Sveučilište u Zagrebu

Prirodoslovno-matematički fakultet

Petar Sirković i Saša Stanko

Otkrivanje blok dijagonalne strukture stohastičkih matrica i identifikacija metastabilnih stanja Markovljevih lanaca

Zagreb, 2011.g.

Ovaj rad je izrađen na Matematičkom odsjeku, Prirodoslovno-matematičkog fakulteta u Zagrebu, pod vodstvom prof. dr. sc. Zlatka Drmača te je predan na natječaj za dodjelu Rektorove nagrade u akademskoj godini 2010./2011.

Sadržaj

1 Uvod

1.1 Definicija problema

Zamislimo blok-dijagonalnu matricu A

$$A = \begin{pmatrix} A_{11} & & & \\ & A_{22} & & \\ & & \ddots & \\ & & & A_{mm} \end{pmatrix}, A_{ii} \in \mathbb{R}^{n_i \times n_i}, n_1 + n_2 + \ldots + n_m = n$$

takvu da su matrice A_{ii} na dijagonali stohastičke. Blok-dijagonalna struktura matrice je prvo pokvarena dodavanjem matrice E koja u nekom smislu nije "velika", a zatim primjenom kongruencije matricom permutacije P. Time dobivamo matricu $B = P(A + E)P^T$. Zadatak je za zadanu matricu B što je moguće bolje rekonstruirati blok-dijagonalnu strukturu matrice A. Važan podzadatak našeg problema jest uopće odrediti broj dijagonalnih blokova matrice A. U nekim inačicama ovog problema broj dijagonalnih blokova je unaprijed poznat.

1.2 Primjene

Ovaj problem definiran kao takav ima svoju primjenu u raznim područjima znanosti i tehnike. Jedna od primjena, na čemu ćemo i testirati svoje metode, jest u kemiji u proučavanju kemijskih reakcija i međustanja do kojih dolazi tijekom reakcije, na primjer u sintezi peptida [1] ili n-pentana [2]. Druga važna primjena je u kompjuterskoj analizi digitalnih slika [3], gdje je važan problem kako iz slike izvući dijelove koji "pripadaju" jednoj cjelini. Ova problematika se spominje i u ekonomiji, točnije u analizi tržišta, gdje je potrebno tržište segmetirati u povezane (slične) cjeline.

1.3 Pregled rada

Prvo opisujemo različite poglede na problem preko teorije nenegativnih matrica, teorije Markovljevih lanaca te teorije grafova, uspostavljamo vezu među njima te navodimo neke od pristupa rješavanju problema koje možemo pronaći u literaturi. Nakon toga slijede opisi i razrada dvaju novih "bottom-up" metoda rješavanja problema koje smo osmislili. Osnova prvog pristupa su svojstva minimalnog razapinjujućeg stabla te Markovljevih lanaca, dok je u drugom pristupu opisana metoda rješavanja preko metaheurističke metode simuliranog kaljenja. Na kraju donosimo rezultate testiranja i usporedbe naših metoda sa metodama dostupnima u literaturi. U dodatku, na kraju rada, se također može vidjeti priručnik za uporabu Matlab GUI aplikacije koju smo razvili kako bi si olakšali testitanje i usporedbu metoda.

2 Konvencije

Navodimo konvencije koje su korištene u radu. Prvenstveno u označavanju, ali i u korištenim terminima.

Matrica A je stohastička blok-dijagonalna matrica sa m blokova $A_{11}, \ldots, A_{mm}$,

$$A = \begin{pmatrix} A_{11} & & & \\ & A_{22} & & \\ & & \ddots & \\ & & & A_{mm} \end{pmatrix}.$$

Sa S_i označimo skup svih indeksa koji su potrebni za indeksiranje elemenata bloka A_{ii}, $i = 1, \ldots, m$. Kako A_{ii} jedinstveno određuje S_i, ali i obratno, skup S_i također nazivamo blokom. Time smo konstruirali particiju $\{S_1, \ldots, S_m\}$ skupa $\{1, \ldots, n\}$ koja bez smanjenja općenitosti zadovoljava $S_1 \leq \cdots \leq S_m$, pri čemu smatramo da je $S_i \leq S_j$ ako je $\max S_i \leq \min S_j$.

Matricu perturbacije označavamo s E, dok permutacijske matrice označavamo s P i Q. Umjesto permutacijska matrica pišemo samo, pogrešno ali kraće, permutacija. Matrica B je uvijek jednaka $P(A + E)P^T$.

Primjenom permutacije P na matricu A, skup elemenata potrebnih za indeksiranje prijašnjeg bloka A_{ii} će biti skup $p(S_i)$. Ako skupovi $p(S_i)$ sadrže samo uzastopne elemente, kažemo da je struktura vidljiva. U suprotnome, kažemo da je skrivena.

Broj blokova matrice B smatramo jednakim broju blokova matrice A. Nadalje, smatramo da je i-ti blok od B određen skupom S_i. O vidljivosti blok-strukture govorimo analogno kao i kod A.

Oznaka $\mathbb{1}$ će poslužiti za matrice čiji su svi elementi jednaki jedan. Ukoliko dimenzije ove matrice nisu jasne iz konteksta, tada će biti zapisane kao indeks same oznake.

3 Različiti pogledi na problem

Proučavanjem problema, naišli smo na raznolike pristupe rješavanju. Četiri takva predstavljamo u ovom poglavlju. Za svaki iznosimo potrebnu teoriju koju ćemo koristiti i u ostatku rada. Nadalje, pokazujemo kako se svi pristupi mogu povezati korištenjem spektralne teorije.

3.1 Nenegativne matrice

Matrice problema su nenegativne i stohastičke. Kroz sljedeće definicije se upoznajemo s tim pojmovima.

Definicija 3.1. *Matrica $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ je **pozitivna (nenegativna)**, ako vrijedi $a_{ij} > 0$ ($a_{ij} \geq 0$) za sve $i, j \in \{1, \ldots, n\}$.*

Definicija 3.2. *Matrica A je **stohastička(po retcima)** ako je nenegativna, a suma elemenata svakog retka je jednaka jedan.*

Definicija 3.3. *Kažemo da je $A \in \mathbb{C}^{n\times n}$, $n \geq 2$, reducibilna ako postoji matrica permutacije P i $r \in \{1, \ldots, n-1\}$ tako da je*

$$P^T AP = \begin{pmatrix} \tilde{A}_{[11]} & \tilde{A}_{[12]} \\ 0 & \tilde{A}_{[22]} \end{pmatrix}, \quad \tilde{A}_{[11]} \in C^{r\times r}. \tag{1}$$

Za $n = 1$ je A reducibilna ako je $A = (0)$. Matrica A je ireducibilna ako nije reducibilna.

Definicija 3.4. *Na prostoru matrica $A \in C^{n\times n}$ definiramo normu $||\cdot||_\infty$ sa*

$$||A||_\infty = max_{i=1}^n \sum_{j=1}^n |a_{ij}|.$$

Teorem 3.1. *(Perron) Neka je $A \in \mathbb{R}^{n\times n}$ pozitivna matrica i $\rho(A)$ njen spektralni radijus. Tada vrijedi:*

- $\rho(A) > 0$ *i* $\rho(A)$ *je svojstvena vrijednosti od* A *algebarske kratnosti jedan, pri čemu se pripadni svojstveni vektor* v *može odabrati sa realnim pozitivnim komponentama,* $Av = \rho(A)v$, $v > 0$.

- $\rho(A)$ *je jedina svojstvena vrijednost koja postiže maksimalnu apsolutnu vrijednost, tj.* $(\forall \lambda \in \sigma(A))\ (\lambda \neq \rho(A) \Longrightarrow |\lambda| < \rho(A))$.

Teorem 3.2. *Neka je* $A \in \mathbb{R}^{n\times n}$ *nenegativna i ireducibilna matrica. Tada vrijedi:*

- $\rho(A) > 0$ *i* $\rho(A)$ *je svojstvena vrijednosti od* A *algebarske kratnosti jedan, pri čemu se pripadni svojstveni vektor* v *može odabrati sa realnim pozitivnim komponentama,* $Av = \rho(A)v$, $v > 0$.

- *Spektralni radijus je strogo rastuća funkcija: Ako povećamo bilo koji element od* A *i dobijemo* $\tilde{A} \geq A$ *i* $\tilde{A} \neq A$, *onda je* $\rho(\tilde{A}) > \rho(A)$.

- *Ako u spektru od* A *točno* ℓ *svojstvenih vrijednosti ima modul jednak* $\rho(A)$, *onda su one jednake*

$$\rho(A)e^{2\pi \mathrm{i} j/\ell},\ j = 0, \ldots, \ell - 1.$$

Za stohastičku matricu A vrijedi $\|A\|_\infty = 1$, ali i $A \cdot \mathbb{1} = 1 \cdot \mathbb{1}$. Zato je 1 njezina svojstvena vrijednost, a $\mathbb{1}$ njezin svojstveni vektor. Iz teorema 3.2 Perron-Frobeniusove teorije dobivamo dovoljne uvjete na jednostrukost svojstvene vrijednosti 1. To su pozitivnost ili nenegativnost i ireducibilnost matrice A.

Ako je A blok-dijagonalna i stohastička, tada je svaki njezin blok također stohastička matrica. Prema spomenutome, zaključujemo da je $\lambda = 1$ svojstvena vrijednost bloka. Nadalje, ulaganjem svojstvenog vektora svojstvene vrijednosti 1 u prostor vektora dimenzije matrice, dobivamo svojstveni vektor za A. Ponovimo li postupak za svaki blok, rezultat će biti skup linearno nezavisnih vektora. Stoga algebarska kratnost svojstvene vrijednosti λ matrice A nije manja od broja blokova. Uz pretpostavku na ireducibilnost blokova, umjesto nejednakosti dobivamo jednakost. Stoga se informacija o broju

blokova matrice A može dobiti određivanjem algebarske kratnosti svojstvene vrijednosti λ. Pozicije blokova se mogu odrediti korištenjem pripadnih svojstvenih vektora. Nakon dodavanja perturbacije, više se neće raditi o svojstvenoj vrijednosti λ nego o njoj bliskim vrijednostima. No, svojstveni vektori će i dalje pružati dobre informacije. Primjena permutacije mijenja samo redoslijed elemenata u svojstvenim vektorima, što ne predstavlja problem dohvaćanja informacije iz svojstvenih vektora. Ovaj pristup je nešto detaljnije razrađen i korišten u [1].

3.2 Markovljevi lanci

Markovljevi lanci su detaljno opisani u knjizi [4], a mi navodimo samo nama najbitnije detalje.

Definicija 3.5. ***Diskretni Markovljev lanac*** *je niz slučajnih varijabli $X_1, X_2, X_3, ...$ takvih da zadovoljavaju Markovljevo svojstvo tj. uz uvjet sadašnjeg stanja buduća i prošla stanja su nezavisna.*

$$P(X_{n+1} = x | X_1 = x_1, X_2 = x_2, ..., X_n = x_n) = P(X_{n+1} = x | X_n = x_n)$$

Prebrojiv skup vrijednosti S koje poprimaju slučajne varijable X_i nazivamo ***skup stanja*** *lanca.*

Definicija 3.6. ***Vremenski neovisan*** *Markovljev lanac je diskretan Markovljev lanac koji zadovoljava svojstvo* ***vremenske homogenosti****, tj.*

$$P(X_{n+1} = x | X_n = y) = P(X_2 = x | X_1 = y) \;\; n \in \mathbb{N}, x, y \in S$$

Definicija 3.7. *Def. Za vremenski neovisan Markovljev lanac možemo definirati* ***matricu prijelaza*** *$P = (p_{ij})$, gdje je $p_{ij} = P(X_2 = j | X_1 = i)$. Zbog svojstva vremenske homogenosti ovako definirana matrica prijelaza opisuje ponašanje Markovljevog lanca u svim vremenskim trenucima, a ne samo u trenutku* 1.

Budući da u našem problemu imamo zadanu stohastičku matricu A, problem možemo promatrati i iz teorije Markovljevih lanaca tako da matricu problema zamislimo kao matricu prijelaza, a skup redaka (vrhova) zamislimo kao skup stanja. U tom slučaju je vjerojatnost prelaska iz stanja i u stanje j, jednaka $[A]_{ij} = a_{ij}$. Blok struktura koju pokušavamo naći će u tom slučaju predstavljati skup meta stanja. Meta stanje je skup osnovnih stanja koja su međusobno jako povezana, a prema ostalim čvorovima nemaju veza (ili su slabo povezani). Postojanje meta stanja je usko povezano sa ireducibilnošću matrice, ako kod definicije reducibilnosti matrice ne zahtjevamo da je vandijagonalni blok jednak 0, već po normi manji od neke vrijednosti.

Definicija 3.8. *Stanje $s \in S$ Markovljevog lanca X je **povratno** ukoliko je $P_s(T_s < \infty) = 1$, to jest vjerojatnost da će se lanac vratiti u stanje s u konačnom vremenu ukoliko je iz njega izašao je 1. Sa $P_s(\cdot)$ smo označili vjerojatnost induciranu Markovljevim lancem koji kreće iz stanja s, a sa T_s vrijeme prvog dolaska u stanje s. Kažemo da je cijeli Markovljev lanac **povratan** ukoliko je svako stanje u njemu povratno.*

Svaki konačan Markovljev lanac, što kod nas i je slučaj, ima barem jedno povratno stanje.

