# Title

Firstname1 Lastname1[1] and Firstname2 Lastname2[2]

[1]Department1, Institution1, City1, State1 Zip1, Country1
[2]Department1, Institution1, City2, State2 Zip2, Country2

**Sažetak**

Polu-nadzirani algoritmi za klasteriranje nastoje poboljšati rezultate klasteriranja koristeći ograničeno nadziranje. Nadziranje je inače dano u obliku parova koji predstavljaju ograničenja; takva ograničenja su prirodna za grafove, no ipak je većina algoritama za polu-nadzirano klasteriranje smišljena za podatke u obliku vektora. U ovom radu ujedinjujemo dva pristupa od kojih se jedan temelji na vektorskom obliku, a drugi na grafovskom. Prvo pokazujemo da se nedavno predložena funkcija cilja za polu-nadzirano klasteriranje temeljena na Skrivenim Markovljevim Slučajnim Poljima (SMSP; Hidden Markov Random Fields, eng.), zajedno sa kvadratom euklidske udaljenosti i određenom klasom kaznenih funkcija ograničenja, može izraziti kao specijalan slučaj funkcije cilja težinskog algoritma jezgrena k-sredina (weighted kernel k-means, eng.). Nedavno iznesena teoretska povezanost između težinskog algoritma jezgrena k-sredina i nekoliko ciljeva klasteriranja grafova omogućava nam da izvršimo polu-nadzirano klasteriranje podataka zadanim u vektorskom obliku, ili u grafovskom obliku. Za podatke u obliku grafa, ovaj rezultat vodi do algoritama za optimizaciju nekoliko novih polu-nadzirajućih funkcija cilja za klasteriranje grafa. Za podatke u obliku vektora, jezgreni pristup nam također omogućava pronalazak klastera sa nelinearnim granicama u prostoru ulaznih podataka. Nadalje, pokazujemo da se nedavni rad na spektralnom učenju može promatrati kao specijalan slučaj naše formulacije. Empirijski pokazujemo da naš algoritam može nadmašiti polu-nadzirano algoritme koji su bili najmoderniji 2008. godine, u doba kada je izašao rad na kojem temeljimo naš seminar, na skupovima podataka zadanim u vektorskom obliku, i u grafovskom obliku.

## 1 Uvod

Fig **??** Polu-nadzirani algoritmi za klasteriranje su nedavno dobili značajnu pozornost u zajednicama za strojno učenje i rudarenje podataka. U tradicionalnim algoritmima za klasteriranje, samo su se neoznačeni podaci koristili za generiranje klastera; u polu-nadziranom klasteriranju, cilj je iskoristiti početne informacije o klasterima u algoritmu kako bi poboljšali rezultate klasteriranja. Sličan problem je polu-nadzirana klasifikacija, koja razmatra kako se neoznačeni podaci mogu iskoristiti za poboljšanje izvođenja klasifikacije na označenim podacima. Nekoliko je radova u zadnje vrijeme istraživalo problem polu-nadziranog klasteriranja. Istraživanja na polu-nadziranom klasteriranju su dosad razmatrala nadziranje u obliku označenih točaka ili ograničenja.

Kao što je uobičajeno za većinu polu-nadziranih algoritama za klasteriranje, mi pretpostavljamo u ovom seminaru da imamo ograničenja zadana u obliku pozitivnih veza (parove točaka koje bi se trebale nalaziti istom klasteru) i negativnih veza (parovi točaka koje bi se trebali nalaziti u različitim klasterima), zajedno sa ulazom. Takva ograničenja se prirodno pojavljuju u mnogim područjima, npr. skup podataka Baza Podataka Interakcijskih Proteina (BPIP) u biologiji sadrži informacije o proteinima koji se zajedno pojavljuju u procesima, što se može vidjeti ograničenje u obliku pozitivne veze tijekom klasteriranja gena. Ograničenja u ovom obliku su također prirodna u kontekstu problema klasteriranja grafa (poznatom kao particioniranje grafa ili particioniranje vrhova), gdje rubovi u grafu predstavljaju veze u obliku parova.

Nedavno je predložen vjerojatnosni okvir za polu-nadzirano klasteriranje sa ograničenjima u obliku parova

na temelju Skrivenih Markovljevih Slučajnih Polja. SMSP okvir predlaže opću funkciju cilja polu-nadziranog klasteriranja temeljenu na maksimiziranju zajedničke vjerojatnosti podataka i ograničenja u SMSP modelu, kao i iterativni algoritam za optimizaciju funkcije cilja, koji će biti sličan k-sredina algoritmu. Međutim, SMSP-KSREDINA može klasterirati samo ulazne podatke u obliku vektora. Dok se velik dio radova na polu-nadziranom klasteriranju fokusirao na ulazne podatke u obliku vektora, vrlo malo se radova bavilo istraživanjem algoritama za polu-nadzirano klasteriranje grafova. U ovom radu objedinjujemo polu-nadzirano klasteriranje na temelju vektora i grafova koristeći jezgreni pristup; konkretno, naš glavni doprinos u ovom radu je sljedeći:

- Pokazujemo da je funkcija cilja SMSP polu-nadziranog klasteriranja sa kvadratom euklidske udaljenosti, i težinskim funkcijama kazne sa sumama veličina klastera, specijalan slučaj funkcije cilja težinskog algoritma jezgrene k-sredine. Sa ulaznim podacima u obliku vektora i ograničenjima u obliku parova, pokazujemo kako izgraditi jezgru tako da pokretanje algoritma jezgrena k-sredina rezultira u monotonom padu ove funkcije cilja polu-nadziranog klasteriranja pri svakoj iteraciji algoritma jezgrena k-sredina.

- Težinski algoritam k-sredine se može koristiti za monotonu optimizaciju široke klase funkcija cilja klasteriranja grafova poput minimizacije normaliziranog reza. Posljedično, naš pristup se može poopćiti kako bi se optimizirao određeni broj funkcija cilja polu-nadziranog klasteriranja grafa za koje je nadziranje na temelju ograničenja prirodnije, uključujući polu-nadzirani normalizirani rez, polu-nadzirani razmjerni rez i polu-nadzirane funkcije cilja razmjerne povezanosti. Ova ekvivalentnost nam daje novi algoritam za klasteriranje PN-JEZGRENA-KSREDINA (SS-KERNEL-KMEANS, eng.) koji može klasterirati podatke dane u obliku vektora ili grafa, zajedno s nadzorom u obliku ograničenja.

- Za podatke u obliku vektora imamo mogućnost primijeniti jezgrenu funkciju kojaa omogućuje algoritmu PN-JEZGRENA-KSREDINA otkrivanje klastera s nelinearnim granicama u prostoru ulaznih vrijednosti.

