УНИВЕРЗИТЕТ У БЕОГРАДУ
МАТЕМАТИЧКИ ФАКУЛТЕТ

Филип Видојевић

МЕТОДЕ ЗА ФАЗИ КЛАСТЕРОВАЊЕ НА КОМПЛЕКСНИМ МРЕЖАМА

мастер рад

Београд, 2021.

Ментор:

проф. др Мирослав Марић, редовни професор
Универзитет у Београду, Математички факултет

Чланови комисије:

др Нина Радојичић Матић, доцент
Универзитет у Београду, Математички факултет

др Стефан Мишковић, доцент
Универзитет у Београду, Математички факултет

Датум одбране: септембар 2021.

Наслов мастер рада: Методе за фази кластеровање на комплексним мрежама

Резиме: Кластеровање представља процес груписања података тако да степен сличности између елемената буде максималан ако припадају истој групи и минималан ако припадају различитим групама. Уколико су подаци над којима се врши кластеровање представљени у форми мреже која описује структуру неког комплексног система, ради се о кластеровању на комплексним мрежама. До сада је у литератури предложено неколико мера за оцену квалитета дисјунктног кластеровања, попут модуларности, E-фунцкије и других. Коришћењем ових мера, проблем кластеровања на мрежи се формулише као проблем комбинаторне оптимизације, а за решавање се могу користити различите методе математичке оптимизације.

У овом раду, извршено је прилагођавање E-функције за преклапајуће (фази) кластеровање на комплексним мрежама у којима кластери нису дисјунктни, већ се преклапају, односно сваки чвор има вероватноћу припадања сваком од кластера. Вредности модификоване функције упоређене су са вредностима фази модуларности, која је показала велики успех у откривању преклапајућих чворова, на Захаријевом карате клубу и великој PGP мрежи као скуповима података.

Поред тога, приказан је општи преглед, као и два савремена алгоритма за фази кластеровање на комплексним мрежама. Као алгоритам заснован на максимизацији модуларности, алгоритам брзе максимизације фази модуларности (енгл. *Fast Fuzzy Modularity Maximization* - `FFMM`) постиже изванредне резултате у терминима временске и просторне сложености извршавања. Алгоритам `FFMM` примењен у више циклуса, при чему се у сваком циклусу примењује на подмреже добијене у претходном кораку, омогућава ефикасно откривање преклапајућих чворова у мрежама великих димензија. Са друге стране, алгоритам пропагације степена припадања (енгл. *Membership Degree Propagation Algorithm* - `MDPA`) базира се на пропагацији ознака. За разлику од већине пропагационих метода, код њега се врши пропагација вектора степена припадања потенцијалним кластерима уместо ознака.

Кључне речи: фази кластеровање, комплексне мреже, модуларност, E-функција

Садржај

Глава 1

Увод

Многи проблеми у реалном свету, као што су друштвене и економске организације, географска клима, људски мозак, делови инфраструкуре, али и читав универзум, природно се моделују системом. Због своје широке примене, појам „систем" је тешко прецизно дефинисати. Уопштено говорећи, систем представља скуп ентитета који заједно са својим међусобним интеракцијама, зависностима и везама формирају јединствену целину. Системи могу имати особине и понашања која се потпуно разликују од особина и понашања објеката који им припадају. На пример, у систему фиксне телефоније, телефони представљају ентитете који појединачно немају никакву улогу. Међутим, међусобним повезивањем више телефона добија се систем који нуди могућност комуникације.

Важна особина система је њихова комплексност. Иако не постоји опште прихваћена дефиниција комплексних система, постоје многи примери комплексности. Систем се може сматрати комплексним уколико, на пример, има хаотично понашање (изузетна осетљивост на почетне услове), настајућа својства (својства која нису видљива из њихових компоненти изоловано, али која су резултат односа и зависности које формирају када се ставе заједно у систем) или га није могуће рачунарски моделирати (зависи од броја параметара који експоненцијално расте са повећањем мреже). Стога, може се рећи да су системи чије понашање не може једноставно да се изведе из њихових особина, односно особина појединачних ентитета, комплексни системи. Компоненте комплексног система се могу представити мрежом, која представља колекцију објеката и релација између њих, при чему се објекти најчешће представљају чворовима графа, а релације ивицама које их повезују. Посматрање

мреже на тај начин омогућава примену теорије графова и науке о мрежама.

Комплексне мреже често имају својство нелинеарности, које подразумева да различито реагују на исте улазе, у зависности од тренутног стања или контекста. Уколико дође до промене на улазу, код таквих мрежа излаз се најчешће драстично промени или га чак не буде уопште. Иако се комплексне мреже изучавају дуги низ година, у последње две деценије је дошло до експлозије примера комплексних мрежа, како у реалном свету, тако и вештачки генерисаних, а самим тим и до повећане потребе за ефикасним алгоритмима који раде са њима.

Често се дешава да у мрежи постоје групе чворова који су боље повезани међу собом него са остатком мреже. Овакве групе чворова се називају кластери (заједнице, модули). Проблем проналажења оваквих група у комплексним мрежама назива се проблем кластеровања на комплексним мрежама. Чворови у комплексним мрежама који припадају истим кластерима углавном имају исте улоге или особине, и обрнуто. На пример, у друштвеним мрежама, особе у истој заједници могу имати исту каријеру, исти хоби, бити из исте државе, док у мрежама протеина, протеини у истом кластеру могу имати сличне улоге. Међутим, једна особа може имати више хобија или два држављанства. Стога, намеће се потреба за преклапајућим чворовима у комплексним мрежама.

Облик кластеровања у којем сваки чвор мреже може истовремено припадати већем броју кластера назива се фази (меко, преклапајуће) кластеровање, док се чвор који је садржан у два или више кластера назива фази (преклапајућим) чвором. Фази кластеровање посебно долази до изражаја у ситуацијама када постоје чворови који представљају спону, односно прелаз између два или више кластера мреже. Из тог разлога, идентификација преклапајућих чворова је једна од важних тема у анализи комплексних мрежа. На пример, због избијања епидемије у 2020. години, многи истраживачи су испитивали важност преклапајућих чворова при ширењу епидемије, а затим су сходно томе развили стратегију имунизације [14]. Када избије епидемија, често је немогуће вакцинисати сво становништво, због ограничене количине ресурса. Због тога, главни циљ је идентификовати утицајне преносника заразе, како би се смањили трошкови вакцинације и ширење заразе.