Definicija 3.9. *Niz $\lambda = (\lambda_i : i \in S)$ naziva se **invarijantna mjera** Markovljevog lanca X (odnosno matrice prijelaza P) ukoliko je $\lambda_i \in [0, \infty\rangle$ za sve $i \in S$, $\lambda_i > 0$ za barem jedan $i \in S$ te ako vrijedi*

$$\lambda = \lambda P.$$

Definicija 3.10. *Niz $\lambda = (\lambda_i : i \in S)$ naziva se **stacionarna distribucija** Markovljevog lanca X (odnosno prijelazne matrice P) ukoliko je $\lambda_i \in [0, \infty\rangle$ za sve $i \in S$ te*

$$\sum_{i \in S} \lambda_i = 1.$$

Postojanje povratnog stanja povlači i postojanje barem jedne invarijantne mjere tog Markovljevog lanca. Kako je lanac u našem slučaju konačan, ta mjera se može skalirati na stacionarnu distribuciju. Ta stacionarna distribucija je jedinstvena u slučaju da je matrica ireducibilna. Dakle, u slučaju da možemo pronaći blok dijagonalnu reprezentaciju naše matrice problema, to povlači da svaki od blokova na dijagonali ima svoju stacionarnu distribuciju, nezavisnu od preostalih blokova na dijagonali. Stacionarne distribucije cijele matrice su u tom slučaju skalirane linearne kombinacije stacionarnih distribucija pojedinih dijagonalnih blokova.

Time zaključujemo da broj meta stanja možemo utvrditi i izračunavanjem spektra matrice te provjerom koliko je svojstvenih vrijednosti jednako ili blizu 1. Uočavamo kako smo dobili isti zaključak kao i kod nenegativnih matrica. Stoga možemo ustvrditi da iako su spomenuti pristupi različiti, u konačnici se svode na isto.

U našem slučaju, granična i stacionarna distribucija su jako bliski pojmovi. Ukoliko bismo imali uvjet da je matrica problema aperiodična, mogli bismo pokazati da je stacinarna distribucija jednaka graničnoj na svakoj od ireducibilnih komponenti.

3.3 Spektralni rezovi na grafovima

Kako mi u našem problemu trebamo "razvrstati" elemente(blokove) matrice i "pokidati" veze među njima, prirodno je promatrati problem i kroz teoriju grafova gdje razvrstavanje blokova predstavlja kidanje veza u grafu. Tu se radi o poznatom problemu spektralnih rezova na grafu.

Definicija 3.11. ***Graf*** *je uređeni par $G = (V, E)$, gdje je $V \neq \emptyset$ i $E \subseteq V \times V$. Elemente skupa $V = V(G)$ nazivamo vrhovi, a elemente skupa $E = E(G)$ bridovi. Ako vrijedi $(\forall (v, u) \in V^2)(v, u) \in E \Leftrightarrow (u, v) \in E$, kažemo da je graf neusmjeren, a u suprotnom da je usmjeren.*

Definicija 3.12. *Neka je $G = (V, E)$ graf. Definiramo* ***težinski graf*** *kao uređenu trojku*

$G_\omega = (V, E, \omega)$, *gdje je* $\omega\colon E \to \mathbb{R}^+$ *težinska funkcija. Za svaki* $e \in E$ *vrijednost* $\omega(e)$ *nazivamo* ***težina brida*** e. *Vrijednost* $\omega(G) = \sum_{e \in E} \omega(e)$ *nazivamo* ***težina grafa*** G.

Sljedeća definicija daje vezu između matrica i grafova.

Definicija 3.13. *Neka je* $A \in \mathbb{R}^{n \times n}$. *Kažemo da je* G_A ***graf induciran matricom*** A, *ako mu je skup vrhova jednak* $\{1, \ldots, n\}$, *a skup bridova jednak* $\{(i, j)\colon 1 \leq i, j \leq n, [A]_{ij} \neq 0\}$. *Težina pojedinog brida jednaka je odgovarajućoj vrijednosti elementa matrice.*

Opišimo sada normalizirani rez – pristup koji se bazira na particioniranju grafova reprezentiranih matricom problema. Zbog konzistentnosti oznaka koje se koriste prilikom računanja normaliziranog reza, matricu problema ćemo označiti s W.

Skup vrhova V grafa $G = (V, E)$, možemo podijeliti na 2 disjunktna dijela, A i B, micanjem bridova koji povezuje vrhove iz različitih dijelova. Za svaku podjelu tj. **rez** definiramo njegovu težinu kao

$$\text{cut}(A, B) = \sum_{u \in A, v \in B} w(u, v),$$

gdje sa $w(u, v)$ označavamo težinu brida između vrhova u i v. Intuitivno je jasno da se minimizacijom težine reza dobivaju "najneovisniji" dijelovi grafa. U terminima našeg problema to povlači da su elementi van blokova definiranih tim skupovima stanja najmanji mogući. Postoje razne efikasne metode za traženje **minimalnog reza**. Međutim, traženje minimalnog reza u našem problemu nije uvijek najbolje, jer se globalni minimum funkcije minimalnog reza često postiže na podjelama koje npr. odvajaju samo jedan vrh od svih preostalih. Iz tog razloga se počeo proučavati tzv. **normalizirani rez**. Njegova težina za određenu podjelu jest

$$\text{Ncut}(A, B) = \frac{\text{cut}(A, B)}{\text{veze}(A, V)} + \frac{\text{cut}(A, B)}{\text{veze}(B, V)},$$

gdje je sa $\text{veze}(A,V) = \sum_{u\in A, v\in V} w(u,v)$ označena ukupno povezanost vrhova iz A sa svim preostalim vrhovima. $\text{Veze}(B,V)$ se definira analogno. Prednost normaliziranog reza jest u tome što na nekin način penalizira podjele u kojima je omjer broja čvorova u grupama dosta velik. Dokazano je da je traženje minimalnog normaliziranog reza NP-potpun problem, čak i za specijalno strukturirane grafove [3]. Međutim, ulaganjem problema u nediskretni slučaj, zadovoljavajuća aproksimacija diskretnog problema se može pronaći dosta efikasno.

Označimo sa $d(i) = \sum_j w(i,j)$ ukupnu povezanost vrha i sa ostatkom grafa. Definiramo dijagonalnu matricu D tako da na dijagonalu postavimo vrijednosti $d(i)$. Ukoliko particiju od V poistovjetimo sa n-dimenzionalnim vektorom koji na mjestu i ima 1 ukoliko je $i \in A$, a -1 inače, naš problem možemo relativno jednostavno matrično zapisati.

$$4\text{Ncut}(A,B) = \frac{x^T(D-W)x + \mathbb{1}^T(D-W)\mathbb{1}}{k(1-k)\mathbb{1}^T D\mathbb{1}} + \frac{2(1-2k)\mathbb{1}^T(D-W)x}{k(1-k)\mathbb{1}^T D\mathbb{1}},$$

gdje je $k = \frac{\sum_{x_i>0} d(i)}{\sum_i d(i)}$, a $\mathbb{1}$ vektor jedinica.

Nizom transformacija može se pokazati da je

$$\min_{x_i=\pm 1}\text{Ncut}(x) = \min_y \frac{y^T(D-W)y}{y^T D y},$$

s uvjetima $y(i) \in \{1, -b\}$ i $y^T D\mathbb{1} = 0$.

Sada uočavanjem Rayleighevog kvocijenta i rješavanjem svojstvenog problema dobivamo

$$(D-W)y = \lambda D y, \text{tj.}$$

$$D^{-\frac{1}{2}}(D-W)D^{-\frac{1}{2}}z = \lambda z,$$

gdje je $z = D^{\frac{1}{2}}y$.

Koristeći činjenicu da se vrijednost Rayleighevog kvocijenta minimizira na najmanjem svojstvenom vektoru okomitom na uvjet [5], dobivamo da je

$$z_1 = \arg\min_{z^T z_0 = 0} \frac{z^T D^{-\frac{1}{2}}(D-W)D^{-\frac{1}{2}}z}{10 \qquad z^T z},$$

gdje su z_0 i z_1 najmanji i drugi najmanji svojstveni vektor, redom. Jedini uvjet koji sada nije zadovoljen jest da vektor y koji dobijemo ne poprima samo dvije vrijednosti, ali ukidanje tog uvjeta je uopće učinilo ovaj pristup mogućim. U ovom slučaju se odredi neka "srednja" vrijednost s te se sve komponente vektora y manje od s dodjeljuju u jednu grupu, a ostale u drugu. Postoji više načina odabira vrijednosti s, jedan je na primjer taj da se uzme podjela koja daje najmanju $Ncut$ vrijednost između n mogućih. Ovaj pristup je detaljnije opisan u [3].

Razmotrili smo kako bismo koristili normalizirani rez na našem problemu. Prije svega, kako matrica problema općenito inducira neusmjeren graf, moramo uvesti pretpostavku na njezinu simetričnost. Tada do rješenja dolazimo rekurzivnom primjenama algoritma na svaku od postojećih grupa.

Generalizacija normaliziranog reza je normalizirani k-rez. Kod tog pristupa se u svakom koraku postojeći graf V dijeli na k-dijelova tako da je minimizirana funkcija normaliziranog k-reza

$$\text{kNcut}(V_1, V_2, \ldots, V_K) = \frac{1}{k} \sum_{i=1}^{k} \frac{\text{veze}(V_i, V \backslash V_i)}{\text{veze}(V_i, V)}.$$

Jedan od aproksimativnih načina provedbe normaliziranog k-reza se zasniva na korištenju prvih k svojstvenih vektora, za razliku od normaliziranog reza kod kojeg koristimo samo 2 najmanja svojstvena vektora. Jedan način kako efikasno implementirati tu ideju je dan u [6].

Uspostavimo vezu ovog pristupa s prethodno navedena dva. Pretpostavit najprije da rješavamo problem s matricom A koja je dodatno i simetrična. Nadalje, smatramo da perturbacija neće previše izmijeniti pokazane rezultate. Provedba normaliziranog reza zahtijeva računanje svojstvenih vektora koji odgovaraju najmanjim svojstvenim vrijednostima matrice

$$D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$$

koja korištenjem svojstava matrice A prelazi u

$$I_n^{-\frac{1}{2}}(I_n - A){I_n}^{-\frac{1}{2}} = I_n - A.$$

No, sve su svojstvene vrijednosti A u segmentu $[0, 1]$. Stoga, traženje najmanjih svojstvenih vrijednosti matrice $I_n - A$ postaje traženje najvećih svojstvenih vrijednosti matrice A. Ponovno, te svojstvene vrijednosti su bliske 1.

3.4 Singularna dekompozicija matrice

SVD dekompozicija je fundamentalna matrična dekompozicija i na nekoliko se načina može koristiti u rješenju problema grupiranja.

Teorem 3.3. *Neka je $A \in \mathbb{R}^{m\times n}$. Tada postoje ortogonalna $m\times m$ matrica U, ortogonalna $n \times n$ matrica V i dijagonalna $m \times n$ matrica Σ tako da vrijedi $A = U\Sigma V^T$, gdje je*

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_n \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} \quad \textit{ili} \quad \Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \ddots & \vdots & \vdots & \cdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & \sigma_m & 0 & \cdots & 0 \end{pmatrix}.$$

Dijagonalni elementi matrice Σ su jedinstveno određeni matricom A i uređeni su tako da je[1]

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_\rho > \sigma_{\rho+1} = \cdots = \sigma_{\min(m,n)} = 0,$$

gdje je $\rho = \rho(A)$ rang matrice A. Kažemo da je $A = U\Sigma V^T$ singularna dekompozicija matrice A, ili kraće: $A = U\Sigma V^T$ je SVD[2] *matrice A.*

[1] Dijagonalni elementi od Σ se mogu pojaviti u proizvoljnom poretku, ali je padajući uređaj najjednostavniji i najelegantniji.

[2] SVD je akronim za *Singular Value Decomposition*.

Nadalje, ako su $U = [u_1, \ldots, u_m]$ *i* $V = [v_1, \ldots, v_n]$ *stupčane particije matrica* U *i* V*, onda vrijedi*

(i) Slika od A *jednaka je* $\mathcal{R}(A) = L(u_1, \ldots, u_\rho)$.

(ii) Jezgra od A *je* $\mathcal{N}(A) = L(v_{\rho+1}, \ldots, v_n)$.

(iii) $A = \sum_{i=1}^{\rho} \sigma_i u_i v_i^T$.

Dokaz ovog teorema, kao i detaljenije o SVD-u, može se pronaći u [7].

Prije nekoliko godina javio se pristup baziran na singularnoj dekompozicij matrice (SVD) i detaljno je prikazan u [8]. Razvijeni algoritam daje dobre rezultate, no u dokazu teorema koji opravdava korektnost, načinjena je pogreška. Detaljnije o pogreškama navedenim u članku moguće je čitati u [9].

Umjesto traženja svojstvenih vektora, u ovom pristupu promatraju se singularni vektori matrice B. Preciznije, promatra se drugi najveći singularni vektor kojega se potom koristi za particioniranje matrice na dvije manje. Postupak se provodi dok god svaka podmatrica, mjereno u nekoj normi, nije velika. Kada se algoritam zaustavi, rezultat bi trebali biti pronađeni blokovi.