- Pokazujemo kako se prethodno predloženi algoritam spektralnog učenja može promatrati kao specijalan slučaj unutar naših okvira. Algoritam spektralnog učenja se može promatrati kao optimizacija funkcije cilja polu-nadziranog klasteriranja; konkretno, on optimizira opuštanje polu-nadziranog razmjernog reza

- Empirijski dokazujemo da algoritam PN-JEZGRENA-KSREDINA nadmašuje rezultat algoritma SMSP-KSREDINA i algoritma spektralnog učenja na nekoliko skupova podataka u vektorskom i grafovskom obliku, uključujući podatke o rukopisnim znamenkama.

# 2 Pozadina i srodni radovi

U ovom odjeljku, pružamo potrebnu podlogu za naš predloženi algoritam polu-nadziranog jezgrenog klasteriranja PN-JEZGRENA-KSREDINA, i opisujemo srodna istraživanja.

$$\Delta = \sum_{i=1}^{N} w_i (x_i - \bar{x})^2 \tag{1}$$

## 2.1 Klasteriranje grafa i algoritam jezgrena k-sredina

Kod klasteriranja grafa, pretpostavlja se da je ulaz zadan u obliku grafa $G = (\mathcal{V}, \mathcal{E}, A)$, gdje je $\mathcal{V}$ skup vrhova, $\mathcal{E}$ skup bridova i $A$ matrica susjedstva grafa G. $A_{ij}$ predstavlja težinu brida između vrhova $i$ i $j$.

Neka je $povezanost(\mathcal{A}, \mathcal{B}) = \sum_{i \in \mathcal{A}, j \in \mathcal{B}} \mathcal{A}_{ij}$, gdje su $\mathcal{A}$ i $\mathcal{B}$ podskupovi od $\mathcal{V}$. Nadalje, neka je $stupanj(\mathcal{A}) = povezanost(\mathcal{A}, \mathcal{V})$. Tražimo k-članu particiju kojom bismo minimizirali određenu funkciju cilja. Tablica 1 nam daje listu čestih funkcija cilja klasteriranja grafa. Razmjerna povezanost nastoji maksimizirati sličnost točaka unutar klastera, normalizirajući doprinos svakog klastera funkciji njegovom veličinom, kako bi se

Tablica 1: Primjeri funkcija cilja klasteriranja grafova

| Ime | Funkcija cilja |
|---|---|
| Razmjerna povezanost | $\max_{\mathcal{V}_1,\ldots,\mathcal{V}_k} \sum_{c=1}^{k} \frac{povezanost(\mathcal{V}_c,\mathcal{V}_c)}{\lvert\mathcal{V}_c\rvert}$ |
| Razmjerni rez | $\min_{\mathcal{V}_1,\ldots,\mathcal{V}_k} \sum_{c=1}^{k} \frac{povezanost(\mathcal{V}_c,\mathcal{V}\setminus\mathcal{V}_c)}{\lvert\mathcal{V}_c\rvert}$ |
| Normalizirani rez | $\min_{\mathcal{V}_1,\ldots,\mathcal{V}_k} \sum_{c=1}^{k} \frac{povezanost(\mathcal{V}_c,\mathcal{V}\setminus\mathcal{V}_c)}{stupanj(\mathcal{V}_c)}$ |

ujednačila veličine klastera. S druge strane, razmjerni rez nastoji minimizirati ukupnu težinu bridova koji prelaze granice klastera - ovo je opet normalizirano veličinom klastera, kako bi se potaknula ujednačenost veličina klastera. Kriterij normaliziranog reza koristi istu funkciju cilja kao i razmjerni rez, ali se normalizira sa ukupnim stupnjem svakog klastera (sumom stupnjeva svih vrhova u klasteru) - ovo potiče klastere da imaju slične stupnjeve.

Da bismo uspostavili vezu između klasteriranja grafa i klasteriranja na temelju vektora, uvest ćemo funkciju cilja težinskog algoritma jezgrene k-sredine. S danim skupom podataka u obliku vektora $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{n}$ gdje je $\mathbf{x}_i \in \mathbb{R}$, cilj težinskog algoritma jezgrene k-sredine je naći k-članu particiju $\{\pi_c\}_{c=1}^{k}$ skupa podataka (gdje $\pi_c$ predstavlja $c-ti$ klaster) tako da postignemo minimum sljedeće funkcije cilja:

$$\mathcal{J}(\{\pi_c\}_{c=1}^{k}) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \alpha_i \|\phi(\mathbf{x}_i) - \mathbf{m}_c\|^2 \quad \text{gdje je} \quad \mathbf{m}_c = \frac{\sum_{\mathbf{x}_i \in \pi_c} \alpha_i \phi(\mathbf{x}_i)}{\sum_{\mathbf{x}_i \in \pi_c} \alpha_i}. \tag{2}$$

Svaki vektor $\mathbf{x}_i$ ima unaprijed zadanu ne-negativnu težinu $\alpha_i$, a $\phi$ je funkcija preslikavanja vektora iz skupa $\mathcal{X}$ u (općenito) višedimenzionalni prostor. Ako postavimo sve težine na vrijednost jedinice i $\phi$ je identiteta, to se svodi na standardni algoritam k-sredina.

Ako proširimo izraz $\|\phi(\mathbf{x}_i) - \mathbf{m}_c\|^2$ iz funkcije $\mathcal{J}$, dobit ćemo sljedeće:

$$\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_i) - \frac{2\sum_{\mathbf{x}_j \in \pi_c} \alpha_j \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)}{\sum_{\mathbf{x}_j \in \pi_c} \alpha_j} + \frac{2\sum_{\mathbf{x}_j, \mathbf{x}_l \in \pi_c} \alpha_j \alpha_l \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_l)}{(\sum_{\mathbf{x}_j \in \pi_c} \alpha_j)^2}. \tag{3}$$

Primijetimo da je sav izračun koji uključuje vektore iz skupa podataka zadan u obliku skalarnog produkta. Zbog toga možemo primijeniti *jezgrenitrik*: ako možemo učinkovito izračunati skalarni produkt $K_{ij} = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ između $\phi(\mathbf{x}_i)$ i $\phi(\mathbf{x}_j)$, moći ćemo izračunati udaljenost između točaka u prostoru kodomene funkcije $\phi$ bez da eksplicitno znamo funkciju $\phi$. Pretpostavljamo jezgrena matrica skalarnih produkta $K$ argument ulaza; lako se može pokazati da bilo koja pozitivna semidefinitna matrica $K$ može smatrati jezgrenom matricom.