Поред самог кластеровања, важно је одредити колико је извршено кластеровање добро, односно измерити квалитет кластеровања. Функција која мери

квалитет кластеровања назива се оценом (мером) квалитета кластеровања и биће јој посвећена посебна пажња у овом раду.

Пошто се комплексне мреже природно представљају графовима, у наредној секцији је дат кратак увод у теорију графова.

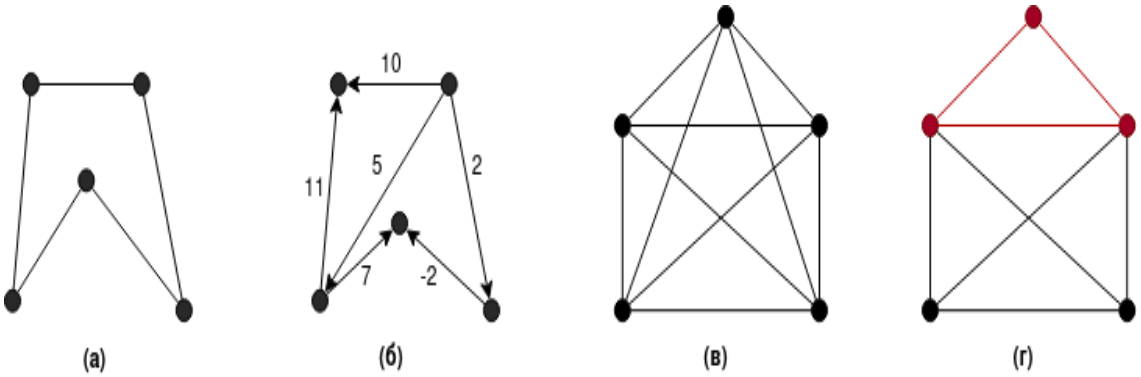
1.1 Графови

1.1.1 Основни појмови

Граф G је уређени пар скупова (V, E), где је V *скуп чворова*, а E *скуп ивица* (*грана*), односно скуп неуређених парова елемената из V. Чворови који одређују једну ивицу се називају *крајевима ивице*, док је одговарајућа ивица *суседна* тим чворовима. Уколико је свака ивица уређени пар чворова, за одговарајући граф кажемо да је *усмерен* (односно *дограф*). У том случају, уређени пар (v, w) је ивица *усмерена* од v ка w, односно ивица са почетком у v и крајем у w. Визуелно, граф је скуп тачака повезаних линијама. Примери графова дати су на слици 1.1.

Често се јавља потреба за разликовањем ивица графа по важности. Стога, може се доделити тежина свакој од ивица. У том случају, граф се назива *тежинским*. У свом основном облику, графови не укључују петље, односно ивице које спајају чвор са самим собом, а ни вишеструке ивице између два чвора. Графови са петљама и вишеструким ивицима се називају *мултиграфовима*. Поред тога, могу се дефинисати ивице које спајају више од два чвора истовремено. Граф који одговара том случају јесте *хиперграф*.

Број чворова у графу представља његов *ред*, док број ивица представља његову *величину*. Ред и величина графа се најчешће означавају словима n и m. Максимална величина графа једнака је броју свих уређених парова чворова из V, односно $n(n-1)/2$. Ако је $|V| = n$ и $|E| = m = n(n-1)/2$, онда се граф назива n-*кликом* (односно *комплетним графом*) и означава са K_n. Удео броја ивица графа G у укупном броју могућих ивица назива се *густином* графа G и означава са d_G. Два чвора су *суседна* уколико постоји ивица која их спаја, при чему се скуп суседа чвора v означава са $\mathbb{N}_v$. *Степен* k_v чвора v је број његових суседа. Ако је у питању усмерен граф, постоји два типа степена чвора: *улазни степен*, односно број ивица са крајем у чвору v, и *излазни степен*, односно број ивица са почетком у чвору v. Аналогно



Слика 1.1: Примери графова са 5 чворова: **(а)** неусмерен граф без тежина, **(б)** усмерен граф са тежинама, **(в)** комплетан граф са 5 чворова, **(г)** подграф индукован чворовима обојеним црвено

степену чвора, у тежинским графовима се дефинише *снага чвора*, односно сума тежина свих ивица суседних том чвору.

Граф $G' = (V', E')$ је *подграф* графа $G = (V, E)$ уколико важи $V' \subset V$ и $E' \subset E$. Ако G' садржи све ивице графа G које повезују чворове из V' онда је подграф G' *индукован* скупом V'. Партиција скупа чворова V на два подскупа S и $V \setminus S$ се назива *резом графа*, при чему је његова *величина* број ивица графа G које спајају чворове скупа S са чворовима скупа $V \setminus S$.

У овом раду, комплексна мрежа је представљена неусмереним графом без тежина, при чему се сви алгоритми и оцене квалитета могу уопштити и за остале случајеве. Иако се термини „граф" и „мрежа" често односе на различите концепте, у контексту кластеровања ће бити коришћени као синоними.

1.1.2 Повезаност

Пут у графу G је граф $\mathcal{P} = (V(\mathcal{P}), E(\mathcal{P}))$, где је $V(\mathcal{P}) = \{x_0, x_1, ..., x_l\}$ и $E(\mathcal{P}) = \{x_0x_1, x_1x_2, ..., x_{l-1}x_l\}$. Чворови x_0 и x_l представљају *почетак* и *крај* пута, а l његову *дужину*. За два пута кажемо да су *независна* уколико немају заједничких елемената, осим почетка и краја. Са друге стране, скуп чворова (ивица) је *независан* уколико не постоје елементи у њему који су суседни. Затворен пут дужине l чији су чворови и ивице међусобно различити назива се *циклусом дужине* l и означава C_l. Најмањи нетривијални циклус је циклус дужине 3, односно троугао C_3.

Захваљујући постојању путева у графу, може се дефинисати повезаност, као и удаљеност између чворова. Граф је *повезан* уколико за свака два

чвора, постоји бар један пут чији су они крајеви. Уопштено говорећи, може постојати више различитих путева између два чвора. *Најкраћи пут* између два произвољна чвора је пут са најмањом дужином, при чему је та дужина *удаљеност (растојање)* између тих чворова. Највећа удаљеност између два произвољна чвора у повезаном графу представља његов *дијаметар*. Уколико не постоји пут између два чвора, граф је подељен на најмање два повезана подграфа. Сваки максималан повезани подграф графа назива се његовом *повезаном компонентом*.