Budući da metoda daje uglavnom zadovoljavajuće rezultate, pokušali smo na drugi način opravdati njenu korektnost. Za matricu problema $B = P(A + E)P^T$ vrijedi $BB^T = P(AA^T + (AE^T + EA^T + EE^T))P^T$. Pretpostavka da je matrica A blok-dijagonalna povlači da je i AA^T blok-dijagonalna s istom strukturom. Stoga bi matrice B i BB^T trebale skrivati istu blok strukturu. Zato matricu problema možemo zamijeniti matricom BB^T. Ovime nužno ne dobivamo stohastičku matricu, ali dobivamo simetričnu. Lako se provjeri da je BB^T stohastička, ukoliko je B bila simetrična. Iskoristimo poznatu činjenicu da su svojstvni vektori matrice BB^T upravo singularni vektori matrice B [7], čime dobivamo vezu između ove i prethodno opisane metode.

3.5 Komentar

Svi navedeni pristupi se mogu svrstati u grupu *top-down* pristupa. Oni razmatraju problem, točnije matricu problema, u cjelini. Zatim dobivaju informaciju o matrici te sam problem dijele na potprobleme. Nad tim potproblemima ponovno provode isti postupak, dok god ima smisla.

Uz sve ove pristupe, nismo uočili niti jedan *bottom-up* pristup. Takvi pristupi promatraju problem „usitnjen" na najmanje dijelove, a potom, iz koraka u korak, vežu one dijelove koji su usko povezani. Iz tog smo se razloga odlučili za razvoj metoda spomenutog tipa.

4 Novi pristup

Matrica A je blok-dijagonalna. Dodavanje matrice E, ne velike po normi, će narušiti ovo svojstvo, no matrica $A + E$ će biti blok-dijagonalno dominantna. Koristeći alate za vizualizaciju, a takav smo i mi razvili, lako se uočava gdje su prije bili blokovi. Problem nastaje tek kada se primijeni permutacija P te dobije matrica $B = P(A + E)P^T$ jer je time blok struktura sakrivena. Naš zadatak je pronaći permutaciju Q koja će poništiti djelovanje matrice P, to jest takvu permutaciju za koju će matrica

$$QBQ^T = (QP)A(QP)^T + (QP)E(QP)^T$$

ponovno imati vidljivu strukturu. Kako su elementi matrice $(QP)E(QP)^T$ jednaki elementima matrice E, no ne nužno istim pozicijama, i dalje se radi o maloj grešci. Stoga će vidljivost blok strukture matrice QBQ^T ovisiti o vidljivosti blok strukture matrice $(QP)A(QP)^T$. Nakon što imamo vidljivu strukturu, nadamo se laganom određivanju blokova. Ako ne nekim zasebnim algoritmom ili u samom postupku određivanja permutacije, onda naknadno korištenjem naše aplikacije.

4.1 Permutacije

Uz pretpostavku da su nam poznati blokovi matrice $P(A + E)P^T$, opišimo kako bismo pronašli permutaciju Q.

Neka je $\mathcal{F} = \{S_1, \ldots, S_m\}$ familija blokova matrice B. Neka je $n_k = |S_k|$ i $S_k = \{s_{k1}, \ldots, s_{kn_k}\}$, $k = 1, \ldots, m$. Definirajmo permutacijsku matricu Q koja odgovara permutaciji

$$q = [s_{11} \ \ldots \ s_{1n_1} \quad \ldots \quad s_{m1} \ \ldots \ s_{mn_m}]. \tag{2}$$

Tvrdimo da matrica QBQ^T ima vidljivu blok-strukturu i da je familija blokova upravo

$$\mathcal{F}' = \{S_1, S_2', \ldots, S_m'\}, S_1' \leq S_2' \leq \ldots \leq S_m'$$

Definirajmo $N_k = \sum_{i=1}^{k-1} n_i$. Odaberimo neke $i \in S'_k$ i $j \in S'_l$, gdje su $k, l \in \{1, \ldots, m\}$, $k \neq l$ unaprijed proizvoljno odabrani. Vrijedi $q(i) = s_{k,i-N_k}$ i $q(j) = s_{l,j-N_l}$. Tada je $[QBQ^T]_{ij} = [B]_{q(i)q(j)}$ pa se radi o vandijagonalnom elementu. Naime, indeksi retka i stupca ne pripadaju istom bloku matrice B. Zbog proizvoljnosti izbora smo pokazali da je $\mathcal{F}'$ uistinu familija blokova matrice PBP^T. Stoga, metoda koja otkriva strukturu mora izgraditi permutaciju u kojoj se navode najprije indeksi jednog bloka, potom drugog, pa trećeg i tako sve do posljednjeg bloka matrice B. Ukoliko permutacija nije ovako izgrađena, nakon primjene na B, nećemo imati u potpunosti vidljivu blok-strukturu.

4.2 Prepoznavanje blokova

U prethodnom razmatranju smo pretpostavili da su nam poznati blokovi matrice $P(A + E)P^T$. Kako to nije slučaj u problemu, bilo je potrebno osmisliti kako identificirati te blokove, to jest kako odrediti skupove S_i, $i = 1, \ldots, n$.

Prvo promatramo slučaj bez perturbacije, to jest $E = 0$. Možda se čini čudnim što tražimo blokove matrice A jer se oni „vide". Razlog leži u činjenici da računalo ne „vidi" te je potrebno osmisliti algoritam kojim će „gledati".

Svaki element matrice A različit od nule pripada nekom bloku. Neka je (i, j) pozicija nekog takvog elementa. Tada možemo zaključiti i da elementi na (j, i), (i, i), (j, j) pripadaju tom istom bloku. Nadalje, promotrimo li i-ti redak matrice A uočit ćemo da svaki element tog retka ili pripada tom bloku ili ne pripada niti jednom. Kažemo da i-ti redak „siječe" blok navedeni blok. Analogno dobivamo da j-ti redak te i-ti i j-ti stupac „sijeku" isti blok.

Navedena razmatranja dobivamo i na matrici PAP^T jer nismo nigdje koristili da su blokovi vidljivi.

4.3 Uvođenje perturbacije

Vraćanjem perturbacije, gornji postupak više nije korektan, no njenom analizom dolazimo do modifikacije gornjih zaključaka.

Vršimo analizu po komponentama. Promatrat ćemo takozvanu max-normu definiranu s $\|E\|_M := max_{i,j}|[E]_{ij}|$. Riječima, vrijednost ove norme jest najveća apsolutna vrijednost komponenti matrice E. Pretpostavimo da je dan $\varepsilon \geq 0$ za kojeg je $\|E\|_M \leq \varepsilon$. Neka je (i, j) pozicija proizvoljnog elementa matrice B koji ne pripada bloku. Tada je

$$|b_{ij}| = |a_{p(i)p(j)} + e_{p(i)p(j)}| = |e_{p(i)p(j)}| \leq \varepsilon.$$

Zato svi elementi strogo veći od ε pripadaju nekom bloku. Kada je $\varepsilon = 0$, zaključci dobiveni na ovaj način su identični onima u prethodnoj analizi. Možemo ustvrditi kako smo dobili poopćenje prethodnih zaključaka na situacije u kojima je poznata gornja ograda za vrijednost $\|E\|_M$.

Ograda ε nam neće biti unaprijed poznata, a želja nam je da razvijena metoda bude robustna, to jest otpornija na što je moguće veće greške. Zato želimo bit pažljivi prilikom proglašavanja pripadnosti nekog elementa bloku. Pretpostavimo da smo za neki element zaključili da pripada nekom bloku. On je strogo veći od ε, ali nije veći od najvećeg elementa matrice B. Zaključujemo da tada i najveći element matrice B pripada nekom bloku.

Razmotrimo slučaj u kojem najveći element matrice B ne pripada bloku. Tada možemo definirati $E' = \varepsilon \mathbb{1}_{n\times n} - A$ za koju je $\|E'\|_M \leq \varepsilon$. No, matrica $A + E'$ ima sve elemente jednake čime je izgubljena svaka informacija o blok-dijagonalnoj strukturi matrice A. Zato ovakve vrijednosti ε smatramo suviše velikim i ne očekujemo da bi neka metoda takvu strukturu mogla restaurirati.

Neka imamo skup elemenata koji mogu pripadati nekom bloku. Zbog do sada navedenoga, mi ćemo samo za najveći element tog skupa reći da pripada nekom bloku.

4.4 Zapis metode

Preostaje nam objediniti dobivene zaključke u metodu koja rješava problem.

Izaberimo neki indeks $i \in \{1, \ldots, n\}$. Tada i-ti redak siječe neki blok, recimo S_k. Pretpostavimo da imamo neki skup $S \subsetneq S_k$. Promotrimo sve retke i stupce čiji se indeksi nalaze u S. Oni sijeku blok pa u tim retcima i stupcima tražimo element koji pripada bloku S_k, a za kojeg to prije nismo ustvrdili. Rekli smo da ćemo birati najveći element kada iz ponuđenog skupa moramo odabrati neki element koji pripada bloku. Na taj smo način dobili novi indeks kojim proširujemo S. Ovaj postupak ponavljamo sve dok ne izgradimo blok S_k. Poznavanje koraka u kojem je blok izgrađen nam nije bitno, jednostavno ćemo nastaviti dalje s opisanim postupkom.

U prvom sljedećem koraku proglašavamo indeks nekog drugog bloka, recimo S_l, dijelom bloka S_k. Ovo je pogrešno, no nastavimo li dalje, sve elemente bloka S_l ćemo proglasiti elementima bloka S_k. Stoga, izgradimo li permutaciju (2), indeks retka/stupca za kojeg smo u i-tom koraku zaključili da pripada bloku S_k ovu „grešku" nećemo primijetiti. Naime, prvo će biti navedeni indeksi bloka S_k, potom bloka S_l, a potom i narednih blokova, a to je ono što smo htjeli. Preostaje nam još zapisati pseudokod metode.

Algoritam 1 Osnovna metoda

Neka je $b_{i_0 j_0}$ najveći element matrice B
$S = \{1, \ldots, n\} \setminus \{i_0\}$
$Bio = \{i_0\}$
$p = [i_0]$
while S neprazan **do**
    Neka je b_{ij} najveći element matrice B sa pozicija $S \times Bio \cup Bio \times S$
    Neka je $k \in \{i, j\}$ koji nije u Bio
    $S = S \setminus \{k\}$
    $Bio = Bio \cup \{k\}$
    $p = [p\ k]$
end while

5 Veza s teorijom grafova

Promotrimo skupove S i Bio u nekom koraku metode. Tada vrijedi

$$\max_{(i,j)\in S\times Bio\cup Bio\times S} [B]_{ij} = \max_{(i,j)\in S\times Bio} \max\{[B]_{ij}, [B]_{ji}\}.$$

Zato je rezultat provođenja metode na matrici B jednak rezultatu provođenja iste na matrici B'. Matrica B' je definirana s $[B']_{ij} = \max\{[B]_{ij}, [B]_{ji}\}$, $i, j = 1, \ldots, n$. Jasno jest da je ona simetrična te da vrijedi $B' = B$ ako je B simetrična. Time u algoritmu 1 matricu B implicitno mijenjamo matricom B'. U [8] je prikazana eksplicitna zamjena matrice B simetričnom matricom BB^T. Zanimljivo jest da je najbliža matrici B najbliža simetrična matrica upravo $\frac{1}{2}(B + B^T)$ [7], no nismo primijetili metodu koja bi ovu informaciju koristila. Napomenimo još da se lako provjera kako sve tri spomenuti matrice imaju vidljivu blok strukturu ako i samo ako ju ima i B.

Promotrimo grafove koje induciraju matrice B i B'. Matrica problema općenito inducira usmjeren graf G_B, dok matrica B' inducira neusmjeren graf $G_{B'}$. Već smo uspostavili vezu između B i B' pa učinimo to sada za G_B i $G_{B'}$. Između svakog para vrhova grafa G_B imat

ćemo najviše dva usmjerena brida. Te bridove ćemo zamijeniti jednim neusmjerenim čija će težina odgovarati najtežem usmjerenom bridu. Graf koji tako dobivamo je upravo $G_{B'}$. Proučavanje ponašanja algoritma na matrici B sada možemo lako zamijeniti proučavanjem ponašanja na grafu $G_{B'}$. Za to je samo potrebno interpretirati liniju 6 algoritma 1 kao traženje najtežeg brida čiji je jedan vrh u skupu Bio, a drugi u skupu S. Tako opisan algoritam za grafove se naziva Primov algoritam [10]. Definirajmo osnovne pojmove iz problematike koju ovaj algoritam rješava.

Definicija 5.1.

Stablo *je graf koji je povezan i ne sadrži cikluse.*

Razapinjuće stablo *povezanog grafa G jest stablo koje ima iste vrhove kao i G, a skup bridova mu je podskup bridova grafa G.*

Maksimalno razapinjuće stablo *težinskog grafa G_ω jest razapinjuće stablo koje, između svih razapinjućih stabala grafa G_ω, ima najveću težinu.*

Sada možemo rezimirati kako naša metoda zapravo pronalazi maksimalno razapinjuće stablo grafa induciranog s B', a permutaciju gradi redoslijedom kojim dodaje nove vrhove u buduće razapinjuće stablo. Time smo uspostavili vezu između našeg problema i problema maksimalnih razapinjućih stabala. Nadalje, otvorili smo vrata novim idejama, ali i već postojećim alatima koji postoje u teoriji grafova. Kako uz Primov postoje još dva poznata algoritma – Kruskalov i Boruvkin [10], odlučili smo odabrati jedan od njih i pokušali ga prilagoditi rješavanju našeg problema. Riječ je o Kruskalovom algoritmu.