Koristeći jezgrenu matricu $K$, izračun udaljenosti $d(\mathbf{x}_i, \mathbf{m}_c) = \|\phi(\mathbf{x}_i) - \mathbf{m}_c\|^2$ se može napisati kao:

$$K_{ii} - \frac{2\sum_{\mathbf{x}_j \in \pi_c} \alpha_j K_{ij}}{\sum_{\mathbf{x}_j \in \pi_c} \alpha_j} + \frac{2\sum_{\mathbf{x}_j, \mathbf{x}_l \in \pi_c} \alpha_j \alpha_l K_{jl}}{(\sum_{\mathbf{x}_j \in \pi_c} \alpha_j)^2}. \tag{4}$$

Kao rezultat toga, možemo izvesti algoritam analogan algoritmu k-sredina kako bismo dobili monotoni pad funkcije cilja $\mathcal{J}$ bez znanja o funkciji $\phi$. Algoritam 1 nam prikazuje pseudokod osnovni težinski algoritam jezgrena k-sredina, koji je identičan standardnom algoritmu k-sredina osim činjenice da se udaljenosti izračunavaju pomoću jezgrene matrice. Primijetite da za razliku od običnog algoritma k-sredina, ovaj algoritam ne ažurira eksplicitno centroide $m_c$.

## 2.2 Polu-nadzirano klasteriranje

Često pri klasteriranju imamo nešto prethodnog znanja o strukturi klastera. Kao što smo prethodno spomenuli, u ovom seminaru pretpostavljamo da to znanje dolazi u obliku pozitivnih i negativnih veza između točaka. Takva ograničenja su prirodna za grafove, s obzirom da su pozitivne veze eksplicitno izražene pomoću bridova u grafu.

**Algorithm 1:** Obični težinski algoritam jezgrene k-sredine

SMSP-KSREDINA($K, k, t_{max}, \alpha, \{\pi_c^{(0)}\}_{c=1}^k$)

**Ulaz**: $K$: jezgrena matrica, $k$: broj klastera, $t_{max}$: proizvoljan maksimalan broj iteracija, $\alpha$: težinski vektor, $\{\pi_c^{(0)}\}_{c=1}^k$): proizvoljni početni klasteri

**Izlaz**: $\{\pi_c^{(0)}\}_{c=1}^k$): konačna particija točaka

1. Inicijaliziraj $k$ klastera iz $\{\pi_c^{(0)}\}_{c=1}^k$), ako je dan u ulazu, inače nasumično.
2. Postavi $t = 0$.
3. Za svaku točku $\mathbf{x}_i$ i svaki klaster $c$, izračunaj $d(\mathbf{x}_i, \mathbf{m}_c)$ kao u (1).
4. Pronađi $c^*(\mathbf{x}_i) = \arg\min_c d(\mathbf{x}_i, \mathbf{m}_c)$, proizvoljno odabirući jedno od rješenja ako ih postoji više.
5. Izračunaj nove klastere sa:
$$\pi_c^{(t+1)} = \{\mathbf{x}_i \ : \ c^*(\mathbf{x}_i) = c\}. \tag{5}$$
6. Ako nema konvergencije ili $t_{max} > t$, postavi $t = t + 1$ i odi na Korak 3; Inače, stani i vrati konačne klastere $\{\pi_c^{(0)}\}_{c=1}^k$).

Tablica 2: Popularne funkcije cilja i odgovarajuće težine i jezgre sa danom matricom susjedstva A

| Ime | Težine vrhova | Jezgra |
|---|---|---|
| Razmjerna povezanost | 1 za sve čvorove | $K = \sigma I + A$ |
| Razmjerni rez | 1 za sve čvorove | $K = \sigma I - L$ |
| Normalizirani rez | Stupanj vrhova | $K = \sigma D + D^{-1}AD^{-1}$ |

## 2.3 SMSP model

Sada ćemo ukratko opisati nedavno predloženu funkciju cilja za polu-nadzirano klasteriranje podataka u obliku vektora, koju ćemo koristiti u našoj formulaciji. Basu i dr. (2004b) su predložili okvir za polu-nadzirano klasteriranje temeljeno na Skrivenim Markovljevim Slučajnim Poljima (SMSP). Odabirom kvadrata euklidske udaljenosti za mjeru distorzije klastera i poopćenog Potts-ovog potencijala (Kleinberg i Tardos 1999) za potencijal narušavanja ograničenja, funkcija cilja polu-nadziranog klasteriranja se može izraziti kao:

$$\mathcal{J}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\phi(\mathbf{x}_i) - \mathbf{m}_c\|^2 + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i \neq l_j}} w_{ij} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} w_{ij}, \tag{6}$$

gdje je $\mathcal{M}$ skup pozitivnih veza između vektora, $\mathcal{C}$ skup negativnih veza između vektora, $w_{ij}$ je iznos kazne za narušavanje ograničenja između vektora $x_i$ i $x_j$, a $l_i$ se odnosi na oznaku klastera kojem pripada vektor $x_i$. Prvi izraz u sumi funkcije $\mathcal{J}$ je standardna funkcija cilja algoritma k-sredina, drugi izraz je kaznena funkcija za narušavanje pozitivnih veza, a treći izraz je kaznena funkcija za narušavanje negativnih veza. Algoritam SMSP-KSREDINA izrađen za minimiziranje ove funkcije cilja koristi pristup iterativnog premještanja poput algoritma k-sredina. Jedan od nedostataka algoritma SMSP-KSREDINA je taj što, u koraku pridruživanja točaka klasterima sa trenutnim centroidima klastera, poredak u kojem se točke pohlepno razmještaju po klasterima određuje nove klastere, tj. različiti poreci proizvode različite klastere. Naći optimalan poredak u ovom slučaju je računski težak problem. Kako ćemo vidjeti u trećem poglavlju, naš predloženi algoritam prirodno izbjegava takav problem poretka.

## 2.4 Spektralno učenje

Nedavno, spektralne metode su postale sve popularnije za klasteriranje. Ovi algoritmi klasteriraju podatke u obliku grafa. Jedan spektralni pristup polu-nadziranom klasteriranju je algoritam SPEKTRALNO-UČENJE (SPECTRAL-LEARNING, eng.) Kamvara i dr. (2003):

- Izgradimo matricu susjedstva $A$: pretpostavlja se da su vrijednosti matrice $A$ normalizirani između 0 i 1, gdje $A_{ij}$ predstavlja stupanj povezanosti između točaka $i$ i $j$.
- Za sve točke $i$, $j$ koje su pozitivno povezane, neka je $A_{ij} = 1$ (maksimalna povezanost); za sve točke $i$, $j$ koje su negativno povezane, neka je $A_{ij} = 0$ (minimalna povezanost).
- Ponovno normaliziraj matricu koristeći aditivnu normalizaciju (Kamvar i dr. 2003): $N = \frac{1}{d_{max}}(A + d_{max}I - D)$.$N$ sad predstavlja normaliziran Markovljev tranzitivan proces.
- Uzmimo gornjih $k$ svojstvenih vektora matrice $A$ za stupce matrice $V$, i klasterirajmo redove od $V$. Ako $N$ ima $k$ po dijelovima konstantnih svojstvenih vektora, onda $N$ ima $k$ jakih klika, tako da klasteriranje svojstvenih vektora zapravo nastoji naći ovih $k$ jako povezanih dijelova ili klastera.