Од посебне важности за теорију графова су повезани графови без циклуса, који се називају *стаблима*. У стаблу, постоји тачно један пут између два чвора. Када би постојала два различита пута, они би формирали циклус, што се не слаже са дефиницијом стабла. Број ивица стабла са n чворова је $n-1$. Уколико би нека ивица стабла била уклоњена, граф би постао неповезан, док у случају додавања нове ивице, у графу би постојао бар један циклус. Стога се за стабло са n чворова каже да је највећи ацикличан, а најмањи повезан граф реда n. Сваки повезан граф садржи *повезујуће стабло*, односно стабло које садржи све чворове графа. Код тежинских графова, може се дефинисати *минимално повезујуће* стабло, тј. повезујуће стабло са најмањим збиром тежина ивица. Минимално повезујуће стабло се често користи код графовских оптимизационих проблема, укључујући и кластеровање.

1.1.3 Графови и матрице

Све информације о топологији графа реда n садржане су у *матрици суседства* $\mathbf{A}_{n\times n}$, чији је елемент a_{ij} једнак 1 ако постоји ивица која спаја чворове i и j, а 0 иначе. Због непостојања петљи, дијагонални елементи матрице $\mathbf{A}$ су једнаки 0. За неусмерен граф, $\mathbf{A}$ је симетрична матрица. Сума елемената i-тог реда или колоне у том случају даје степен чвора i. Ако је у питању тежински граф, дефинише се матрица тежина $\mathbf{W}$, која представља уопштење матрице суседства $\mathbf{A}$. Елемент w_{ij} представња тежину ивице између чворова i и j.

Уколико са $\mathbf{k} = (k_1, k_2, ..., k_n)^T$ означимо вектор степена чворова графа G и $\|\mathbf{A}\| = \sum_{i,j} a_{ij}$, формулом

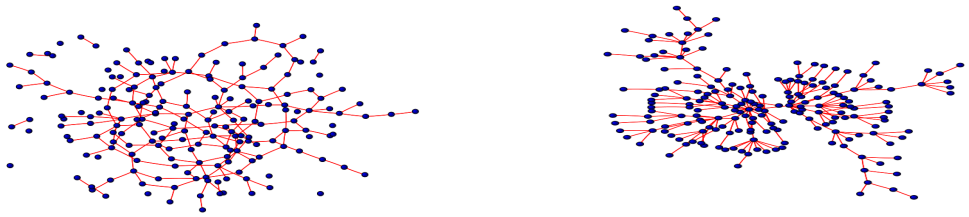
$$\mathbf{B} = \mathbf{A} - \frac{\mathbf{k}\mathbf{k}^T}{\|\mathbf{A}\|}, \tag{1.1}$$

дефинисана је *матрица модуларности* графа G [18]. Матрица модуларности користи се за рачунање квалитета кластеровања и о њој ће више речи бити у секцији 2.2.1.

С обзиром да је матрица $\mathbf{A}$ квадратна, могу се одредити њене сопствене вредности. *Спектар* графа G јесте скуп сопствених вредности његове матрице суседства $\mathbf{A}$. Спектралне особине матрица играју важну улогу у теорији графова, а самим тим и у кластеровању над графовима. Једна од важнијих матрица за откривање кластера је *Лапласијан* $\mathbf{L} = \mathbf{D} - \mathbf{A}$ [28], при чему је $\mathbf{D}$ дијагонална матрица чији је елемент D_{ii} једнак степену чвора i. Сопствени вектори Лапласијана имају битну улогу у спектралном кластеровању [13].

1.1.4 Модели графова

Да би се описали реални комплексни системи, конструисани су модели графова, односно графови са вештачки генерисаном структуром. Таква структура обично не показује тендецију ка формирању кластера, па може служити за тестирање нових метода кластеровања. У контексту детекције кластера, такви графови се обично називају *нултим моделима* [6].



(а) Случајни граф, $n = 100, p = 0.02$

(б) Граф заснован на динамичком додавању, $n = 100, \langle k \rangle = 2$

Слика 1.2: Примери нултих модела

Један од најстаријих модела графа јесте *случајни граф* (слика 1.2а), чији су параметри број чворова n и број p, који представља вероватноћу постојања ивице између два произвољна чвора графа G. Очекивани број ивица графа износи $pn(n-1)/2$, при чему је расподела степена чворова биномна. Стога, готово сви чворови графа имају исти степен, близу просечне вредности која износи $\langle k \rangle = p(n-1)$. Занимљиво својство овог модела јесте постојање разноврсних компоненти повезаности у зависности од броја $\langle k \rangle$, када $n \to \infty$. За $\langle k \rangle < 1$, граф је подељен на више компоненти повезаности, при чему је

њихова величина знатно мања од величине читавог система. За $\langle k \rangle > 1$, једна компонента постаје огромна у односу на остале, али коначна.

Са друге стране, неки нулти модели се базирају на динамичком додавању чворова у почетни граф (слика 1.2б). Вероватноћа да нови чвор буде суседан већ постојећем, пропорционална је степену постојећег чвора. На тај начин, чворови са већим степеном имају већу веровантоћу да буду сусед новог чвора. Уколико се то догоди, њихов степен се повећава, а самим тим и веровантоћа да буду изабрани у будућности.

Глава 2

Фази кластеровање на комплексним мрежама и оцене квалитета

У овом поглављу биће описан проблем преклапајућег кластеровања на комплексним мрежама, као и главни изазови при имплементацији алгоритама за откривање заједница у великим мрежама. У првој секцији су дефинисани општи појмови за партиционисање графа као репрезентације комплексне мреже, док су у другом делу представљене функције за оцену квалитета извршеног кластеровања. Од посебног значаја је модификација E-функције за случај фази кластеровања, која је показала велики потенцијал у одређивању квалитетних партиција мреже.

2.1 Кластери и партиције

Иако на први поглед интуитиван, проблем кластеровања над графом није јасно дефинисан. Најважнији елементи кластеровања, односно концепти кластера и партиције, захтевају одређену дозу произвољности. Постоји пуно примера реалних мрежа, при чему свака садржи одређене карактеристике које је разликују од осталих. Због тога се у литератури може наћи много метода које се базирају на тим карактеристикама, па је готово немогуће дати универзалну дефиницију кластеровања.