5.1 Kruskalov algoritam

Opišimo ukratko Kruskalov algoritam. Započinjemo s n stabala, svaki vrh predstavlja jedno. U koraku algoritma, dok god imamo barem dva stabla, biramo najteži brid u grafu čiji se vrhovi nalaze u različitim stablima. Spojimo dva stabla pronađenim bridom te nastavljamo postupak. Rezultat je jedno, maksimalno razapinjuće, stablo.

Vratimo se sada s grafova na matrice. Stabla promatramo samo kao skup vrhova. Svaki od tih skupova je na početku jednočlan i podskup nekog bloka. Traženje najtežeg brida koji spaja dva različita stabla postaje traženje najvećeg elementa matrice za kojeg još ne znamo da li pripada bloku. Ako je on „velik" spajanje dva stabla tumačimo kao spajanje dva dijela bloka u jedan. Ako je on „malen", svi nakon njega neće biti teži. Stoga smatramo da su svi blokovi pronađeni, a daljnje spajanje će grupirati blokove po veličini najvećeg vandijagonalnog elementa. Permutaciju gradimo istovremeno sa spajanjem stabala. Prilikom spajanja dva stabla, osiguravamo da su prvo navedeni vrhovi jednog stabla nakon čega neposredno slijede vrhovi drugog stabla. Rezultirajuća permutacija bi tada trebala imati oblik kao (2). Naposlijetku, zapisujemo pseudokod Kruskalovog algoritma za matrice.

Algoritam 2 Prilagođeni Kruskal

1: $S_k = \{k\}$, $k = 1, \ldots, n$
2: $p_k = [k]$, $k = 1, \ldots, n$
3: $blokova = n$
4: **while** $blokova > 1$ **do**
5: Neka je b_{ij} najveći element matrice B sa pozicija $\bigcup_{k \neq l} S_k \times S_l$
6: $S_i = S_i \cup S_j$
7: $p_i = [p_i \; p_j]$
8: Obriši S_j
9: Obriši p_j
10: $blokova = blokova - 1$
11: **end while**

5.2 Analiza rada nove metode

U knjizi [11], između ostaloga, možemo čitati o povezanosti razapinjućih stabala i Kruskalovog algoritma s grupiranjem objekata prema njihovoj udaljenosti. Iako se u našem

slučaju ne radi o udaljenostima, prikazane rezultate je moguće prilagoditi našem problemu. Stoga iskazujemo i dokazujemo teorem koji daje dovoljne uvjete za uspješan rad razvijenih metoda.

Teorem 5.1. *Neka je dana matrica B, neka je $\mathcal{F} = \{S_1, \ldots, S_m\}$ njezina particija blokova. Neka je G graf induciran s B, a $G_1, \ldots, G_m$ grafovi inducirani odgovarajućim podmatricama. Neka su $T_1, \ldots, T_m$ proizvoljna maksimalna razapinjuća stabla odgovarajućih grafova. Ako za svaki $k = 1, \ldots, m$ vrijedi*

$$\min \omega\left(E(T_k)\right) > \max \omega\big(\{(u,v) \in V(G_k) \times V(G_k)^c \cup V(G_k)^c \times V(G_k)\}\big),$$

tada obje naše metode otkrivaju $\mathcal{F}$.

Dokaz (algoritam 1): Definirajmo $n_k = |S_k|$, $k = 1, \ldots, m$. Neka je $i \in \{1, \ldots, n\}$ proizvoljan i $k \in \{1, \ldots, m\}$ takav da je $i \in S_k$. Pretpostavimo da algoritam započinje svoje izvođenje u čvoru i. Pogledajmo prvih $n_k - 1$ koraka, nakon čega ćemo imati izgrađenih prvih n_k vrijednosti vektora permutacije $p = [i_1 \ \ldots \ i_{n_k}]$. Pokažimo da svi i_j, $j \in \{1, \ldots, n_k\}$ pripadaju istom bloku. Pretpostavimo li suprotno, možemo izabrati najmanji $j\{1, \ldots, n_k\}$ za kojeg $i_j \notin S_k$. Tada je $i_1, \ldots, i_{j-1} \in S_k$, a brid $e = (i_{j-1}, i_j)$ spaja blok S_k s nekim drugim blokom. Stablo T_i je povezano pa postoji brid koji spaja $\{i_1, \ldots, i_{j-1}\}$ sa $S_k \setminus \{i_1, \ldots, i_{j-1}\}$. Označimo ga s e' te primijetimo da je

$$\begin{aligned} \omega(e') &\geq \min \omega(E(T_k)) > \\ &> \max \omega\big(\{(u,v) \in V(G_i) \times V(G_i)^c \cup V(G_i)^c \times V(G_i)\}\big) \geq \omega(e) \end{aligned}$$

što nije moguće. Dakle, uz uvjet teorema, algoritam u n_k koraka otkriva cijeli blok S_k. U sljedećem koraku dobiva indeks nekog novog bloka kojeg dodaje na kraj dosad izgrađenog vektora permutacije. Kako algoritam više ne promatra veze sa S_k, gornji dokaz lako ponovimo za blok kojem novi indeks pripada, a potom i za sve naredne. Rezultat algoritma je permutacija koja otkriva blok strukturu, pa je teorem dokazan.

■

Dokaz (algoritam 2): Zaustavimo algoritam u nekom koraku. Pretpostavimo da svi vektori p_i zadovoljavaju jedan od sljedeća dva uvjeta:

- pojavljuju se samo indeksi jednog bloka
- pojavljuju se svi indeksi nekog broja blokova, pri čemu oni iz istog bloka dolaze uzastopce

Ova tvrdnja je zadovoljena na samom početku algoritma, a tvrdimo da će biti zadovoljena i nakon provođenja narednog koraka. To neće biti slučaj jedino ako spajamo stablo čiji skup vrhova ne čini cijeli blok s nekim drugim stablom. No, isto rezimiranje o težini bridova kao i u dokazu za algoritam 1 pokazuje da to nije moguće. Kako u svakom koraku vrijedi jedan od dva uvjeta, zaključujemo da će u posljednjem koraku vrijediti drugi uvjet. Stoga izgrađenom permutacijom otkrivamo blok strukturu i time završavamo dokaz.

■

Bitno je naglasiti da dokaz teorema ne ovisi o izboru stabala $T_1, \ldots, T_m$. Razlog leži u tome što težina najlakšeg brida MST-a grafa G_i ne ovisi o izboru MST-a. Ovo je posljedica sljedećeg teorema, čiji je dokaz pak neposredna posljedica rezultata dobivenih u [12].

Teorem 5.2. *Neka je dan graf G te njegova dva MST-a T i T'. Za svaki $\omega_0 \in \mathbb{R}^+$ vrijedi da je broj bridova težine ω_0 u T i T' jednak.*

Promotrimo neki blok S_k veličine $n_k = |S_k|$. MST grafa induciranog s tim blokom ima $n_k - 1$ bridova. Intuicija nam kaže kako bi tada i najlakši brid MST-a trebao biti velik u odnosu na ostale elemente bloka. Da bi neki element van blok-dijagonale bio veći od njega, greška po komponentama bi morala biti velika. Kako smo već spomenuli, tada bi informacija o blok-strukturi trebala biti nepovratno izgubljena. Stoga pokazani rezultat ide u prilog kvaliteti ovih metoda.

6 Blok-Kruskalov algoritam

Iako bismo prema navedenome mogli zaključiti kako će razvijeni algoritmi biti dovoljni za rješavanje problema, njihov problem je nedostatak robustnosti. Naime, samo jedno pogrešno zaključivanje može narušiti rezultat, a nekoliko pogrešnih može ga znatno ugroziti. Stoga u koraku Kruskalovog algoritma nećemo tražiti najteži brid koji spaja dva nespojena stabla. Umjesto toga, tražit ćemo dva nespojena stabla između kojih postoji „najjača veza". Pojam „najjača veza" ne definiramo, ali zahtjevamo da ovisi samo o dva promatrana stabla i bridovima između njih. Na taj način dobivamo općeniti Blok-Kruskal algoritam, koji bi se po potrebi moga prilagođavati problemima tipa sličnog kao i naš. Ovdje dajemo pseudokod predloška algoritma.

Algoritam 3 Blok-Kruskal

1: $S_k = \{k\}$, $k = 1, \ldots, n$
2: $p_k = [k]$, $k = 1, \ldots, n$
3: $blokova = n$
4: **while** $blokova > 1$ **do**
5: Neka su $i \neq j$ indeksi blokova između kojih postoji najjača veza
6: $S_i = S_i \cup S_j$
7: $p_i = [p_i\ p_j]$
8: Obriši S_j
9: Obriši p_j
10: $blokova = blokova - 1$
11: **end while**

Ovako zapisan algoritam, korištenjem svojstava Markovljevih lanaca, prilagođavamo našem problemu.

6.1 Interpretacija „veze" Markovljevim lancima

Neka je $(X_t)_{t\in\mathbb{N}}$ Markovljev lanac kojemu je B matrica prijelaza. Neka su S_k, S_l neka dva skupa stanja. Vjerojatnost prelaska lanca iz skupa stanja S_k u skup stanja S_l u vremenskom koraku t iznosi:

$$\begin{aligned} P(X_{t+1} \in S_l \mid X_t \in S_k) &= \frac{\sum_{j\in S_l, i\in S_k} P(X_{t+1} = j, \mid X_t = i)P(X_t = i)}{\sum_{i\in S_k} P(X_t = i)} \\ &= \frac{\sum_{j\in S_l, i\in S_k} [B]_{ij} P(X_t = i)}{\sum_{i\in S_k} P(X_t = i)}. \end{aligned}$$

Problem s ovim izrazom jest što on ovisi o distribuciji lanca u trenutku t. Ona pak ovisi o početnoj distribuciji koja unaprijed nije poznata. Uz oznaku π za stacionarnu distribuciju lanca, za dovoljno velike t vrijedi

$$P(X_{t+1} \in S_l \mid X_t \in S_k) \approx \frac{\sum_{j\in S_l, i\in S_k} [B]_{ij}\pi_i}{\sum_{i\in S_k} \pi_i}$$

što ne ovisi o polaznoj distribuciji. Ovo možemo opravdati. U slučaju kada je Markovljev lanac aperiodičan i ireducibilan te ima stacionarnu distribuciju π, postoji granična distribucija i jednaka je stacionarnoj.

$$\lim_{t\to\infty} P(X_t = j) = \pi_j, \forall i, j \in S$$

Smijemo pretpostaviti da je lanac ireducibilan, jer kad ne bi bio, to bi povlačilo i da je naša matrica problema reducibilna te bi problem mogli trivijalno svesti na dva manja ireducibilna problema. Stacionarna distribucija postoji jer je lanac ireducibilan i konačan. To smo već objasnili u teoretskom dijelu o Markovljevim lancima. Pretpostavku na aperiodičnost ne možemo striktno opravdati ali je dosta intuitivna i postoje dosta slabi dovoljni uvjeti za nju. Na primjer, dovoljno je da je jedan element na dijagonali različit od 0 kako bi ireducibilan konačan lanac bio aperiodičan, što nije presmiono za pretpostaviti posebno u našem problemu gdje bi trebali imati nekakvu blok dijagonalnu strukturu.

Preostaje nam još iskoristiti ove rezultate za definiranje pojma „jačina veze" u Blok-Kruskal algoritmu. Jačinu veze želimo mjeriti vjerojatnošću prijelaza iz skupa S_k u S_l,

ali i obratno. Stoga pojam „jačina veze" definiramo kao produkt

$$\frac{\sum_{j\in S_l, i\in S_k}[B]_{ij}\pi_i}{\sum_{i\in S_k}\pi_i}\cdot\frac{\sum_{i\in S_k, j\in S_l}[B]_{ji}\pi_j}{\sum_{j\in S_l}\pi_i}$$

6.2 Implementacija

Naivni pristup implementaciji algoritma 3 ima složenost $\mathcal{O}(n^3)$. Na većim matricama, to je velika cijena koju je potrebno platiti. Vremensku složenost implementacije koju smo mi izveli korištenjem indeksirane hrpe [10] u grubo možemo ocijeniti s $\mathcal{O}(n^2\lg(n))$, dok je memorijska složenost $\mathcal{O}(n^2)$ kao posljedica pohrane indeksa. Daljnja poboljšanja su moguća u vidu d-hrpe [13], pametnije organizacije indeksa i sličnog.

U kratkim crtama dajemo opis implementacije gornje metode. Prvi korak je izgradnja hrpe čiji su elementi trojke (v, i, j), gdje je v vrijednost ovisna o blokovima S_i i S_j. Bez smanjenja općenitosti koristimo $i < j$. U matrici H je na mjestu (i, j) pohranjena pozicija trojke (v, i, j) u indeksiranoj hrpi. U koraku algoritma uzimamo trojku (v, i, j) s vrha hrpe, gradimo blok $S_i \cup S_j$ i izbacujemo trojku iz hrpe. Koristeći H obnavljamo hrpu - spajamo blokove nastale u prethodnom koraku, brišemo nepotrebne trojke, rekonstruiramo strukturu hrpe.