U gornjem algoritmu, $d_{max}$ je maksimalna suma po redovima od matrice $A$ i $D$ je matrica stupnjeva (stupnjeva vrhova u grafu). Primijetimo da je aditivna normalizacija jednaka $I - \frac{1}{d_{max}}L$, što je ekvivalentno $d_{max}I - L$ do na skalarni faktor, gdje je $L = D - A$ Laplaceova matrica grafa od A. Nadalje, gornji svojstveni vektori od $d_{max}I - L$ su isti kao donji svojstveni vektor od $L$. U prezentaciji Kamvara i dr. (2003), algoritam SPEKTRALNO-UČENJE nema eksplicitnu funkciju cilja. No ipak, u odjeljku 4.3. pokazat ćemo da se ovaj algoritam može promatrati kao specijalni slučaj našeg ujedinjenog okvira polu-nadziranog klasteriranja. Primijetimo da taj algoritam treba $O(kn)$ memorije kako bi pohranio $k$ svojstvenih vektora za $n$ točaka, što može biti znatno opterećenje za velike grafove.

# 3 Jezgreni pristup polu-nadziranog klasteriranja za algoritam SMSP-KSREDINA

U ovom odjeljku pokazat ćemo vezu između funkcije cilja polunadziranog klasteriranja za algoritam SMSP-KSREDINA i funkcije cilja jezgrene k-sredine. To nas vodi do jezgrenog algoritma za optimizaciju funkcije cilja algoritma SMSP-KSREDINA sa kvadratom euklidske udaljenosti i kaznama ponderiranim veličinom klastera.

## 3.1 Izgradnja jezgre

Prisjetimo se funkcije cilja za polu-nadzirano klasteriranje u algoritmu SMSP iz (2), koristeći kvadrat euklidske udaljenosti za mjeru distorzije klastera i poopćeni Potts-ov potencijal za kaznu narušavanja ograničenja. Zamijenimo ovu kaznenu funkciju: umjesto dodavanja kaznenog izraza za narušavanje ograničenja pozitivnih veza ako se dvije točke nalaze u različitim klastera, *nagradit* ćemo zadovoljenje ograničenja ako se točke nalaze u istom klasteru oduzimanjem odgovarajućeg kaznenog izraza u funkciji. Ako je zbroj svih težina za ograničenje pozitivnih veza konstantan, onda je ovo ekvivalentno originalnoj funkciji cilja do na aditivnu konstantnu. Stoga minimiziranje $\mathcal{J}(\{\pi_c\}_{c=1}^k)$ je ekvivalentno minimizaciji:

$$\sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\phi(\mathbf{x}_i) - \mathbf{m}_c\|^2 - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} w_{ij} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} w_{ij}, \tag{7}$$

Također uvodimo pojam težinskog kažnjavanja ponderirano veličinama klastera: ako se dvije točke koje su negativno povezano nalaze u istom klaster, kazna će biti veća ako je pripadajući klaster manji. Slično, nagrada će biti veća ako su dvije točke u malom klasteru pozitivno povezane. Stoga, podijelit ćemo svaki $w_{ij}$ sa veličinom klastera u kojem se nalaze točke. Dobivamo:

$$\mathcal{J}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\mathbf{x}_i - \mathbf{m}_c\|^2 - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} \frac{w_{ij}}{|\pi_{l_i}|} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} \frac{w_{ij}}{|\pi_{l_i}|}. \tag{8}$$

Korištenjem sljedeće jednakosti (vidi Duda i Hart 1973):

$$\sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} 2\|\mathbf{x}_i - \mathbf{m}_c\|^2 = \sum_{c=1}^{k} \sum_{\mathbf{x}_i, \mathbf{x}_j \in \pi_c} \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{|\pi_c|}, \tag{9}$$

i prepravkom sume, minimiziranje ciljne funkcije postaje ekvivalentno minimiziranju:

$$\mathcal{J}(\{\pi_c\}_{c=1}^{k}) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i, \mathbf{x}_j \in \pi_c} \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{|\pi_c|} - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} \frac{2w_{ij}}{|\pi_{l_i}|} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} \frac{2w_{ij}}{|\pi_{l_i}|}. \tag{10}$$

Prepravak sume nam daje sljedeće:

$$\mathcal{J}(\{\pi_c\}_{c=1}^{k}) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i, \mathbf{x}_j \in \pi_c} \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{|\pi_c|} - \sum_{c=1}^{k} \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \pi_c \\ (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}}} \frac{2w_{ij}}{|\pi_{l_i}|} + \sum_{c=1}^{k} \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \pi_c \\ (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}}} \frac{2w_{ij}}{|\pi_{l_i}|}. \tag{11}$$

Neka je $E$ matrica vrijednosti kvadrata euklidske udaljenosti između točaka podataka, t.d. $E_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2$ i $W$ neka je matrica ograničenja t.d. je $W_{ij}$ jednak $-w_{ij}$ za negativne veze, $w_{ij}$ za pozitivne veze i 0 inače. Nadalje, uvest ćemo vektor indikacije $\mathbf{z}_c$ za klaster $c$. Ovaj vektor je duljine $n$ i $\mathbf{z}_c(i) = 0$ ako se $\mathbf{x}_i$ ne nalazi u klasteru $c$, a 1 inače. Sad možemo gornju funkciju cilja napisati na ovaj način:

$$\mathcal{J}(\{\pi_c\}_{c=1}^{k}) = \sum_{c=1}^{k} \frac{\mathbf{z}_c^T (E - 2W)\mathbf{z}_c}{\mathbf{z}_c^T \mathbf{z}_c}, \tag{12}$$

gdje je $\mathbf{z}_c^T \mathbf{z}_c$ veličina klastera $\pi_c$, a $\mathbf{z}_c^T (E - 2W)\mathbf{z}_c$ nam daje sumu $E_{ijn} - 2W_{ij}$ preko svih $x_i$ i $x_j$ unutar $\pi_c$. Sada definiramo matricu matricu $\tilde{Z}$ t.d. je $\tilde{Z}_c$, $c$-ti stupac od $\tilde{Z}$, jednak $\mathbf{z}_c/(\mathbf{z}_c^T \mathbf{z}_c)^{1/2}$. $\tilde{Z}$ je ortonormirana matrica ($\tilde{Z}^T \tilde{Z} = I_k$), te funkciju cilja koji želimo minimizirati možemo zapisati kao:

$$\sum_{c=1}^{k} \tilde{Z}_c^T (E - 2W)\tilde{Z}_c = trag(\tilde{Z}^T (E - 2W)\tilde{Z}). \tag{13}$$

Na kraju smo pokazali da se funkcija cilja jezgrene k-sredine može izraziti kao problem optimizacije traga (Dhillon i dr. 2004a). Konkretno, funkcija cilja jezgrene k-sredine je napisana kao maksimizacija od $trag(Y^T KY)$, gdje je $Y$ analogan $\tilde{Z}$ (ortonormirana matrica indikacije). Jedina razlika između ovih funkcija ciljeva je da naša funkcija cilja polu-nadzora je izražena kao minimiziranje traga, dok je algoritam jezgrene k-sredine izražen kao maksimiziranje traga. Kako bi naš problem pretvorili u problem maksimizacije, napominjemo da za kvadrat euklidske udaljenosti vrijedi $\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \mathbf{x}_i^T \mathbf{x}_i + \mathbf{x}_j^T \mathbf{x}_j - 2\mathbf{x}_i^T \mathbf{x}_j$. Neka je $S$ matrica sličnosti ($S_{ij} = \mathbf{x}_i^T \mathbf{x}_j$) i neka je $\tilde{S}$ matrica t.d. je $\tilde{S}_{ij} = \tilde{S}_{ii} * \tilde{S}_{jj}$. Tada je $E = \tilde{S} - 2S$.