Нека је $G = (V, E)$ граф са n чворова и m грана, и нека је c цео број, $1 < c < n$. *Партиција* $\mathcal{P}$ графа G је колекција од c подскупова C_i скупа V

таква да важи:

- $C_i \neq \emptyset,\ \forall i \in \{1, ..., c\}$
- $C_i \cap C_j = \emptyset,\ \forall i, j \in \{1, ..., c\}, i \neq j$
- $\bigcup\limits_{i \in \{1,...,c\}} C_i = V$

Дакле, *кластеровање* представља одређивање партиције $\mathcal{P}$, при чему подскупови C_i добијени у том процесу индукују подграфове G_{C_i} које називамо *кластерима*. Број могућих партиција графа расте брже него експоненцијално са повећањем броја чворова у графу[1], па је енумерација свих партиција графа готово немогућа, осим у случају графова са малим бројем чворова. Партиције могу бити хијерархијски организоване, уколико граф има структуру са више нивоа. У том случају, кластери се састоје од мањих кластера, који такође могу садржати мање кластере, итд. На пример, друштвена мрежа која се састоји од деце из истог града се може партиционисати на кластере који представљају децу из истих школа, при чему се они додатно састоје од деце из различитих одељења.

Према наведеној дефиницији партиције, један чвор припада тачно једном кластеру. Међутим, у реалним мрежама, чворови често припадају различитим групама чворова истовремено. Додатно, један чвор може са различитим тежинама припадати различитим групама. Стога, потребно је дефинисати структуре за рад са партицијама које дозвољавају делимично припадање чвора кластеру.

Нека је c цео број, $1 < c < n$, при чему је n број чворова из скупа V. *C-партиција* скупа V је скуп од укупно cn вредности $\{u_{ki}\}$ које формирају матрицу $\mathbf{U} = [u_{ki}]$[2]. Елемент u_{ki} означава *степен* (*јачину*, *тежину*) припадања чвора v_i кластеру k.

Постоји три скупа c-партиција:

$$M_{pcn} = \left\{ \mathbf{U} \in \mathbb{R}^{c \times n}; 0 \leq u_{ki} \leq 1, \forall k, i; \sum_{k=1}^{c} u_{ki} > 0, \forall i; 0 < \sum_{i=1}^{n} u_{ki} < n, \forall k \right\},$$

$$M_{fcn} = \left\{ \mathbf{U} \in M_{pcn}; \sum_{k=1}^{c} u_{ki} = 1, \forall i \right\},$$

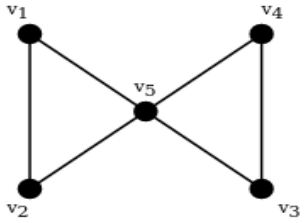
[1]Број укупних партиција графа реда n јесте n-ти Белов број [16].

[2]Надаље ће се под термином *партициона матрица* подразумевати c-партиција $\mathbf{U}$.

$$M_{hcn} = \Big\{\mathbf{U} \in M_{fcn}; u_{ki} \in 0, 1, \forall k, i\Big\}.$$

Претходне једнакости дефинишу, редом, скупове *недегенерисаних вишеструких фази* (*вишеструких*) и *строгих* c-партиција скупа V [12]. Може се уочити да важи $M_{hcn} \subset M_{fcn} \subset M_{pcn}$.

Нека је дат граф $G = (V, E)$, при чему је $V = \{v_1, v_2, v_3, v_4, v_5\}$ и $E = \{\{v_1, v_2\}, \{v_3, v_4\}, \{v_1, v_5\}, \{v_2, v_5\}, \{v_3, v_5\}, \{v_4, v_5\}\}$, приказан на следећој слици.



Слика 2.1: Граф са 5 чворова и 6 грана

Примери 2-партиција графа G су:

$$\mathbf{U}_1 = \begin{pmatrix} 0.9 & 0.8 & 0.2 & 0 & 1 \\ 0.4 & 0.5 & 0.7 & 1 & 1 \end{pmatrix}, \mathbf{U}_2 = \begin{pmatrix} 0.9 & 0.8 & 0.1 & 0.1 & 0.5 \\ 0.1 & 0.2 & 0.9 & 0.9 & 0.5 \end{pmatrix}, \mathbf{U}_3 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Важи $\mathbf{U}_1 \in M_{pcn}$, $\mathbf{U}_2 \in M_{fcn}$ и $\mathbf{U}_3 \in M_{hcn}$.

Дакле, у контексту налажења преклапајућих чворова, кластеровање над графом G подразумева налажење c-партиција $\mathbf{U} \in M_{pcn}$ скупа V. Као резултат алгоритама кластеровања добија се једна или више матрица $\mathbf{U} \in M_{pcn}$, које представљају кандидате за финалну c-партицију графа. Уколико са CP означимо скуп свих партиција кандидата за кластеровање над графом G, поставља се питање: која матрица $\mathbf{U} \in CP$ најбоље описује структуру графа G? Одговор на то питање је тема валидације кластера, односно мерења квалитета добијених партиција.

2.2 Валидација кластеровања

Осим посебних метода за налажење, од велике важности је и вредновање откривених кластера, тј. партиција $\mathbf{U}$. Стога је потребно дефинисати квантитативни критеријум који говори колико је партиција графа добра.

Нека је дат граф $G = (V, E)$ и партиција $\mathbf{U}$ графа G. Функција која додељује број, односно вредност, партицији $\mathbf{U}$ назива се *оценом* (*мером*) *ква-*

литета партиције[3] **U**. Партиције које имају веће вредности ове функције су „боље", па су оне са највећом вредношћу најбоље. Међутим, може се догодити да оцене квалитета у неким случајевима не осликавају добро стварни квалитет партиције. На једном скупу података, функција квалитета може доделити велике вредности добрим партицијама, док на другом те вредности могу бити потпуно неочекиване. Стога, функције квалитета често зависе од структуре мреже и дефиниције концепта кластера.