7 Metaheuristički pristup

Kako je zadani problem dosta općenit i teško je reći što stvarno je rješenje, jedna od ideja koje smo se dosjetili su i heurističke metode. Nakon istraživanja i analize metoda koje bi mogli koristiti, metoda simuliranog kaljenja nam se učinila kao jedna od najprilagodljivijih i najuniverzalnijih.

7.1 Simulirano kaljenje - opis metode

Simulirano kaljenje [14] (eng. *Simulated annealing* - **SA**) je jedna od metaheurističkih metoda koja je primjenjiva na razne probleme. Ime i ideja potječu iz metalurgije, gdje se tehnikom grijanja i kontroliranog hlađenja povećavaju kristali u materijalu, te smanjuju defekti. Toplina omogućuje atomima iskakanje iz njihovih početnih pozicija (lokalnih minimuma unutrašnje energije) i "putovanje" kroz stanja viših energija. Sporije hlađenje povećava vjerojatnost pronalaska stanja sa manjom unutrašnjom energijom od one početne. Analogno fizikalnom procesu, svaki korak SA algoritma zamijenjuje trenutno stanje sa slučajnim "bliskim" stanjem, odabranim sa vjerojatnošću koja ovisi o razlici energetskih vrijednosti između ta dva stanja, te globalnom parametru T (temperaturi), koji se postupno smanjuje tijekom procesa. Ovisnost o temperaturi je takva da se trenutno stanje na početku kad je T velik mijenja gotovo potpuno slučajno, a kako se T smanjuje većinom se odabiru samo "bolja" stanja od trenutnog. Dopuštanje metodi odlaske u "lošija" stanja potencijalno ju spašava od zaglavljivanja u lokalnom minimumu, što je velika mana svih pohlepnih metoda. Uspješnost metode simuliranog kaljenja ovisi o izboru pojedinih parametara kao što su promjer grafa stanja, prijelazne vjerojatnosti između stanja, odabir kandidata tj. susjednih stanja te rasporedu hlađenja.

7.2 Simulirano kaljenje - teorijska pozadina

Prvo ćemo definirati par pojmova vezanih uz teoriju algoritma simuliranog kaljenja.

Definicija 7.1. ***Rasporedom hlađenja*** *zovemo padajući niz strogo pozitivnih realnih brojeva takvih da je* $lim_{k\to\infty} T_k = 0$. *Pojedini element tog niza* T_k *nazivamo* ***temperaturom*** k*-tog trenutka.*

Definicija 7.2. ***Susjedstvo*** *stanja* x *jest skup svih onih stanja u koje se može doci iz stanja* x. *Označavamo ga sa* $N(x)$.

Sada kada smo opisali dva osnovna pojma možemo objasniti kako se proces simuliranog kaljenja uklapa u teoriju Markovljevih lanaca. Skup stanja S u Markovljevom lancu je jednak onim stanjima u kojima proces simuliranog kaljenja može dospjeti. Prijelazne vjerojatnosti između stanja ovise o temperaturi kao što smo to već ranije napomenuli. U k-tom koraku vrijedi:

$$\begin{aligned} P_k(x,y) &= P(X_{k+1} = y, X_k = x) \\ &= \begin{cases} 0 & y \notin N(x) \text{ i } y \neq x, \\ R(x,y)exp(-(V(y)-V(x))^+/T_k) & y \in N(x) \text{ i } y \neq x, \\ 1 - \sum_{z\neq x} P_k(x,z) & x = y \end{cases} \end{aligned}$$

s time da sa $R(x,y)$ označavamo vjerojatnost odabira y među svim susjedima od x, a sa $V(x)$ energiju stanja x.

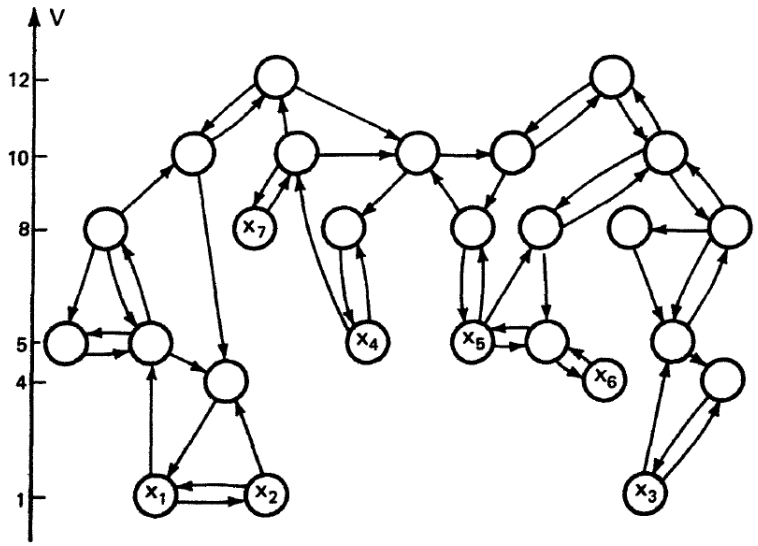
Time je proces simuliranog kaljenja opisan kroz teoriju Markovljevih lanaca. Nadalje definiramo još poneke pojmove koji su bitni za analizu metode.

Definicija 7.3. *Stanje* y *je* ***dostupno na visini*** E ***iz*** x *ako postoji niz* $(x = x_0, x_1, \ldots, x_p = y)$ *takav da* $x_{k+1} \in N(x_k)$ *i* $V(x) \leq E$ *za sve* $k \in \{0, 1, 2, \ldots, p\}$, *gdje smo sa* $V(x)$ *označili energiju stanja* x.

Definicija 7.4. *Kažemo da je proces* ***slabo reverzibilan*** *ako za svaki realan broj* $E > 0$ *i svaka dva stanja* x *i* y *vrijedi da je* x *dostupan iz* y *na visini* E *ako i samo ako je* y *dostupan iz* x *na visini* E.

Definicija 7.5. *Stanje x je **lokalni minimum** ako ne postoji stanje y takvo da je $V(y) < V(x)$ i y je dostupno iz x na visini $V(x)$.*

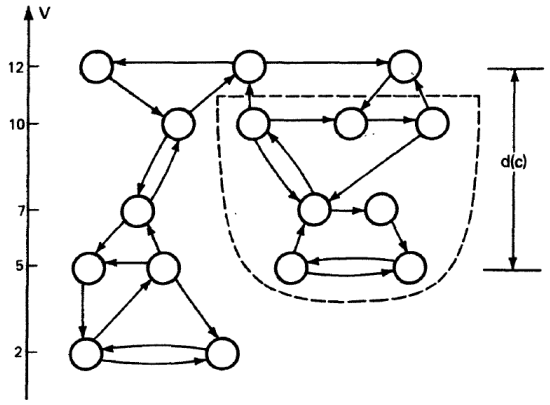
Definicija 7.6. ***Dubina lokalnog minimuma*** *x (u oznaci $d(x)$) je najmanji realan broj $E > 0$ takav da postoji stanje y sa svojstvom da $V(y) < V(x)$ i y je dostupno iz x na visini $V(x) + E$. Dubinu globalnog minimuma definiramo da je $+\infty$.*



Slika 1: Stanja x_1, x_2 i x_3 predstavljaju globalne minimume. x_4, x_6 i x_7 su lokalni minimumi dubina $5, 6$ i 2. x_5 nije lokalni minimum. x_2 je dostupno na visini 1 iz x_1. x_3 je dostupno na visini 12 iz x_1. (Slika preuzeta iz [15].)

Definicija 7.7. ***Udolina*** *C je skup stanja takvih da za neki zadani broj $E > 0$ i stanje x vrijedi $C = \{y : y$ dostupan iz x na visini $E\}$. **Dno udoline** je skup stanja C sa minimalnom energijom.*

Sada možemo iskazati osnovni teorem [15] koji jamči konvergenciju procesa simuliranog kaljenja.



Slika 2: Označeni dio predstavlja jednu udolinu C. Njena dubina je jednaka $d(C) = 7$, a dno joj se sastoji od 2 stanja. (Slika preuzeta iz [15].)

Teorem 7.1. *Za zadani skup stanja S, pripadne energetske vrijednosti V, susjedstva svih stanja N te raspored hlađenja $(T_n)_{n\in\mathbb{N}}$ koji je ireducibilan i slabo reverzibilan vrijedi: (a) Za svako stanje koje nije lokalni minimum vrijedi*

$$\lim_{k\to\infty} P(X_k = x) = 0.$$

(b) Pretpostavimo da je B dno udoline dubine d te da su stanja u B lokalni minimumi dubine d. Tada je

$$\lim_{k\to\infty} P(X_k \in B) = 0,$$

ako i samo ako je

$$\sum_{k=1}^{\infty} \exp(-d/T_k) = \infty.$$

(c) Neka je S^* skup globalnih minimuma, a d^* maksimum dubina svih lokalnih minimuma koji nisu globalni minimumi. Tada

$$\lim_{k\to\infty} P(X_k \in S^*) = 1,$$

ako i samo ako

$$\sum_{k=1}^{\infty} \exp(-d^*/T_k) = \infty.$$

Dokaz ovog teorema je jako kompleksan i može se vidjeti u članku [15]. Važnost ovog teorema je u tome što nam jamči da metoda simuliranog kaljenja ukoliko joj omogućimo dovoljno velik broj koraka konvergira globalnom minimumu. Nažalost, nije moguće eksplicitno utvrditi koliki je taj broj koraka za koji metoda konvergira globalnom minimumu.

8 Simulirano kaljenje u našem problemu

8.1 Modeliranje

Sada ćemo opisati kako smo naš problem preveli u jezik metode simuliranog kaljenja te kako smo odabrali pojedine parametre.

- **Stanja**: U našem problemu, trenutno stanje u kojem se proces nalazi će, naravno, biti permutacija i pripadna blok struktura, tj. particija redaka na klase koje pripadaju istom bloku.

- **Energetska vrijednost stanja**: Funkcija kojom ćemo mjeriti koliko je određeno stanje dobro ili loše. U našem slucaju je to prosjek elemenata u bloku na dijagonali.

- **Skup susjednih stanja**: Kako bi pojednostavili jednu iteraciju metode, što omogućuje izvršavanje većeg broja iteracija, odabrali smo varijantu, gdje se do proizvoljnog susjednog stanja dolazi manjim i jednostavnijim operacijama. Dopuštene operacije su zamjena redoslijeda dva susjedna bloka redaka, spajanje dva susjedna bloka redaka u jedan te rastavljanje određenog bloka na dva bloka manjih veličina.

- **Raspored hlađenja**: Raspored hlađenja je usko vezan uz prijelazne vjerojatnosti kao što smo već pojasnili u teoretskom dijelu. Odabrali smo da nam raspored hlađenja bude geometrijski niz realnih brojeva s početkom otprilike u 1 i krajem otprilike u 0.

- **Prijelazne vjerojatnosti**: Zamjena dva uzastopna bloka se uvijek odvija, jer ne postoji način kako odrediti vodi li ta promjena točnom rješenju ili ne, te nema razloga da ju ne napravimo. Vjerojatnost dijeljenja jednog bloka S na dva bloka S_1 i S_2, je jednaka suprotnoj vjerojatnosti spajanja blokova S_1 i S_2 u S. Time smo osigurali simetričnost našeg problema. Promotrimo sada jednu 2×2 blok-matricu A. Za nju ćemo opisati kako određujemo treba li A_{11} i A_{22} spojiti u jedan blok ili

matricu A podijeliti na 2 bloka A_{11} i A_{22}.

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

Vjerojatnost određujemo izračunavanjem prosjeka elemenata u dijagonalnim blokovima A_{11} i A_{22}, te prosjeka elemenata u cijeloj 2×2 blok-matrici. Prirodno je zahtjevati ukoliko A_{11} i A_{22} pripadaju istom bloku da je onda prosjek vandijagonalnih elemenata njihovog reda veličine, to jest da su prosjeci elemenata u njima otprilike jednaki. Tu ovodimo novu konstantu C. Kažemo da je spajanje dobro, i sigurno će se dogoditi, ukoliko je prosjek vandijagonalnih elemenata barem C puta veći od prosjeka u dijagonalnim elementima. Prirodno je naravno uzeti konstantu $C \leq 1$, ali nije nužno. Važno je napomenuti, da se promjene "na bolje" događaju uvijek, a provođenje promjena "na gore" ovisi o trenutnoj temperaturi cijelog sustava.

Pokažimo još da naš model procesa simuliranog kaljenja zadovoljava uvjete teorema kojeg smo prije iskazali (do na izvršavanje u beskonačno mnogo iteracija). Skup stanja, susjedstva i energetske vrijednosti svakog od stanja smo definirali. Naš niz temperatura konverigira u 0. Zbog odabira prijelaznih vjerojatnosti tako da proces bude simetričan on očito zadovoljava uvjet slabe reverzibilnosti. Time su svi uvjeti teorema ispunjeni.