Pripadajućom supstitucijom $E$ u minimizaciji traga, problem je ekvivalentan minimizaciji od $trag(\tilde{Z}(\tilde{S} - 2S - 2W)\tilde{Z})$. S obzirom da je $trag(\tilde{Z}^T \tilde{S}\tilde{Z})$ jednak $2 \cdot trag(S)$, a to je konstanta, taj trag se onda može ignorirati u optimizaciji. To nas vodi do ekvivalencije našeg problema sa maksimizacijom od $trag(\tilde{Z}(S + W)\tilde{Z})$. Ako definiramo matricu $K = S + W$, naš problem se može izraziti kao maksimizacija od $trag(\tilde{Z}^T K\tilde{Z})$, i matematički je ekvivalentan bestežinskom algoritmu jezgrene k-sredine.

Primijetimo da ova matrica $K$ ne mora biti pozitivno semidefinitna, što je uvjet konvergencije za algoritam jezgrene k-sredine. Ovaj se problem može izbjeći primjenom dijagonalnog pomaka (Dhillon i dr. 2007) za pojezgrenje matrice $K$, kao što je pokazano u drugom koraku Algoritma 2. To se postiže dodavanjem $\sigma I$ matrici $K$. Primijetimo:

$$trag(\tilde{Z}^T(\sigma I + K)\tilde{Z}) = \sigma \cdot trag(\tilde{Z}^T \tilde{Z}) + trag(\tilde{Z}^T K\tilde{Z}) \tag{14}$$

$$= \sigma k + trag(\tilde{Z}^T K\tilde{Z}). \tag{15}$$

**Algorithm 2:** Obični težinski algoritam jezgrene k-sredine

SMSP-KSREDINA($K, k, t_{max}, \alpha, \{\pi_c^{(0)}\}_{c=1}^k$)
**Ulaz**: $K$: jezgrena matrica, $k$: broj klastera, $t_{max}$: proizvoljan maksimalan broj iteracija, $\alpha$: težinski vektor, $\{\pi_c^{(0)}\}_{c=1}^k$): proizvoljni početni klasteri
**Izlaz**: $\{\pi_c^{(0)}\}_{c=1}^k$): konačna particija točaka
1. Inicijaliziraj $k$ klastera iz $\{\pi_c^{(0)}\}_{c=1}^k$), ako je dan u ulazu, inače nasumično.
2. Postavi $t = 0$.
3. Za svaku točku $\mathbf{x}_i$ i svaki klaster $c$, izračunaj $d(\mathbf{x}_i, \mathbf{m}_c)$ kao u (1).
4. Pronađi $c^*(\mathbf{x}_i) = \arg\min_c d(\mathbf{x}_i, \mathbf{m}_c)$, proizvoljno odabirući jedno od rješenja ako ih postoji više.
5. Izračunaj nove klastere sa:
$$\pi_c^{(t+1)} = \{\mathbf{x}_i \ : \ c^*(\mathbf{x}_i) = c\}. \tag{16}$$
6. Ako nema konvergencije ili $t_{max} > t$, postavi $t = t + 1$ i odi na Korak 3; Inače, stani i vrati konačne klastere $\{\pi_c^{(0)}\}_{c=1}^k$).

**Algorithm 3:** Obični težinski algoritam jezgrene k-sredine

SMSP-KSREDINA($K, k, t_{max}, \alpha, \{\pi_c^{(0)}\}_{c=1}^k$)
**Ulaz**: $K$: jezgrena matrica, $k$: broj klastera, $t_{max}$: proizvoljan maksimalan broj iteracija, $\alpha$: težinski vektor, $\{\pi_c^{(0)}\}_{c=1}^k$): proizvoljni početni klasteri
**Izlaz**: $\{\pi_c^{(0)}\}_{c=1}^k$): konačna particija točaka
1. Inicijaliziraj $k$ klastera iz $\{\pi_c^{(0)}\}_{c=1}^k$), ako je dan u ulazu, inače nasumično.
2. Postavi $t = 0$.
3. Za svaku točku $\mathbf{x}_i$ i svaki klaster $c$, izračunaj $d(\mathbf{x}_i, \mathbf{m}_c)$ kao u (1).
4. Pronađi $c^*(\mathbf{x}_i) = \arg\min_c d(\mathbf{x}_i, \mathbf{m}_c)$, proizvoljno odabirući jedno od rješenja ako ih postoji više.
5. Izračunaj nove klastere sa:
$$\pi_c^{(t+1)} = \{\mathbf{x}_i \ : \ c^*(\mathbf{x}_i) = c\}. \tag{17}$$
6. Ako nema konvergencije ili $t_{max} > t$, postavi $t = t + 1$ i odi na Korak 3; Inače, stani i vrati konačne klastere $\{\pi_c^{(0)}\}_{c=1}^k$).

Konstanta je dodana funkciji cilja tako da je globalno optimalno rješenje jednako i za pomaknutu i nepomaknutu matricu. Stoga, pomicanje dijagonale nam dozvoljava izradu pozitivno semidefinitne matrice $K$ (s obzirom da dodavanje $\sigma I$ matrici $K$ povećava svojstvene vrijednosti za $\sigma$), a to nam jamči monotonu konvergenciju algoritma jezgrene k-sredine ka lokalnom optimumu, istodobno zadržavajući globalno optimalno rješenje. Kao što je spomenuto u odjeljku 2.1, algoritam jezgrene k-sredine se može pokrenuti direktno na matrici $K$ - iako nemamo teoretsko jamstvo konvergencije, općenito algoritam monotono konvergira u praksi.

## 3.2 Algoritam

Rezimirajući rezultat iz prethodnog odjeljka, pokazali smo ekvivalenciju između bestežinskog algoritma jezgrene k-sredine i algoritma SMSP-KSREDINA sa kvadratom euklidske udaljenosti i težinskim kaznenim funkcijama ponderiranim veličinama klastera. Sada možemo raspravljati o algoritmu na temelju jezgre za funkciju cilja algoritma SMSP-KSREDINA konstruirajući pripradnu jezgrenu matricu $K$ (vidi Algoritam 2). Ovaj algoritam konstruira jezgru (izvedenu kao u prošlom odjeljku), izvodi pripadnu inicijalizaciju klastera i pokreće algoritam jezgrene k-sredine.