Са друге стране, функције које мере сличност између две партиције називају се *мерама сличности* партиција. Што је вредност мере сличности две партиције већа, то партиције више личе једна на другу. Оне су посебно корисне у примерима мрежа код којих је структура кластера већ позната, односно при тестирању нових алгоритама за кластеровање. Поређењем резултујуће партиције неког алгоритма кластеровања и већ познате партиције, добија се увид у то колико је тај алгоритам ефективан. Више о мерама сличности партиција може се видети у раду [6].

2.2.1 Модуларност

Једна од најпопуларнијих оцена квалитета за кластеровање на комплексним мрежама јесте Њуман-Гирванова модуларност. Идеја која стоји иза ње јесте да случајни граф нема структуру кластера, па се постојање кластера открива поређењем густине ивица унутар подграфа графа G и очекиване густине ивица у односу на неки нулти модел. Нулти модел представља копију графа G која садржи део његове структуре, али су ивице распоређене на случајан начин, тако да граф G нема јасно изражену структуру кластера. Примери нултих модела могу се видети у секцији 1.1.4.

Нека је V подељен на c подскупова $\{C_1, ..., C_c\}$. Модуларност се тада може записати као:

$$Q = \frac{1}{2m} \sum_{ij} (a_{ij} - P_{ij}) \delta(C_i, C_j), \tag{2.1}$$

при чему је **A** матрица суседства, m укупан број ивица графа, а P_{ij} очекивани број ивица између чворова i и j у нултом моделу. Функција δ враћа 1 уколико су чворови i и j у истом кластеру ($C_i = C_j$), а 0 иначе. Избор нултог модела значајно утиче на вредност модуларности, али је произвољан. Једна могућност је да се за вредност P_{ij} узме $p = 2m/[n(n-1)], \forall i, j$ (Бернулијев

[3]Аналогно се дефинише и у случају партиција за дисјунктно кластеровање

случајни граф). Идеја иза овог приступа јесте да нулти модел садржи исти број ивица као и оригинални граф, али да те ивице буду расподељене међу чворовима на случајан начин, са истом вероватноћом. Међутим, овај модел лоше осликава структуру реалних мрежа.

Додатно, уз број ивица у графу, може се сачувати степен чвора. У таквом нултом моделу, вероватноћа да су чворови i и j повезани износи $k_i k_j / 4m^2$; да би постојала ивица између два чвора, потребно је на случајан начин изабрати те чворове као крајње чворове ивице, са вероватноћом која зависи од степена тих чворова. С обзиром да постоји k_i ивица које су суседне чвору i, а $2m$ ивица укупно у графу, вероватноћа p_i да чвор i буде изабран износи $k_i/2m$. Тада је вероватноћа да постоји ивица између чворова i и j производ $p_i p_j$, односно $k_i k_j / 4m^2$. Пошто има укупно $2m$ ивица, очекивани број ивица између чворова i и j износи $P_{ij} = 2m p_i p_j = k_i k_j / 2m$. Дакле, модуларност се сада може представити формулом:

$$Q = \frac{1}{2m} \sum_{ij} \Big(a_{ij} - \frac{k_i k_j}{2m} \Big) \delta(C_i, C_j). \tag{2.2}$$

С обзиром да само чворови који припадају истом кластеру доприносе укупној вредности модуларности, наведена једнакост се може преформулисати тако да не садржи суму по паровима чворова, већ суму по кластерима:

$$Q = \sum_{k=1}^{c} \Big[\frac{l_k}{m} - \Big(\frac{s_k}{2m} \Big)^2 \Big], \tag{2.3}$$

где је l_k укупан број ивица унутар кластера k, а s_k сума степена свих чворова кластера k. У једнакости 2.3, први део израза унутар великих заграда представља однос између броја ивица унутар кластера и укупног броја ивица у графу, док други део представља очекивану вредност истог односа у графу са чворовима истог степена а случајно постављеним гранама. Према једначини, што је број грана унутар кластера већи од очекиваног, то је кластер боље раздвојен од остатка мреже. Стога, велике позитивне вредности модуларности подразумевају добре партиције. Вредност модуларности је увек мања од 1, а може бити и негативна. На пример, уколико су сви кластери једночлани, први део једнакости 2.3 је увек 0, па је укупна вредност мања од 0.

Наведена дефиниција модуларности подразумева строгу поделу на подскупове C_i, тј. кластер C_i садржи чвор у потпуности, или га не садржи уопште. Поставља се питање какву улогу имају строге c-партиције у израчунавању

модуларности [12]. Уколико употребимо дефиницију функције δ, једнакост 2.2 можемо записати као:

$$Q_h = \frac{1}{\|\mathbf{A}\|} \sum_{k=1}^{c} \sum_{i,j \in C_k} \Big(a_{ij} - \frac{k_i k_j}{\|\mathbf{A}\|} \Big), \tag{2.4}$$

при чему h у потпису функције Q означава да је у питању строга подела. Докажимо лему која омогућава дефинисање модуларности у случају преклапајућег кластеровања.

Lema 2.2.1. *Нека је дат граф* $\mathbf{G} = (V, E)$*, при чему је скуп* V *подељен на* c *подскупова* $C_1, ..., C_c$*. Једнакост 2.4 се може записати као:*

$$Q_h = tr(\mathbf{U}\mathbf{B}\mathbf{U}^T) / \|\mathbf{A}\|, \ \mathbf{U} \in M_{hcn}, \tag{2.5}$$

при чему је $\mathbf{B}$ *матрица модуларности графа* G*.*

Доказ. Подсетимо се да вредност $u_{ki} \in M_{hcn}$ представља степен припадања чвора v_i кластеру k, при чему додатно важи $u_{ki} \in \{0, 1\}$. Користећи вредност u_{ki} као индикатор припадања кластеру, за једнакост 2.4 важи

$$\begin{aligned} Q_h &= \frac{1}{\|\mathbf{A}\|} \sum_{k=1}^{c} \sum_{i,j=1}^{n} \Big(a_{ij} - \frac{k_i k_j}{\|\mathbf{A}\|} \Big) u_{ki} u_{kj} \\ &= \frac{1}{\|\mathbf{A}\|} \sum_{k=1}^{c} \mathbf{U}_k \mathbf{B} \mathbf{U}_k^T \\ &= tr(\mathbf{U}\mathbf{B}\mathbf{U}^T) / \|\mathbf{A}\|. \end{aligned}$$

□

Претходна лема нам експлицитно показује како матрица $\mathbf{U}$ утиче на вредност модуларности. Још важније од тога, једнакост 2.4 је добро дефинисина за све типове c-партиција, не само за строге. Стога, може се дефинисати *генерализована модуларност* c-партиције $\mathbf{U}$ графа $G = (V, E)$ као:

$$Q_g = tr(\mathbf{U}\mathbf{B}\mathbf{U}^T) / \|\mathbf{A}\|, \ \mathbf{U} \in M_{pcn}. \tag{2.6}$$

Q_g је валидна генерализација Њуман-Гирванове модуларности, јер се за строге c-партиције своди на 2.2. Додатно, уколико је матрица $\mathbf{U}$ фази c-партиција ($\mathbf{U} \in M_{fcn}$), Q_g се назива *фази модуларност* матрице $\mathbf{U}$.