8.2 Algoritam metode

Algoritam 4 Simulirano kaljenje

1: A je matrica problema
2: inicijaliziraj na početno stanje sustava - svaki dijagonalni element određuje blok za sebe
3: inicijaliziraj temperaturne uvjete, $T = 1$
4: **while** $T > 0$ **do**
5: odaberi tip promjene koji će pokušati napraviti
6: odaberi elemente na kojima ćes probati napraviti promjenu
7: izračunaj razliku energija prije i poslije promjene te pripadnu vjerojatnost promjene P
8: sa vjerojatnosću P provedi promjenu
9: **end while**

Ova metoda na kraju ima zapamćenu blok strukturu u svakom trenutku njenog izvođenja te i to može biti jedan od izlaznih parametara.

8.3 Modifikacije i druge ideje

Prva sljedeća modifikacija koju smo napravili jest pokretanje procesa mutiranja više puta zaredom za različite vrijednosti koeficijenta C. Za koeficijente smo uzimali padajuće nizove brojeva (na primjer od 1 do 0.6). Za svaku sljedeću vrijednost koeficijenta C, početna pozicija bi bila krajnje stanje procesa sa prethodnom vrijednošću koeficijenta. Time smo postigli da se prvo dogadađaju one "najbolje promjene", ali ujedno ostavili i mogućnost promjena unatrag. Pokazalo se u testiranju da se te promjene unatrag baš i ne događaju.

Na to se onda prirodno nadovezuje i modifikacija u kojoj smo nakon završetka procesa za pojedini koeficijent C, učvrstili promjene koje su se dogodile, te svaki od postojećih

blokova saželi u jedno stanje, te time postigli da nam se matrica problema smanjila, a samim time i izvršavanje algoritma ubrzalo. Točnost se nije pokvarila, čak se i poboljšala, a brzina rada se umnogostručila.

Postojala je ideja da se umjesto ovakvog pristupa i odabira funkcije cilja, odaberemo funkciju cilja baziranu na udaljenostima redaka unutar blokova. Uz pretpostavku da su (stohastički) retci matrice uniformno distribuirani, relativno lako se mogu izračunati i statistički parametri za blok veličine k, očekivanje i standardnu devijaciju udaljenosti 2 retka u bloku veličine k. Ideja je bila da će u optimalnom rasporedu retci koji bi trebali biti u istom bloku većinom biti dovoljno blizu jedan drugome. Nažalost, nismo uspjeli to sprovesti u djelo. Metoda je davala dosta loše rezultate. Pretpostavljamo da je probleme predstavljala činjenica da retci ipak nisu baš uniformno odabrani te je ta greška u potpunosti pokvarila metodu.

Isprobali smo i model algoritma simuliranog kaljenja u kojoj umjesto promjena na susjednim elementima, dopuštamo spajanje udaljenih blokova. To je naravno dovelo do toga da operacija zamjena poretka gubi svoj smisao, te ju nismo u ovom slučaju niti implementirali. Prednost ove implementacije jest znatno manji promjer grafa traženja što je naravno dobra stvar. Ali kako to uvijek biva, kako je i spomenuto ranije, jedan korak algoritma u ovoj implementaciji ima veću složenost te je znatno sporiji. Probleme prouzročava nemogućnost pronalaska prikladne strukture podataka koja će pamtiti međurezultate i njih efikasno osvježavati prilikom svake promjene uzimajući u obzir da se promjena može dogoditi na najudaljenijim dijelovima matrice. Ovaj model ima također svoje modifikacije kao i prethodni, kao što su odabir više koeficijenata C te (ne)učvršćivanje promjena iz prethodnog niza mutacija.

9 Rezultati

9.1 Generiranje testnih problema

Radi testiranja naših metoda, morali smo sami generirati testne primjere problema na kojima ćemo ih testirati. To je vrlo delikatno, jer često o kvaliteti primjera na kojima se metode testiraju ovisi i konačna kvaliteta metoda, tj. uklanjanje grešaka i potencijalnih problema u njima.

Stvari koje smo trebali odabrati za generiranje vjerodostojnih test primjera jesu:

1. Za zadanu veličinu matrice problema n kako odabirati k - broj blokova na dijagonali blok dijagonalne stohastičke matrice A
2. Za zadani broj dijagonalnih blokova kako odrediti njihove veličine - $|A_1|, |A_2|, \ldots, |A_k|$
3. U svakom od dijagonalnih blokova odrediti pojedine elemente bloka tj. odrediti $A_1, A_2, \ldots, A_k$
4. Za zadanu maksimalnu gresku ϵ odrediti prihvatljivu distribuciju grešaka te matricu greške E

Model 1

Istraživajući nama dostupne materijale te metodom pokušaja i pogrešaka odlučili smo se za sljedeći način generiranja testnih primjera problema slično kao u [16]:

1. Broj dijagonalnih blokova smo određivali formulom $k = 2 + \lfloor \ln(\text{random}(n) + 1) \rfloor$. Koeficijent 2 smo stavili u formulu jer problem nema smisla za broj blokova manji od 2. Koeficijent 1 pod logaritamskom funkcijom smo dodali kako bi osigurali da je rezultat logaritamske funkcije uvijek pozitivan. Odlučili smo se logaritamsku funkciju jer daje rezultat relativno mali s obzirom na n što nam se činilo dosta pogodno za kvalitetne testne primjere

2. Za velicine blokova $|A_1|, |A_2|, \ldots, |A_k|$ smo slučajno odabrali jednu od točaka iz skupa $\{\mathbf{x} \in \mathbb{N}^k : \sum_{i=1}^{k} x_i = n\}$ s time da se svaka točka ravnine izabire jednako vjerojatno.

3. Elemente pojedinog bloka A_i smo određivali tako da smo generirali redak po redak matrice A_i, jer na svaki redak imamo zaseban uvjet da je stohastički. Pojedini redak smo odabrali slučajno iz skupa $\{\mathbf{x} \in \mathbb{R}^{|A_i|} : \sum_{i=1}^{|A_i|} x_i = 1, x_i \geq 0\}$

4. Svakom od blokova E_{ij} matrice greške E, dodijelili smo pripadnu "težinu" ϵ_{ij} kao slucajan broj iz intervala $[0, 1]$ te svaki element tog bloka smo postavili na slučajni element iz intervala $[0, \epsilon_{ij}]$ sa uniformnom razdiobom. Na kraju skaliramo svaki redak matrice E tako da suma svakog retka bude jednaka 1.

Na kraju ovog procesa generiramo matricu C kao konveksnu kombinaciju matrica A i E, kako bi osigurali da je C stohastička.

$$C = (1 - \epsilon)A + \epsilon E$$

Naravno na kraju ovog procesa, na dobivenu matricu C, primijenimo slučajnu matricu permutacije P te dobijemo matricu problema $\mathbf{B} = \mathbf{P}^T\mathbf{C}\mathbf{P}$. Time smo u ovisnosti o parametrima n i ϵ dobili način kako generirati testni primjer koji ima veličinu problema n, a greška koji smo učinili na njemu nije veća od ϵ.

Model 2

Uz ovaj već opisani način generiranja test primjera, osmislili smo još jedan model za njihovo generiranje kako bi dobili na vjerodostojnosti. Ovaj model u odnosu na prije spomenuti ima jedan dodatni međukorak. Nakon što generiramo matricu C, u ovisnoti o konstantama c_{diag} i c_{ostali} poništimo određene elemente matrice C. c_{diag} određuje broj poništenih elemenata u dijagonalnim blokovima, dok c_{ostali} određuje broj poništenih elemenata u preostalim blokovima. Nakon toga se matrica ponovno skalira na stohastičku te se iz nje, kao i prije, generira matrica B.

9.2 Testiranje na umjetno generiranim matricama

Odlučili smo se usporediti naše metoda sa metodama koje smo susreli u literaturi. Njih smo detaljno proučili i implementirali kako bi testovi bili vjerodostojni. Testirali smo samo na primjerima modela 2 jer smo zaključili da bi oni mogli biti vjerodostojniji od primjera iz modela 1.

Također, osmislili smo i način kako bodovati određeno rješenje. To je potrebno jer je rezultat koji dobijemo nekad skoro u potpunosti dobar, a nekad u potpunosti krivi. Zaključili smo da bi jedan vjerodostojan pokazatelj "dobrote" rješenja X trebao biti broj susjednih parova elemenata u vektoru permutacije koji ne pripadaju istom bloku - $f(X)$. Ukoliko metoda pronađe točno rješenje, različiti susjedni elementi će se pojavljivati samo na dodiru blokova, dakle $f(X)$ će biti jednak $f(X) = k - 1$, gdje je k broj blokova u točnom rješenju. Lako se pokaže da je $k - 1$ minimum funkcije f te da se svakim dijeljenjem nekog od blokova iz točnog rješenja na više dijelova u X, vrijednost funkcije f povećava za 1. Očekivano, uspješnost pojedine metode na nekom od test primjera smo onda ocjenjivali brojem $f(X) - (k - 1)$. Optimalno rješenje ima ocjenu 0.

Proveli smo 2 niza testiranja, jedan na matricama veličine 150×150, a drugi na matricama veličine 300×300. Varirali smo koeficijent ε na vrijednostima $0.2, 0.4, 0.6$ i 0.8, dok smo koeficijente c_{diag} i c_{ostali} držali fiksnima, $c_{\text{diag}} = 0.1$ i $c_{\text{ostali}} = 0.2$.

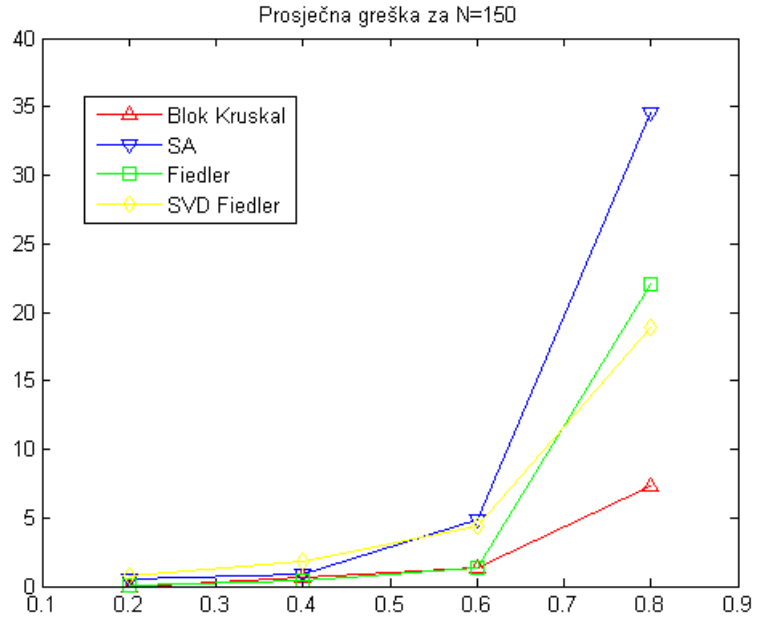
Za svaku od kombinacija veličine matrice i koeficijenta ε smo proveli testiranje metoda na 100 slučajno generiranih problema modela 2 i izračunali im prosječnu ocjenu i standardnu devijaciju ocjene.

9.3 Testiranje na matricama iz primjene

Kako ne bi testirali samo na umjetno generiranim matricama, odlučili smo se testirati svoje metode i na matricama iz primjene. Autori članka [8] su nam udijelili matrice Ph300 i Ph500 koje potječu iz pročavanja meta-stabilnih stanja u kemijskim reakcijama

Tablica 1: Rezultati za $N = 150$

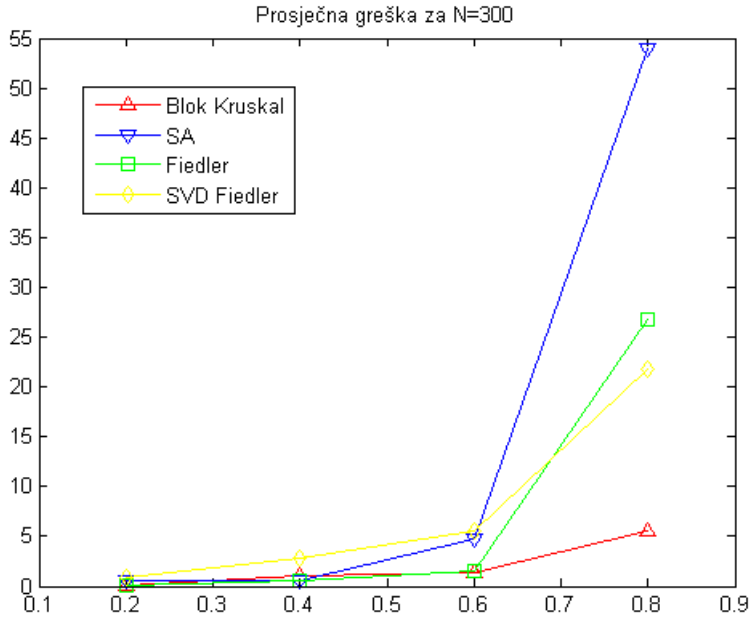
Metoda	Parametar	$\varepsilon = 0.2$	$\varepsilon = 0.4$	$\varepsilon = 0.6$	$\varepsilon = 0.8$
Blok Kruskal	E	0	0.67	1.29	7.34
	Stddev	0	0.63	0.88	12.68
SA	E	0.48	0.91	4.82	34.51
	Stddev	1.03	1.44	5.7	22.29
Fiedler	E	0.09	0.38	1.32	22
	Stddev	0.35	0.73	2.74	23.77
SVD Fiedler	E	0.72	1.82	4.43	18.91
	Stddev	2.24	2.98	4.55	15.51



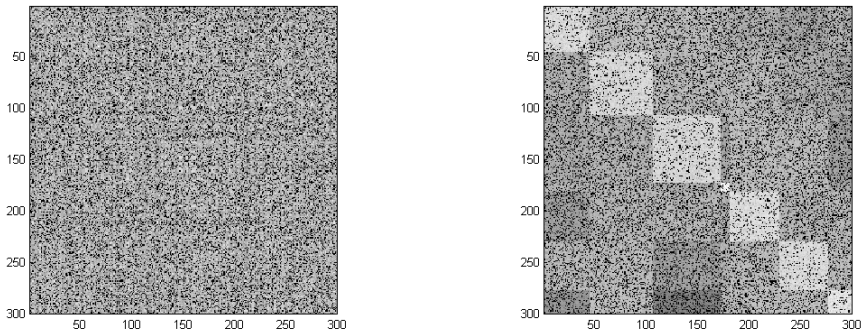
Slika 3: Usporedba metoda prema prosječnoj grešci na matricama veličine 150 $\times$ 150 i u ovisnosti o parametru ε.