Ključna prednost ovog pristupa je taj da algoritam pretpostavlja da se matrica sličnosti nalazi u ulazu. Kao što je dano u izvodu, matrica sličnosti je matrica skalarnih produkta između vektora, ali se ovo lako

može poopćiti primjenom jezgrene funkcije na vektorskim podacima kako bi se izradila matrica sličnosti. Stoga, za razliku od prethodnih pristupa optimizaciji funkcije cilja algoritma SMSP-KSREDINA, naš pristup se može lako poopćiti za obradu nelinearnih granica klastera, npr. primjenom Gaussove jezgre prilikom izrade matrice sličnosti.

Korak koji još nismo raspravili je inicijalizacija klastera prije pokretanja algoritma k-sredina. U standardnom algoritmu k-sredine, moguće su mnoge sheme inicijalizacije za generiranje početnih klastera. Međutim, u našem slučaju, informacije o ograničenju nam daju važne naznake o strukturi klastera, a ove informacije možemo uključiti u inicijalizacijski korak algoritma. Za inicijalizaciju klastera, slijedimo pristup od Basu i dr. (2004b). Najprije uzimamo tranzitivno zatvorenje pozitivnih veza, pretpostavljajući da su sva ograničenja konzistentna. To nam daje skup povezanih komponenti pozitivnih veza.

Zatim izvodimo sljedeće negativne veze na idući način: ako postoji negativna veza između dvije povezane komponente, dodajemo ograničenje negativnih veza između svih parova točaka $a$ i $b$, gdje $a$ pripada prvoj povezanoj komponenti, a $b$ pripada drugoj povezanoj komponenti.

Nakon ovog koraka generiramo početne klastere pomoću algoritma prvi-najudaljeniji (farthest-first, eng.). Izračunamo povezane komponente i onda izaberemo njih $k$ za početne klastere. Prvi-najudaljeniji algoritam prvo odabire najveću povezanu komponentu, a zatim iterativno odabire klaster (komponentu) koji je najudaljeniji od trenutno odabranog klastera, gdje se udaljenost između klastera mjeri ukupnom udaljenošću između svih točaka u ovim klasterima. Nakon što je $k$ početnih klastera odabrano, inicijaliziramo sve točke tako što ih smjestimo u njima najbliži klaster. Primijetite da se svi izračuni udaljenosti u ovoj proceduri mogu izvesti učinkovito u prostoru jezgre.

# 4 Polu-nadzirano klasteriranje grafa

Kao što smo već spomenuli, postoji izravna matematička veza između težinske funkcije cilja jezgrene k-sredine i široke klase funkcija cilja za klasteriranje grafa. Do sada smo razmatrali vektorski slučaj dok smo izvodili vezu između algoritma SMSP-KSREDINA i algoritma jezgrene k-sredine. U ovom odjeljku, generaliziramo ovaj rezultat za slučaj nekoliko funkcija cilja za polu-nadzirano klasteriranje grafa.

## 4.1 Funkcije cilja

Funkcija cilja algoritma SMSP-KSREDINA je motivirana funkcijom cilja sa tri izraza: jedan izraz bi predstavljao cilj klasteriranja, drugi izraz poticanje ograničenja pozitvnih veza, a treći poticanje ograničenja negativnih veza. U istom smislu, sada ćemo razmotriti tri funkcije cilja za polu-nadzirano klasteriranje grafa, pri čemu su ciljevi klasteriranja temeljeni grafovski, umjesto na vektorski.

*Polu-nadzirani normalizirani rez* Polu-nadzirana verzija funkcije cilja normaliziranog reza grafa nastoji minimizirati:

$$\sum_{c=1}^{k} \frac{povezanost(\mathcal{V}_c, \mathcal{V} \setminus \mathcal{V}_c)}{stupanj(\mathcal{V}_c} - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} \frac{w_{ij}}{stupanj(\mathcal{V}_{l_i})} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} \frac{w_{ij}}{stupanj(\mathcal{V}_{l_i})} \tag{18}$$

Primijetimo da, umjesto veličina klastera, kod kaznenih funkcija nam stupnjevi predstavljaju težine kod klastera.

*Polu-nadzirani razmjerni rez* Slično, funkcija cilja za polu-nadzirani razmjerni rez se može napisati kao minimizacija od:

$$\sum_{c=1}^{k} \frac{povezanost(\mathcal{V}_c, \mathcal{V} \setminus \mathcal{V}_c)}{|\mathcal{V}_c|} - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} \frac{w_{ij}}{|\mathcal{V}_{l_i}|} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} \frac{w_{ij}}{|\mathcal{V}_{l_i}|}. \tag{19}$$

Tablica 3: Popularne funkcije cilja i odgovarajuće težine i jezgre sa danom matricom susjedstva A

| Ime | Težine vrhova | Jezgra |
|---|---|---|
| Razmjerna povezanost | 1 za sve čvorove | $K = \sigma I + A$ |
| Razmjerni rez | 1 za sve čvorove | $K = \sigma I - L$ |
| Normalizirani rez | Stupanj vrhova | $K = \sigma D + D^{-1}AD^{-1}$ |

**Algorithm 4:** Obični težinski algoritam jezgrene k-sredine

SMSP-KSREDINA($K, k, t_{max}, \alpha, \{\pi_c^{(0)}\}_{c=1}^k$)

**Ulaz**: $K$: jezgrena matrica, $k$: broj klastera, $t_{max}$: proizvoljan maksimalan broj iteracija, $\alpha$: težinski vektor, $\{\pi_c^{(0)}\}_{c=1}^k$): proizvoljni početni klasteri

**Izlaz**: $\{\pi_c^{(0)}\}_{c=1}^k$): konačna particija točaka

1. Inicijaliziraj $k$ klastera iz $\{\pi_c^{(0)}\}_{c=1}^k$), ako je dan u ulazu, inače nasumično.
2. Postavi $t = 0$.
3. Za svaku točku $\mathbf{x}_i$ i svaki klaster $c$, izračunaj $d(\mathbf{x}_i, \mathbf{m}_c)$ kao u (1).
4. Pronađi $c^*(\mathbf{x}_i) = \arg\min_c d(\mathbf{x}_i, \mathbf{m}_c)$, proizvoljno odabirući jedno od rješenja ako ih postoji više.
5. Izračunaj nove klastere sa:
$$\pi_c^{(t+1)} = \{\mathbf{x}_i \ : \ c^*(\mathbf{x}_i) = c\}. \tag{21}$$
6. Ako nema konvergencije ili $t_{max} > t$, postavi $t = t + 1$ i odi na Korak 3; Inače, stani i vrati konačne klastere $\{\pi_c^{(0)}\}_{c=1}^k$).