Модуларност се користи као оцена квалитета у многим алгоритмима, посебно код алгоритама заснованих на оптимизацији. Алгоритам `FFMM` који директно максимизује фази модуларност биће приказан у глави 3. С обзиром да максимална вредност модуларности расте са повећањем мреже и броја добро раздвојених кластера, модуларност не би требало користити за поређење квалитета у случају мрежа различитих величина.

2.2.2 E-функција

Да би се превазишли недостаци модуларности, развијена је експоненцијална E-функција за оцену квалитета дисјунктног кластеровања [5, 4]. Истраживања су показала да она није подложна проблемима на које наилази модуларност, при чему има велики потенцијал за откривање кластера у комплексним мрежама. Поставља се питање да ли се ова функција може модификовати за спровођење преклапајућег кластеровања.

Идеја E-функције огледа се у квантитативном представљању квалитета унутрашње структуре кластера, као и његове повезаности са осталим кластерима мреже. Добар представник унутрашње структуре кластера k јесте његова густина d_k. Међутим, у густим мрежама (или комплетним), велика густина кластера ће бити последица велике густине целокупне мреже, односно графа G. Стога, уместо густине кластера, боље је посматрати разлику $(d_k - d_G)$. У том случају би сваки подграф са већом густином од густине целог графа био добар кандидат за кластер. Проблем се јавља у случају малих подграфова. Сваки подграф који се састоји од 2 повезана чвора би био идеалан кластер. Да би се то превазишло, вредност $(d_k - d_g)$ се множи бројем чворова у кластеру k, односно n_k. Додатно, може се искористити експоненцијална фунцкија због уочљивости малих промена у густини или броју чворова кластера. Стога, квалитет унутрашње структуре кластера k у графу G се може представити формулом:

$$EQ^+(k) = \begin{cases} e^{n_k(d_k - d_G)}, & n_k \neq 1 \\ 0, & n_k = 1 \end{cases}. \tag{2.7}$$

Са друге стране, оно што повезује кластер са другим кластерима у мрежи јесте број спољашњих грана l_k, односно грана између чворова унутар кластера k и чворова из других кластера. С обзиром да очекивани број спољашњих

грана расте са порастом броја чворова унутар кластера k, за повезаност кластера са другим кластерима у мрежи се може користити вредност $r\frac{2l_k}{n_k}$, при чему параметар r регулише утицај спољашњих грана на укупан квалитет кластера. Дакле, коришћењем експоненцијалне функције, имамо:

$$EQ^{-}(k) = e^{r\frac{2l_k}{n_k}}. \tag{2.8}$$

Укупан квалитет кластера k се дефинише као разлика $EQ^{+}(k) - EQ^{-}(k)$, док је укупан квалитет партиције $\mathcal{P}$:

$$\begin{aligned} EQ(\mathcal{P}) &= \sum_{k=1}^{c} \Big[EQ^{+}(k) - EQ^{-}(k) \Big] \\ &= \sum_{k=1}^{c} \Big[e^{n_k(\frac{2m_k}{n_k(n_k-1)} - \frac{2m}{n(n-1)})} - e^{r\frac{2l_k}{n_k}} \Big]. \end{aligned} \tag{2.9}$$

Поставља се питање како одредити густину кластера, као и број спољашњих грана у случају преклапајућих чворова. Када су познате само вероватноће припадања потенцијалним кластерима, број чворова и грана није смислено дефинисати као целобројну вредност.

Нека је дат граф $G = (V, E)$ и нека је $\mathbf{U}$ фази c-партиција скупа V, $1 < c < n$. С обзиром да се вредност u_{ki} може посматрати као део чвора v_i који припада кластеру k, *фази број чворова* у кластеру k ће бити

$$n_{fk} = \sum_{i=1}^{n} u_{ki}.$$

Пошто је грана унутрашња за кластер k уколико њени крајњи чворови припадају том кластеру, вероватноћа припадања гране између чворова i и j кластеру k износи $u_{ki}u_{kj}$. Стога, *фази број īрана* у кластеру k дефинише се као

