tokom radne akcije. Naime, mravi kada prolaze nekim putem ostavljaju za sobom trag feromona tako da dugi mravi mogu ići istim putem. Što više mravi prođe određenim putem to je miris feromona jači te tako privlači i druge mrave na taj isti put. Ako rješenje optimizacijskog rješenja reprezentiramo kao jedan put kojima mravi mogu putovati možemo

kreirati dobar algoritam koji rješava dani problem. Osim mravljeg algoritma ističe se algoritam roja pčela[liter]. Evolucijski algoritmi su oni koji su u ovom radu puno zanimljiviji i algoritmi za grupiranje podataka nerjetko imaju elemente upravo ove metode. Evolucijski algoritmi simuliraju životni ciklus neke populacije. Razvoj algoritma može se prikazati ako životni ciklus jedinke neke populacije. Pređimo tako na slučaj kada su rješenja skoro dobra. Pretpostavimo da svako rješenje predstavlja jednu jedinku, to znači da u prirodi imamo jedinku koja je spremna opstati. Nakon što selektiramo sve jedinke koje su sposobne opstatit nužan je proces razmnožavanja. Prilikom razmnožavanja dobriz jediniki, to jest onih koje su opstale, nastaju nove jedinke sa dobrim genetskim naterijalom. Upravo u ovom evolucijskom korako, ako su jedinke rješenja, vidimo analogan proces traženju poboljšanja rješenja. Također postoji mogućnost da neke jedike prilikom nastanka mutiraju i tako znatno promjene svoje svojstvo. Ova mogućnost u rješavanju optimizacijskog problema ima dva značenja. Prvo značenje je pozitivno i pomaže kako bi izašli iz lokalnog optimuma. Naime, kada jednom imamo dva skoro dobra rješenja tada smo vrlo vjerojatno u nekom lokalnom optimumu i daljnim postupkom razmnožavanja samo onih koji su jako dobri sve više se približavamo sličnim rješenjima iz potprostora lokalnog optimuma. Mutacija se nameće kao logično rješenje bjega iz lokalnog optimuma. Mutiranjem jedinka mjenja svojstva te tako postaje ili puno gora ili barem jednako dobra kao neko već postignuto rješenja, ali ključna stavka je da mutirano rješenje vrlo često nije iz područja lokalnog optimuma gdje je nastalo. Kod evolucijskih algoritama ističe se nekoliko ključnih koraka.

- Svori inicijalnu populaciju.
- Izaberi najbolje iz populacije, ostale obriši.
- Kako bi postigi ravnotežu razmnožavaj preostale jedinke do popunjenja populacije.
- Mutiraj neke jedinke
- Ponovi postupak selekcije
- Ako neka jedinka iz trenutne populacije zadovoljava tražene uvijete stani.

Najpoznatiji predstavnici navedene metode su *genetski algoritam-GA*, *umjetni imunološki algoritam-IA*[liter] i *algoritam diferencijske evolucije-DE*[liter]. Ostalo je objasniti što su to jedinke, kako rješenja predstavljano kao jedinke, te kako ćemo obavljati selekciju unutar populacije. Navedena pitanja rješavaju se slično za sve algoritme, ali zbog posebnosti svakog algoritma postoje i specifičnosti tako da generalitiranog postupka za prikaz jedinki i selekciju nema. Kako ćemo u ovom radu posebnu pažnju posvetiti genetskim algoritmima za njih ćemo prikazati postupak kreiranja rješenja, te kako provesti sve procese evolucijskog razvoja nad tako kreiranim rješenjima.

2.3 Genetski algoritam

Ideja za razvoj algoritma temelji se na Darwinovoj teoriji o postanku vrste[liter]. Teorija se temelji na pet postavki:

1. plodnost vrste - potomaka uvijek ima dovoljno
2. održivost - veličina populacije je uvijek stalna
3. količina hrane je ograničena
4. prilikom procesa razmnožavanja nema potomaka jednakih roditeljima
5. najveći dio varijacije prenosi se nasljeđem

Postoji puno izvedbi genetskog algoritma, ali generalna ideja je sljedeća. Algoritam radi s populacijom jedinki. Svaka jedinka jedno je moguće rješenje danog problema. U našem slučaju svaka jedinka je jedna particija rješenja. Jedinke u genetskom algoritmu često se nazivaju kromosomi. U kasnijim razmatranjima intuitivnije je govoriti o križanju i mutaciji kromosoma. Svakoj jedinki računa se funkcija dobrote. U našem slučaju funkcija dobrote je suma razlika svih jedinki u pojedniom klasteru. Jednike sa manjom vrijednosti dane funkcije su bolje jednike. Nakon računanja dobrote svih jedinki izvršava se postupak selekcije. Selekcija je postupak izbora jedinki koje imaju ulogu roditelja u novoj populaciji. Jedinke djeca nastaju operatorom križanja i mutacije. Ciklus se nastavlja zadani broj puta. Iz navedenog opisa možemo izdvojiti četiri osnovne operacije koje se izvršavju u svakom ciklusu: *selekcija*, *funkcija dobrote*, *križanje*, *mutacija*.