Tablica 2: Rezultati za $N = 300$

Metoda	Parametar	$\varepsilon = 0.2$	$\varepsilon = 0.4$	$\varepsilon = 0.6$	$\varepsilon = 0.8$
Blok Kruskal	E	0.13	1.01	1.33	5.47
	Stddev	0.36	0.93	0.98	12.26
SA	E	0.54	0.62	4.73	54.01
	Stddev	1	1.1	8.98	35.21
Fiedler	E	0.14	0.59	1.5	26.73
	Stddev	0.35	1.08	2.04	29.48
SVD Fiedler	E	0.88	2.78	5.59	21.81
	Stddev	2.53	4.61	7.07	20.12

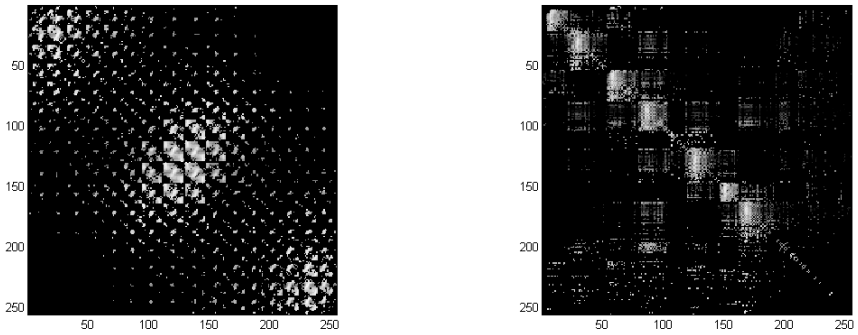


Slika 4: Usporedba metoda prema prosječnoj grešci na matricama veličine 300 $\times$ 300 i u ovisnosti o parametru ε.

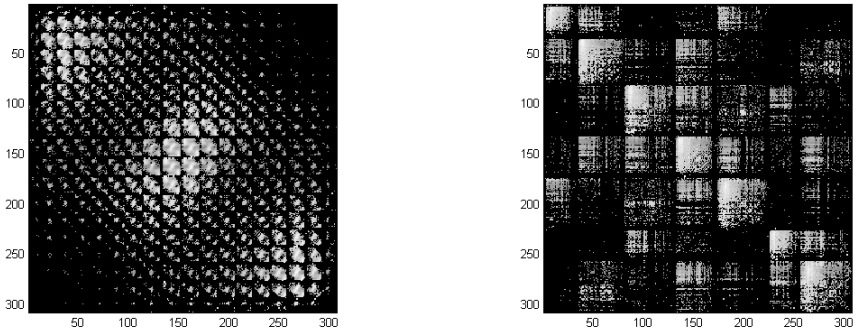


Slika 5: Prikaz inicijalne matrice problema i našeg izračunatog točnog rješenja. Parametri ovog test primjera su $N = 300$ i $\varepsilon = 0.4$.

n-pentana. Te matrice su strukturirane jako loše te su naše metode davale relativno loše rezultate na njima, tj. blok dijagonalna struktura nije baš bila uočljiva. Međutim na rezultatima metode 'Blok Kruskal' se moglo primjetiti da bi ih se možda dalo popraviti. Primjenili smo na dobivene rezultate metodu k-sredina kao korektor, tako da smo joj kao početne centre zadali točke za koje smo bili sigurni da čine neki zaseban blok. Nakon svega par koraka metoda k-sredina je iskonvergirala (što inače nije slučaj) i rezultat je bio jako dobar. Golim okom se mogla uočiti blok dijagonalna struktura. Također, uporabom naše aplikacije smo izračunali i da dijagolni blokovi našeg rješenja imaju veće 1 i π norme od onih koje su prikazane kao rezultati u članku [8]. To također potvrđuje usješnost naše metode.



Slika 6: Prikaz matrice $Ph300$, veličine 255×255, i našeg izračunatog rješenja.



Slika 7: Prikaz matrice $Ph500$, veličine 307×307, i našeg izračunatog rješenja.

10 Zaključak

Proučili smo i implementirali postojeće metode koje rješavaju naš problem te neke njemu slične probleme. U literaturi smo pronašli samo metode koje koriste "top-down" pristup. To nas je potaklo da sami probamo pronaći metodu za rješavanje problema koja koristi "bottom-up" pristup.

Osmislili smo sami dva nova načina pristupa problemu. To su egzaktni pristup inspiriran Kruskalovim algoritmom i najmanjim razapinjajućim stablom te metaheuristički pristup pomoću metode simuliranog kaljenja. Proučili smo teoriju koja stoji iza njih te ih uspješno

implementirali. Obje metode su prošle kroz više faza razvoja. Testirane su sa različitim parametrima te njihovim raznim modifikacijama. U egzaktnom slučaju (matrica perturbacije $E = 0$) naše metode egzaktno rekonstruiraju blok-dijagonalnu strukturu te to opravdava njihovo korištenje. Također, na umjetno generiranim primjerima koji odgovaraju matematičkom modelu one daju jako dobre rezultate. Usporedili smo rezultate naših metoda sa rezultatima metoda koje smo susreli u literaturi, baziranim na Fiedlerovom vektoru koji koristi svojstveni vektor odnosno singularni vektor. Metoda zasnovana na simuliranom kaljenja ima rezultate u rangu onih koje koriste Fiedlerov vektor. Metoda zasnovana na Kruskalovom algoritmu ima znatno bolje rezultate od onih koje koriste Fiedlerov vektor, te time možemo opravdati njeno korištenje u primjenama za rješavanje problema.

Na primjerima iz primjene (kemijske reakcije) naše metode vrše određeni pomak prema rješenju, ali ne daju točno rješenje. Razlog tome je što za te primjere iz primjene niti ne postoji egzaktno rješenje. Kako bi to djelomično popravili iskoristili smo metodu k-sredina (eng. *k-means*) kao korektor. Kombinacijom pristupa inspiriranim Kruskalovim algoritmom kao prediktorom i heurističkom metodom k-means uspjeli smo ostvariti izvrsne rezultate. Postignute norme blokova na dijagonali su veće nego u literaturi [8].

Kako bi uspjšno testirali različite pristupe rješavanju ovog, ali i sličnih problema razvili smo Matlab GUI aplikaciju koja omogućava usporedbu različitih metoda, daje dodatne informacije o kvaliteti rješenja te omogučava korisniku određene intervencije na pronađenom rješenju. Naš software smo stavili na raspolaganje i nekima od vodećih istraživača na ovom području.

Zahvale

Zahvaljujemo se našem mentoru, prof. dr. sc. Zlatku Drmaču, na vrlo zanimljivoj temi i problemu u koji nas je uveo, velikom razumijevanju, te mnogim korisnim i stručnim savjetima koji su nam uvelike pomogli.

Također, željeli bi se zahvaliti i autorima članka [8] profesorima Davidu Fritzscheu (Bergische Universitat Wuppertal), Volkeru Mehrmannu (Technische Universitat Berlin), Danielu B. Szyldu (Temple University) te Eleni Virnik (Technische Universitat Berlin) na tome što su nam velikodušno ustupili primjerke nekih matrica za testiranje.

Literatura

[1] Svitlana Ruzhytska, Martin Nilsson Jacobi, Christian H. Jensen, and Dmitry Nerukh. Identification of metastable states in peptide's dynamics.

[2] P. Deuflhard, W. Huisinga, A. Fischer, and Ch. Schütte. Identification of almost invariant aggregates in reversible nearly uncoupled Markov chains. *Linear algebra and its applications*, 315:39–59, 2000.

[3] Jitendra Malik Jianbo Shi. Normalized cuts and image segmentation.

[4] Daniel W. Stroock. *An Introduction to Markov Processes*. Springer.

[5] James W. Demmel. *Applied Numerical Linear Algebra*. SIAM.

[6] Stella X. Yu and Jianbo Shi. Multiclass spectral clustering. 2003.

[7] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[8] David Fritzche, Volker Mehrmann, Daniel B. Szyld, and Elena Virnik. An SVD approach to identifying metastable states of Markov chains. *Electronic Transactions on Numerical Analysis*, 29:46–69, 2008.

[9] Ryan. M. Tifenbach. Issues concernig "An SVD approach to identifying metastable states of Markov chains". *Electronic Transactions on Numerical Analysis*, 38:17–33, 2011.

[10] Robert Sedgewick. *Algorithms in C++, part 5, third edition*. Addison-Wesley.

[11] Jon Kleinberg and Éva Tardos. *Algorithm Design*. Addison-Wesley, 2005.

[12] David Kravitz. Two comments on minimum spanning trees.

[13] Tarique Mesbaul Islam and M. Kaykobad. Worst-case analysis of generalized heapsort algorithm revisited. *International Journal of Computer Mathematics*, 83:59–67.

[14] Peter J. M. Laarhoven and Emile H. L. Aarts. *Simulated annealing: theory and applications*. SPRINGER, 1987.

[15] Bruce Hajek. Cooling schedule for optimal annealing. *Mathematics of Operations Research*.

[16] Martin Nilsson Jacobi. A robust spectral method for finding lumpings and meta stable states of non-reversible Markov chains. *arXiv*, 2010.

[17] Mathworks. `http://www.mathworks.com/matlabcentral/`.

[18] Patrick Marchand and O. Thomas Holland. *Graphics and GUIs with Matlab, third edition*. Chapman & Hall/CRC.

Sažetak

Petar Sirković i Saša Stanko

Otkrivanje blok dijagonalne strukture stohastičkih matrica i identifikacija metastabilnih stanja Markovljevih lanaca

Ključne riječi: Kruskal, simulirano kaljenje, metastabilna stanja, particioniranje grafa, Markovljevi lanci

U ovom smo radu proučavali problem otkrivanja blok dijagonalne strukture perturbirane blok-dijagonalne stohastičke matrice, tj. pronalazak metastabilnih stanja Markovljevih lanaca. Opisali smo gdje se sve ovaj problem u tom ili sličnom obliku može pronaći u primjeni te ukratko objasnili postojeće metode za rješavanje ovog problema koje smo pronašli u literaturi.

U nastavku donosimo opis dva nova pristupa rješavanju problema koje smo osmislili. Jedan se zasniva na interpretaciji matrice pomoću pripadnog grafa te svojstvima najmanjeg razapinjujućeg stabla i Markovljevih lanaca. Koristili smo Kruskalov algoritam i njegovo svojstvo da $k-1$ koraka prije kraja on pronalazi $n-k$ međusobno najudaljenijih komponenti. Povezanost između čvorova ili skupa čvorova grafa smo mjerili vjerojatnošću prelaska iz jednog skupa stanja u drugi u pripadnom Markovljevom lancu. Drugi pristup rješavanju koji smo opisali je metaheuristički preko metode simuliranog kaljenja. Donosimo matematički model problema u terminima metode simuliranog kaljenja i opravdavamo zašto takav pristup vodi rješenju.

Na kraju donosimo rezultate testiranja svojih metoda i usporedbu tih rezultata sa rezultatima drugih metoda koje smo susreli u literaturi (particioniranje preko Fiedlerovih vektora traženjem svojstvenog odnosno singularnog vektora). Osmislili smo način generiranja testnih primjera te valoriziranja rezultata koji smo matematički opravdali. Rezultati metode zasnovane na simuliranom kaljenju su otprilike u rangu rezultata koje su prikazale metode

bazirane na Fiedlerovom vektoru, dok su rezultati metode zasnovane na Kruskalovom algoritmu znatno bolje. Također, donosimo i primjer rada metode na dvjema matricama iz primjene na kojima naša metoda prikazuje jako dobre rezultate. Naši trenutni rezultati su ohrabrujući i motiviraju nas na daljnje istraživanje na ovom području.

Dodatno, osmislili smo i implementirali Matlab GUI aplikaciju koja olakšava testiranje, usporedbu te razvijanje novih metoda. Aplikacija je rađena, imajući na umu, da bude korisna i na pročavanju drugih problema na matricama.

Summary

Petar Sirković i Saša Stanko

Revealing block-diagonal structure of stochastic matrices and identification of metastable states of Markov chains

Keywords: Kruskal, simulated annealing, metastable states, graph partitioning, Markov chains

In this work we have studied the problem of revealing block-diagonal structure of perturbed block-diagonal stochastic matrix - finding metastable states of Markov chains. We have described where the problem can be found in this or in similar form in the applications and shortly described existing methods for solving this problem that we found in the literature.