$$m_{fk} = \sum_{i=1}^{n} \sum_{j=i}^{n} a_{ij} u_{ki} u_{kj}.$$

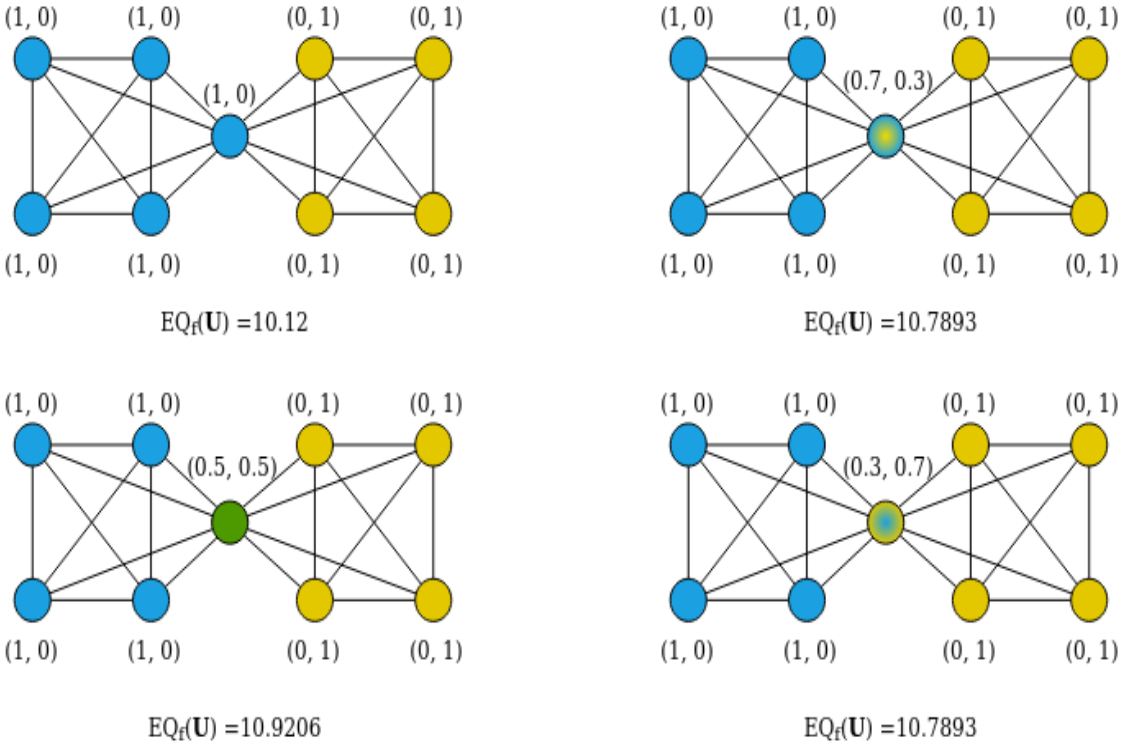
Аналогно, грана је спољашња за кластер k ако му припада тачно један њен крај, па имамо

$$l_{fk} = \sum_{i=1}^{n} \sum_{j=i}^{n} a_{ij}(u_{ki}(1 - u_{kj}) + u_{kj}(1 - u_{ki})).$$

На основу једнакости (2.9), *фази E-квалитет* c-партиције $\mathbf{U}$ дефинишемо као:

$$EQ_f(\mathbf{U}) = \sum_{k=1}^{c} \left[e^{n_{fk}\left(\frac{2m_{fk}}{n_{fk}(n_{fk}-1)} - \frac{2m}{n(n-1)}\right)} - e^{r\frac{2l_{fk}}{n_{fk}}} \right].$$

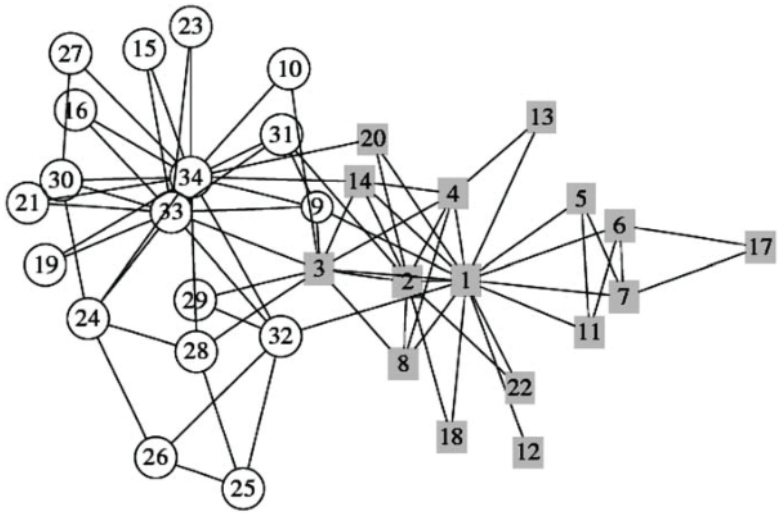
На слици 2.2 приказане су вредности E-функције за различите вредности u_{ki}. Као што је очекивано, максимум функције се достиже када „средњи" чвор припада кластерима са једнаким вероватноћама.



Слика 2.2: Вредности фази E-функције добијене варирањем степена припадања преклапајућег чвора, на графу који се састоји од 2 клике K_4 међусобно повезаних преко једног чвора. Степени припадања кластерима дати су у облику (u_{1i}, u_{2i}), $1 \leq i \leq 9$

Експериментални резултати

На слици 2.4 дат је графички приказ вредности функција Q_g і EQ_f на фази партицијама добијеним као резултат извршавања алгоритма `FFMM`. Алгоритам `FFMM` је базиран на итеративном побољшавању текућег решења тако да се вредност функције Q_g максимизује (детаљи алгоритма су представљени у глави 3). За представљање резултата коришћени су скупови података *За-*

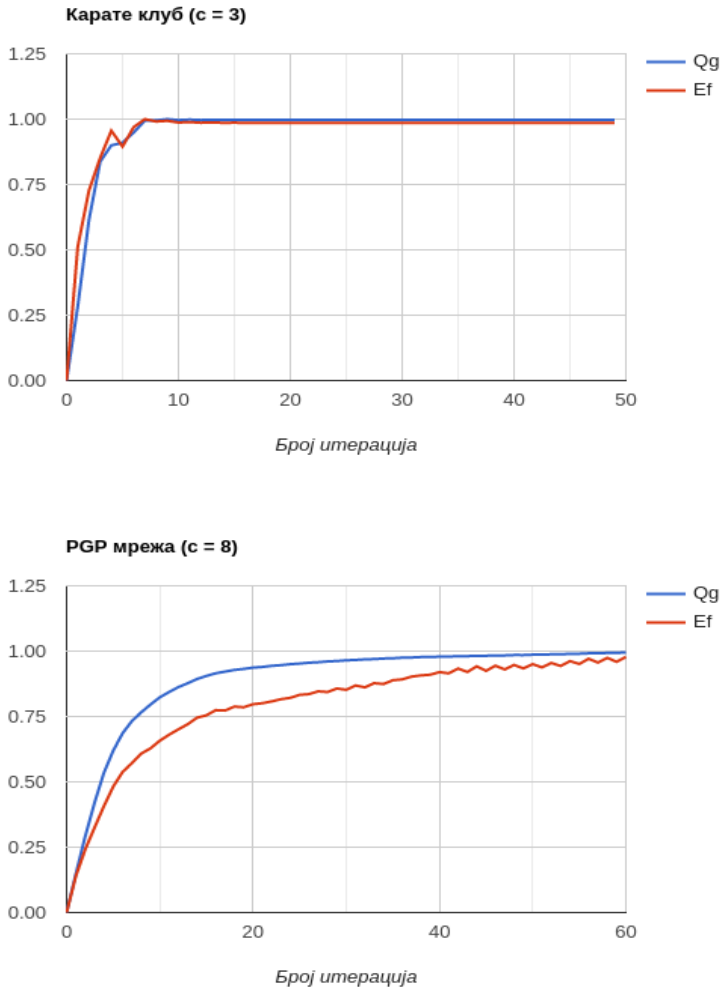


Слика 2.3: Захаријев карате клуб

харијев карате клуб (слика 2.3), са 34 чвора и 78 ивица [19], и *PGP мрежа*, са 10680 чворова и 24316 ивица [24].