*Polu-nadzirana razmjerna povezanost* Konačno, funkcija cilja za polu-nadzirani razmjerni odnos se može napisati kao maksimizacija od:

$$\sum_{c=1}^{k} \frac{povezanost(\mathcal{V}_c, \mathcal{V}_c)}{|\mathcal{V}_c|} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} \frac{w_{ij}}{|\mathcal{V}_{l_i}|} - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} \frac{w_{ij}}{|\mathcal{V}_{l_i}|}. \tag{20}$$

Primijetimo kako se drugi i treći izraz u ovoj funkciji cilja imaju suprotne znakove od pripadajućih izraza u gornjim funkcijama; to je zato što ovdje nastojimo maksimizirati povezanost.

## 4.2 Jezgreni algoritam za polu-nadzirano klasteriranje grafa

Sada ćemo iznijeti opći algoritam za polu-nadzirano klasteriranje grafa koje može optimizirati sve tri funkcije cilja koje su navedene gore. Da bismo to postigli, moramo pokazati da svaka od ovih funkcija se može zapisati kao specijalan slučaj težinske funkcije cilja jezgrene k-sredine.

Prvo, razmotrimo minimizaciju polu-nadziranog normaliziranog reza. Na temelju analize Yu i Shi (2003), prvi izraz u funkciji cilja polu-nadziranog normaliziranog reza se može izraziti kao $k - trag(\tilde{Y}^T D^{-1/2} A D^{-1/2} \tilde{Y})$, gdje je $\tilde{Y}$ ortonormirana matrica indikacije klastera, $D$ je dijagonalna matrica stupnjeva vrhova, a $A$ je matrica susjedstva u grafu. Drugi i treći izraz funkcije cilja se može sažeto izraziti sa $-trag(\tilde{Y}^T D^{-1/2} W D^{-1/2} \tilde{Y})$, koristeći sličnu analizu kao u izvodima (5) i (6). Kombinirajući ova dva izraza, dobijemo da je funkcija cilja polu-nadziranog normaliziranog reza ekvivalentna minimiziranju $k - trag(\tilde{Y}^T D^{-1/2}(A + W) D^{-1/2} \tilde{Y})$ ili, ekvivalentno, maksimiziranju $trag(\tilde{Y}^T D^{-1/2}(A + W) D^{-1/2} \tilde{Y})$. Generalizacija analize Dhillona i dr. (2007) nam omogućuje da izrazimo ovu maksimizaciju traga jednako kao težinski algoritam jezgrene k-sredine, na sljedeći način. Postavimo $A' = A + W$ kao u bestežinskom slučaju, i pokrenimo težinski algoritam k-sredina na $\sigma D^{-1} + D^{-1} A' D^{-1}$ gdje su težine jednake stupnjevima vrhova. Pokretanje težinskog algoritma jezgrene k-sredine znači da sa ovom jezgrom i težinama uzrokovati monoton pad funkcije cilja u (7) za graf sa matricom susjedstva $A$.

Analogni rezultati se mogu dobiti za polu-nadzirani razmjerni rez i polu-nadziranu razmjernu povezanost, koristeći slični analiza. Za polu-nadzirani razmjerni rez, pokrećemo težinski algoritam k-sredina na $\sigma I - L + W$ ($L$ je Laplaceova matrica grafa od $A$), gdje su sve težine jednake jedinici. U razmjernoj povezanosti, pokrećemo težinski algoritam k-sredina na $\sigma I + A + W$, gdje su sve težine jednake jedinici. U svim slučajevima, dodavanje nadzora težinskom algoritmu k-sredine jednostavno zahtjeva dodavanje matrice ograničenja $W$ na prikladan način jezgrenoj matrici.

Tablica 3 sažeto prikazuje izgradnju jezgre i težina za svaku od tih funkcija cilja polu-nadziranog klasteriranja kako bi bile ekvivalentne težinskim funkcijama cilja jezgrene k-sredine, a Algoritam 4 opisuje opći jezgreni algoritam polu-nadtiranog klasteriranja. Primijetimo da je daljnja generalizacija moguća; može se dobiti opća, težinska funkcija cilja polu-nadziranog klasteiranja grafa koja je normalizirana zbrojem proizvoljnih težina umjesto stupnjevima ili veličinama klastera. U ovom slučaju $\alpha$ bi bio proizvoljan težinski vektor.

Također primijetimo da u slučaju *polu-nadziranog algoritma razmjerne povezanosti, algoritam identičan alguritmu* SMSP-KSREDINA, *gdje se matrica susjedstva u grafu koristi za ulaznu matricu sličnosti.*

### 4.3 Poveznice sa spektralnim učenjem

Algoritam SPEKTRALNO-UČENJE (Kanvar i dr. 2003), opisan u odjeljku 2.2.2, možemo promatrati u našim okvirima na sljedeći način: za ograničenja pozitivnih veze, ako $A_{ij}$ predstavlja sličnost između $\mathbf{x}_i$ i $\mathbf{x}_j$, postavit ćemo $W_{ij} = 1 - A_{ij}$ (i posljedično, odgovarajuća vrijednost u $A + W$ je 0). Slično, za negativne veze, postavljamo $W_{ij} = -A_{ij}$ (i posljedično, odgovarajuća vrijednost u $A + W$ je jednaka 0). Sa ovim posebnim odabirom težina za ograničenje, matrica $A + W$ je identična matrici iz algoritma SPEKTRALNO-UČENJE prije aditivne normalizacije. To nam ostavlja samo korak aditivne normalizacije, što je istovjetno normalizaciji potrebnoj za polu-nadzirani razmjerni rez (do na dijagonalni pomak, koji ne mijenja globalno optimalno rješenje).

Dobro poznata relaksacija za naš problem maksimizacije traga je postignuta uzimanjem gornjih $k$ svojstvenih vrijednosti matrice $K$. Matrica dobivena sa tim svojstvenim vrijednostima je optimalna $Y$ matrica, uz relaksaciju da je $Y$ proizvoljna ortonormirana matrica. Takva relaksacija nas dovodi do algoritma SPEKTRALNO-UČENJE koji je detaljno opisan u odjeljku 2.2.2.

Stoga, algoritam SPEKTRALNO-UČENJE može se promatrati kao rješavanje funkcije cilja relaksiranog polu-nadziranog razmjernog reza. Štoviše, naša ranija analiza pokazuje da se brzi algoritam jezgrene k-sredine može iskoristiti za optimizaciju ove funkcije, čime se uklanja svaka potreba za skupom operacijom spektralne dekompozicije (za velike skupove podataka). Alternativno, ova analiza sugerira druge spektralne algoritme temeljene na polu-nadziranom razmjernom rezu ili polu-nadziranoj razmjernoj povezanosti.