Next, we describe two new approaches to solving the problem that we have designed. One approach is based on the graph interpretation of the matrix, well-known properties of the minimum spanning tree and Markov chains. We have used the Kruskal algorithm and its property that $k-1$ iterations before the end it finds a $n-k$ mutually most distant components. We have measured the connection between the vertices or groups of vertices using the transition probability between the states in the corresponding Markov chain. The second approach that we have designed, is metaheuristical by using simulated annealing method. We bring mathematical model of this problem in the terms of the simulated annealing method and justify why it leads to the solution.

In the end, we bring the results of testing and comparing our methods against two methods that we have found in the literature (partitioning via Fiedler vector by computing singular or eigen vector). We have designed a way to generate test examples and evaluate the results, which we have mathematically justified. Results of the method based on simulated annealing are somewhere around the results of the two methods we have compared it to,

and the results of the Kruskal based method are significantly better. Also, we bring the results of testing of our method on matrices from the applications, which are very good. Our preliminary results are encouraging and they motivate us for further research on this topic.

Additionally, we have designed and implemented Matlab GUI application that makes testing, comparing and designing new solution methods much more easier. We hope that this software will be used as a useful designing tool in variety of applications.

Kratki životopisi autora

Petar Sirković

Rođen je 14. rujna 1988.g. u Zagrebu, gdje je 2007.g. završio V. gimnaziju. Tijekom srednjoškolskog obrazovanja je sudjelovao na brojnim natjecanjima iz matematike, fizike, informatike i logike. Redovito je ostvarivao zapažene rezultate. Najveći uspjesi su brončana medalja sa Međunarodne matematičke olimpijade u Vijetnamu 2007.g., te osvojena 1. mjesta na međunarodnim informatičkim natjecanjima ACSL 2007.g. i TopCoder High School Competition 2007.g. Nakon završenog srednjoškolskog obrazovanja upisao je predimplomski studij Matematika na Prirodoslovno matematičkom fakultetu u Zagrebu. Završio ga je 2010.g. sa prosjekom ocjena 4.97, a tijekom studija je sudjelovao više puta na međunarodnim studentskim natjecanjima iz matematike. Najveći uspjeh je 2. nagrada sa International mathematics competition 2009.g. u Budimpešti. Nakon toga, 2010. g. je upisao diplomski studij Primjenjena matematika na Prirodoslovno matematičkom fakultetu u Zagrebu, kojeg studira i trenutno. Dosadašnji prosjek ocjena mu je 5.00.

Saša Stanko

Rođen je 4. travnja 1988.g. u Zagrebu, gdje je 2007.g. završio V. gimnaziju. Tijekom srednjoškolskog obrazovanja je sudjelovao na brojnim natjecanjima iz matematike, fizike i informatike. Redovito je ostvarivao zapažene rezultate. Najveći uspjesi su sudjelovanje na Međunarodnoj matematičkoj olimpijadi u Vijetnamu 2007.g., 1. mjesto na državnom natjecanu iz matematike 2007.g. te osvojena 1. mjesta na međunarodnom informatičkom natjecanju ACSL 2007.g. Nakon završenog srednjoškolskog obrazovanja upisao je predimplomski studij Matematika na Prirodoslovno matematičkom fakultetu u Zagrebu. Završio ga je 2010.g. sa prosjekom ocjena 5.00, a nagrađen je i kao jedan od najboljih studenata fakulteta. Tijekom studija je sudjelovao više puta na međunarodnim

studentskim natjecanjima iz matematike. Nakon toga, 2010. g. je upisao diplomski studij Računarstvo i matematika na Prirodoslovno matematičkom fakultetu u Zagrebu, kojeg studira i trenutno. Dosadašnji prosjek ocjena mu je 5.00.

A Aplikacija

Motivacija

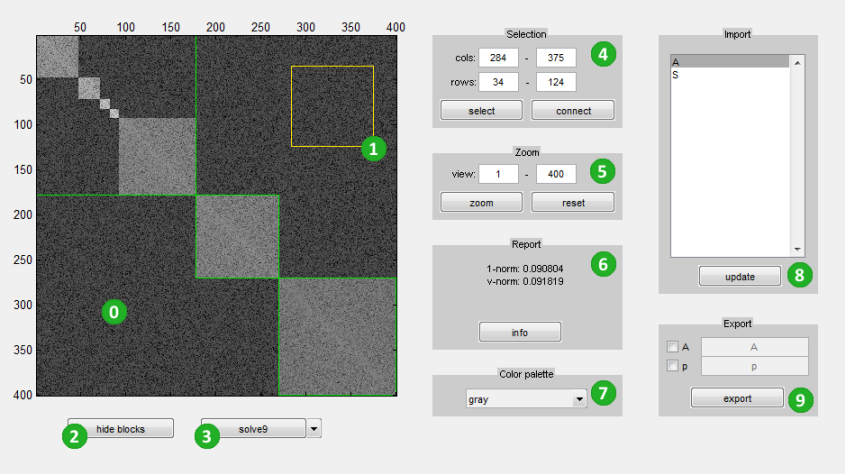
Motivaciju za naš rad smo dobili čitajući [8] gdje je predstavljeno rješenje temeljeno na SVD dekompoziciji. Zapazili smo da je u nekim slučajevima golim okom moguće uočiti poboljšanja izračunatog rješenja. To nas je motiviralo na razvoj aplikacije koja će kroz grafičko sučelje omogućiti pozivanje metoda, pregled rezultata te spomenutu intervenciju.

Mogućnosti pokretanja različitih metoda i pregleda rezultata može se koristiti za isprobavanje novih ideja te usporedbu potojećih. Samim time, aplikacija može koristiti i drugim znanstvenicima u njihovom radu. Interveniranje korisnika se provodilo označavanjem pojedinog dijela matrice čime je omogućen pristup dodatnim informacijama. Ugrađene su još neke mogućnosti, a sve u svrhu boljeg razumijevanja problema, ali i lakšeg razvoja metoda.

Prilikom razvijanja naše metode te proučavanja ovog problema, sama aplikacija nam je bila od velike koristi. Nadamo se da će biti i ostalim korisnicima koji se bave sličnom problematikom.

Pokretanje aplikacije

Sve datoteke koje dolaze uz aplikaciju potrebno je pohraniti u jedan direktorij. Aplikacija se pokreće iz Matlaba, na primjer, upisivanjem njenog imena u komandnu liniju. Slika 8 prikazuje aplikaciju u trenutku rada. Manja odstupanja u izgledu su moguća. Na slici smo pojedine komponente aplikacije naznačili različitim brojevima kako bismo olakšali opisivanje.



Slika 8: Aplikacija u radu

Početak rada

Kako bi učitali matricu problema u aplikaciju ona mora biti pohranjena u Matlab radnom okruženju (*Workspace*). Sve varijable pohranjene u radnom okruženju su prikazane na listi (oznaka 8). Ako je tijekom rada aplikacije došlo do promjena u radnom okruženju, to se neće primijetiti na ovoj listi sve dok se ne pritisne *update* gumb. Odabirom imena matrice, ona će biti prikazati u oknu (oznaka 0). Pretpostavlja se da je matrica kvadratna te dimenizije veće od 1.

Matrica se zamijenjuje rasterskom slikom veličine $n \times n$, gdje je n dimenzija matrice problema. Potom se slici promijeni veličina tako da ispuni okno. Za određivanje intenziteta pojedinog piksela korištena je logaritamska skala. Paletu boja koja će se koristiti za prikaz matrice moguće je odabrati u padajućem izborniku (oznaka 7).

Korištenje postojećih metoda

Metoda za rješavanje problema se odabire kombinacijom padajućeg izbornika i gumba (oznaka 3). Pritiskom na strelicu prikazat će se izbornik u kojemu su ponuđene sve

dostupne metode. Nakon odabira metode, pritiskom na gumb započinje njeno provođenje. Gumb pritom mijenja ime gumba u *working*. Nakon završetka rada dobiveni rezultat je prikazan u oknu, a gumbu je vraćeno prijašnje ime.

Sljedeća funkcionalnost je još u razvoju. Aplikacija trenutno ne nude informacije o blokovima, ali nudi njihovo pronalaženje putem gumba *show blocks* (oznaka 2). Metoda koji to obavlja koristi pohlepni pristup te često ne daje željene rezultate.

Dodavanje novih metoda

Korisnik može napraviti svoju metodu i dodati je u padajući izbornik. Metodu je potrebno napisati u Matlab prepoznatljivom formatu (*m file*, *mex file* ili slično) te joj dodijeliti ime koje ne sadrži razmake. Metoda mora omogućiti primanje točno jednog parametra – matricu problema, te vraćanje točno jednog parametra – redak permutacije. U direktorij gdje je pohranjena aplikacija, potrebno je stvoriti datoteku `algorithms.txt` te u njoj, red po red, navesti imena vlastitih metoda. Također, same metode je potrebno smjestiti u taj direktorij. Pri sljedećem pokretanju aplikacije, metode čija su imena zapisana u toj datoteci će biti smještene u padajući izbornik. Napomenimo da metode koje dolaze s aplikacijom koriste isti pristup. Stoga je i njihova imena potrebno na isti način navesti u datoteci `algorithms.txt`, ukoliko ih korisnik želi koristiti.

Pravokutnik izbora

Kliknite lijevom tipkom miša bilo gdje na okno u kojem se prikazuje matrica i zadržite. Povucite miš u nekom smjeru, zadržite ga te ispustite tipku. Dobili ste ono što nazivamo pravokutnikom izbora, žuti pravokutnik (oznaka 1) određen točkama u kojima ste pritisnuli, odnosno otpustili tipku miša. U *Selection* području (oznaka 4) pojavit će se brojevi prvog i zadnjeg retka i stupca koje pravokutnik zauzima. Te vrijednosti je moguće mijenjati, a pritiskom na *select* gumb promjene će biti ostvarene.

U *Report* području (oznaka 6) korisnik može saznati neke informacije o označenom dijelu matrice. Trenutno su to $\mathbb{1}$-norma i v-norma koje se iznova računaju prilikom svake promjene pravokutnika izbora. U istom području možemo primijetiti kombinaciju gumba i padajućeg izbornika. Njihovim korištenjem moguće je dobiti grafički prikaz različitih informacija. Četiri mogućnosti su prikaz prva dva lijeva ili desna svojstvena ili singularna vektora. Pritiskom na gumb ti vektori će biti prikazani prikazati. Peta mogućnost je prikazivanje informacija dobivenih korištenjem aproksimacije matrice matricom ranga tri.

Time smo naveli jednu ulogu, a sada navodimo primarnu ulogu – spajanje blokova. Pritiskom na *connect* gumb, dva skupa stanja određena pravokutnikom spojit će se u jedan. Retci koje pravokutnik obuhvaća predstavljaju jedan skup, a stupci drugi skup stanja.

Također, moguće je pokrenuti izvršavanje odabrane metode na odabranoj podmatrici. Podmatrica mora biti glavna podmatrica trenutne matrice. Ukoliko to nije slučaj, prilikom pokretanja metode, odabrana podmatrica će biti zamijenjena onom koja zadovoljava ovaj uvjet. Metoda se pokreće pritiskom na *solve* gumb.

Dodali smo i mogućnost uvećavanja slike (zoom). Korisnik može uvećati određenu podmatricu upisivanjem njenih koordinata u *Zoom* područje (oznaka 5). Pritiskom na *zoom* gumb, podmatrica će biti uvećana. Za povratak na izvorni pogled potrebno je pritisnuti *reset* gumb.

Za uklanjanje pravokutnika izbora, potrebno je samo jednom kliknuti na okno.

Zatvaranje aplikacije

Spremanje rezultata se vrši u *Export* području (oznaka 9). Najprije se odabire koji će se podatci pohraniti. Opcija „A" je za pohranu matrice dobivenu kao rezultat rada. Opcija „p" je za pohranu permutacijskog redka kojim se od polazne matrice dolazi do trenutno prikazane. Nakon toga je potrebno upisati imena koja će biti dodijeljena odgovarajućim varijable te pritisnuti *export* gumb.

Napomene

U ovom paketu se nalaze četiri naše metode. Prve dvije su jednostavni pokušaji rješavanja problema iz kojih je nastao algoritam 1. Algoritam 2 je ponuđen kao treća metoda. Posljednja metoda je Blok-Kruskal, no ne implementiran na način spomenut u radu. Još nekoliko metoda je u razvoju te bi trebale biti dodane uskoro.

Blok-Kruskal baziran na hrpi i metoda bazirana na simuliranom kaljenju su pisane u programskom jeziku C. One neće biti prepisane u Matlab kôd jer bi time izgubile na efikasnosti. Umjesto toga, planiramo iskoristiti sučelje koje Matlab pruža prema kôdovima pisanima u jeziku C.

Ova aplikacija je razvijena pod Matlab verzija 7.1. Kako bi osigurali portabilnost na druge verzije Matlaba, sav kod aplikacije je ručno napisan bez korištenja Matlab GUIDE-a. Ukoliko pronađete greške ili iskusite probleme u radu, voljeli bismo znati. Sugestije, dojmovi i kritike su svakako dobrodošli.

Od velike pomoći nam je bila pomoć ugrađena u Matlab, službene web stranice [17] te knjiga [18].