Poglavlje 3

Poznati algoritmi i analiza

3.1 Alg 1

3.2 Alg 2

3.3 Alg 3

Poglavlje 4

Tehnike za paralelizaciju algoritama

4.1 Osnovni pojmovi MPI tehnologije

4.2 Topologije

4.3 Prednosti paralelizacije i cijena komunikacije

Poglavlje 5

Konstrukcija paralelnih algoritama za grupiranje

5.1 Algoritam 1 heurisika

Opis

Analiza

5.2 Algoritam 2 iterativno

Opis

Analiza

5.3 Algoritam 3 hibrid

Opis

Analiza

Poglavlje 6

Ostale moderne metode

6.1 Programiranje na grafičkim karticama

6.2 MapReduce metoda

Bibliografija

[1] D. E. Dutkay, D. Han, Q. Sun i E. Weber, *Hearing the Hausdorff dimension*, (2009), http://arxiv.org/abs/0910.5433.

Sažetak

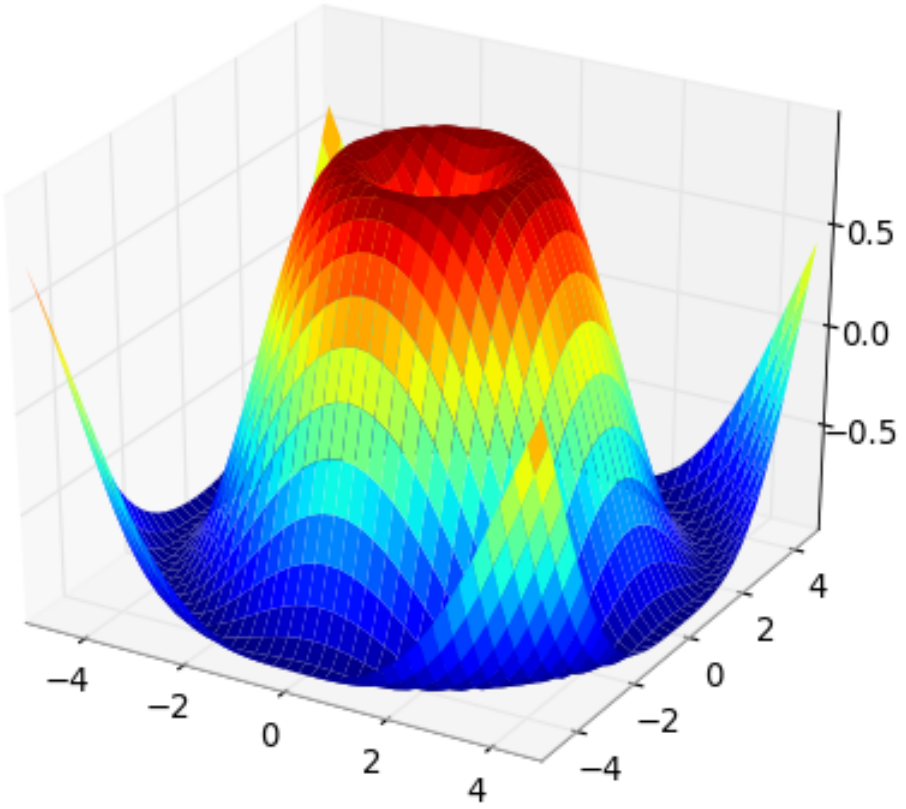
Ukratko ...

Summary

In this ...

Životopis

Na slici 1.1.7 se nalazi 3D graf neke funkcije.



Slika 6.1: Druga slika

kao i jedna vrlo kompliciran formula koja slijedi iz (**??**)

$$\sum_{i=1}^{\infty} A_{x_{1}} \times A_{\alpha_{2}} \oslash \iint_{\Omega} x^{2} \ddagger \limsup_{n \in \mathbb{N}} \frac{\alpha + \theta + \gamma}{n^{\omega}} \text{ je u stvari } \bigcup_{r \in \mathbb{Q}}^{\left(+ \right)} \overline{\Xi_{i} \underset{\substack{j \in \mathbb{C} \\ j \ni i \mathbb{Q}}}{\Theta} \Upsilon^{k^{j}} \underset{*}{\Psi} \hbar |_{\{\alpha\}}}.$$