## 5 Rezultati istraživanja

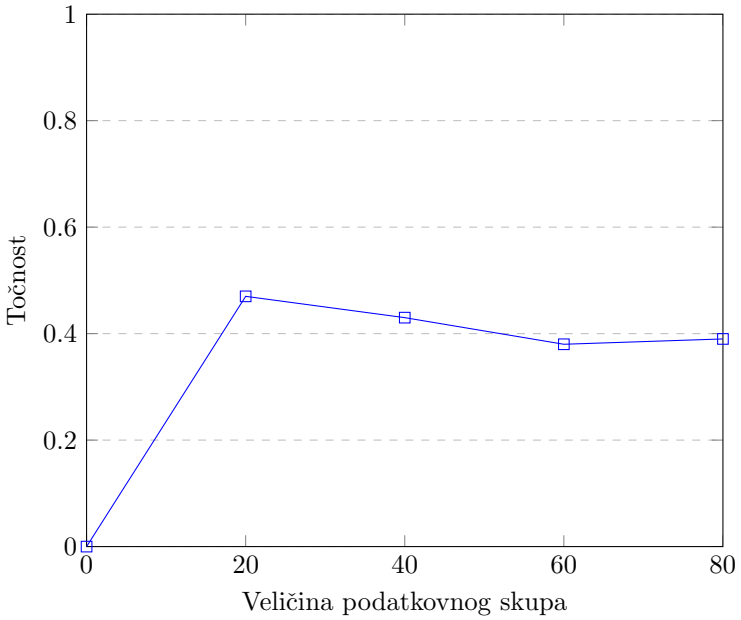
U ovom poglavlju, predstavljamo rezultate istraživanja na vektorskim podacima, uspoređujući algoritam SMSP-KSREDINA i algoritam k-sredina.

### 5.1 Skup podataka

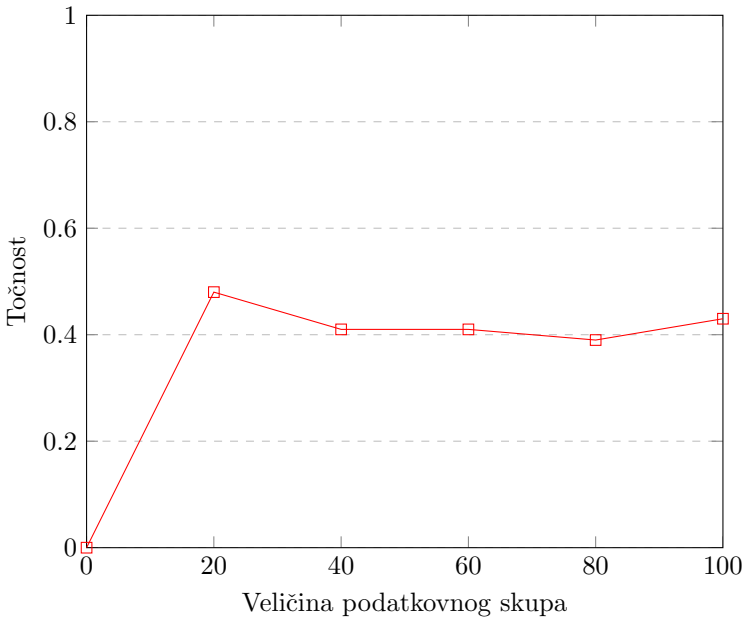
Za skup podataka smo koristili znamenke iz MNIST skupa podataka, i to znamenke 3, 6 i 9, s obzirom na njihovu sličnost.

## 5.2 Rezultati istraživanja

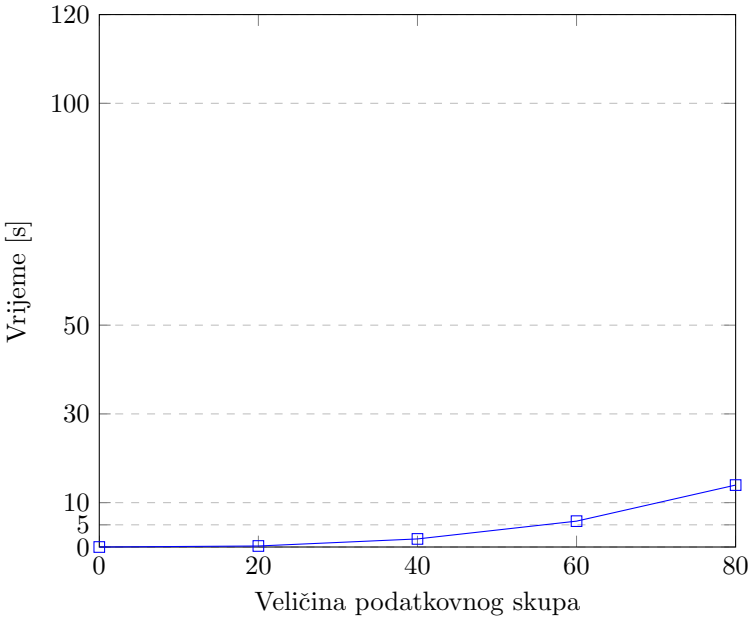
SMSP-KSREDINA: Prosječna točnost po veličini skupa podataka



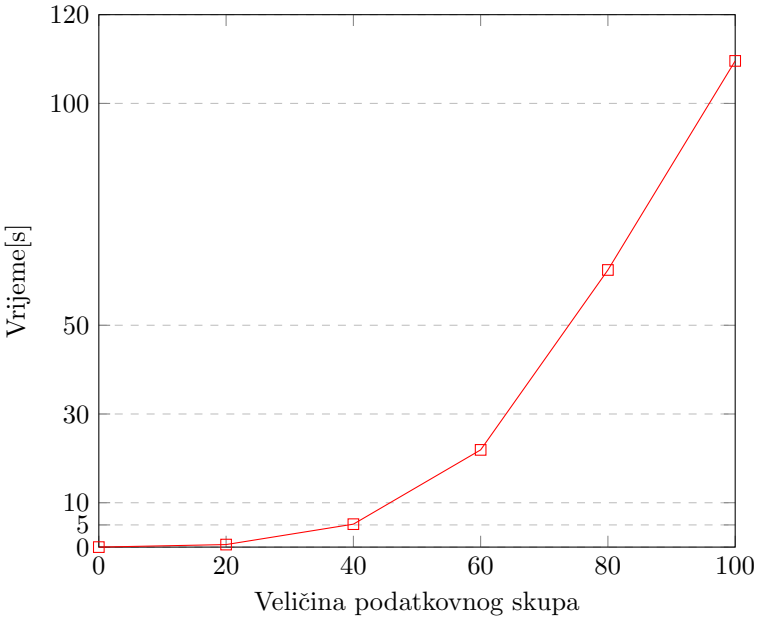

k-sredina: Prosječna točnost po veličini skupa podataka



SMSP-KSREDINA: Prosječno vrijeme izvršavanja algoritma po veličini skupa podataka




k-sredina: Prosječno vrijeme izvršavanja algoritma po veličini skupa podataka




# A Appendix Heading