Вредности функција добијене у 50 итерација алгоритма су нормализоване, тако да припадају интервалу $[0, 1]$. Разлог томе је то што оригиналне вредности функција припадају различитим скалама. На пример, на PGP мрежи, за 50 добијених вредности важи $Q_g \in [0.0735, 0.665]$ и $EQ_f \in [-240.003, 180.682]$.

Може се приметити да тренд раста E-функције у великој мери прати тренд раста функције Q_g, која се често користи у литератури за оцену квалитета фази кластеровања (детаљна анализа функције Q_g дата је у раду [12]). С обзиром да је Захаријев карате клуб мрежа са малим бројем чворова, до конвергенције долази у малом броју итерација. На PGP мрежи, иако се просечна вредност E-функције повећава са бројем итерација, у каснијим итерацијама долази до осцилација у њеној вредности. То је и очекивано, због осетљивости експоненцијалне функције на мале промене аргумента, па било какве промене у партицији значајно утичу на њену вредност.



Слика 2.4: Анализа вредности фази E-функције и фази модуларности на Захаријевом карате клубу и PGP мрежи као скуповима података

Глава 3

Алгоритми за фази кластеровање на комплексним мрежама

Због разноврсности реалних комплексних мрежа, постоји огроман број приступа решавању проблема преклапајућег кластеровања. Важно је нагласити да ниједан приступ није савршен, односно не постоји општи шаблон за конструкцију алгоритама који раде са комплексним мрежама. У првом делу овог поглавља биће дат општи преглед метода, а потом и два конкретна алгоритма за преклапајуће кластеровање, која су направила значајан помак по питању временске и просторне сложености извршавања.

3.1 Општи преглед

По начину на који врше детекцију кластера, алгоритми за фази кластеровање се оквирно могу поделити на алгоритме засноване на *чворовима*, *мрежи* или *хијерархијском кластеровању* [3].

Основна идеја алгоритама заснованих на чворовима огледа се у томе да слични чворови мреже имају већу вероватноћу да заврше у истом кластеру. Најпопуларнији метод овог типа јесте метод *филтрирања кластера* [20]. Овај алгоритам је заснован на тражењу највећих комплетних подграфова у мрежи. Међутим, његова мана је веома висока сложеност, па је за велике мреже неупотребљив.

Алгоритми засновани на мрежи се базирају на методама за рад са графовима. Они се могу поделити на:

- *методе сегментације графа* – већина метода овог типа своди се на поделу графа на два дела, при чему се сваки део поново дели док се не добије одређени број подграфова. Пример алгоритма овог типа је *спектрално кластеровање* [22], засновано на сопственим векторима матрице графа. Међутим, за његово извршавање потребан је унапред познат број чворова унутар кластера, као и сам број кластера.

- *методе пропагације ознака* – ове методе ажурирају ознаке кластера којима чвор припада на основу сличности са другим чворовима. Алгоритме овог типа углавном карактерише ниска сложеност извршавања, па могу бити примењене на мреже са великим бројем чворова. Међутим, процес ажурирања ознака може бити у великој мери случајан, што доводи до лоше стабилности и партиција мреже. Тенденцију ка превазилажењу ових проблема показао је алгоритам максимизације вектора припадања и биће приказан на крају овог поглавља.

- *методе дубоког учења* – са развојем дубоког учења, омогућени су нови приступи кластеровању, с обзиром на способност неуронских мрежа да науче карактеристике комплексне мреже. У раду [21] предложен је начин за пресликавање чворова у латентни простор. Слично томе, у раду [26] велике мреже се пресликавају у нискодимензиони векторски простор. То омогућава извршавање алгоритама на мрежама великих размера.

Циљ хијерархијског кластеровања јесте формирање стабла састављеног од кластера, тако да се мрежа може анализирати на различитим нивоима. Један представник овог типа јесте *раздвајајуће хијерархијско кластеровање*. Уопштено говорећи, ова метода третира целу мрежу као један кластер, а потом дели тај кластер на више мањих користећи метод сегментације графа. Нажалост, овај алгоритам је због своје временске сложености практично неупотребљив за мреже великих размера. Са друге стране, *аглоративно хијерархијско кластеровање* подразумева формирање бинарног стабла кластеровања (дендрограм), на основу степена различитости између чворова. То стабло представља хијерархију партиција. Одсецањем грана стабла на различитим нивоима, могу се добити партиције различитих структура.

У наставку следи алгоритам брзе максимизације фази модуларности, заснован на идеји математичке оптимизације, и његова модификација, заснована на хијерархијском кластеровању. Са приближно линеарним временом извршавања, овај алгоритам је посебно важан за кластеровање на огромним комплексним мрежама.

3.2 Алгоритам брзе максимизације фази модуларности

Алгоритам брзе максимизације фази модуларности (енгл. *Fast Fuzzy Modularity Maximization* - `FFMM`), представљен у раду [27], користи једначине за итеративно израчунавање повећања модуларности при промени степена припадања чворова кластерима. Развијена метода омогућава ефикасну модификацију модуларности, сводећи рачунску сложеност на линеарну функцију броја чворова и заједница у мрежи. Да би се додатно смањила рачунска сложеност за веома велике мреже, развијен је вишециклусни `FFMM` (енгл. *Multicycle Fast Fuzzy Modularity Maximization* - `McFFMM`). Наиме, разбијајући мрежу на више подмрежа и примењујући `FFMM` на сваку од њих, овај алгоритам омогућава да се детекција заједница изврши у реалном времену за велике мреже.

3.2.1 Ажурирање колона партиционе матрице

Идеја алгоритма `FFMM` је следећа: ажурирати сваку колону партиционе матрице (односно ажурирати вероватноће припадања једног чвора потенцијалним кластерима), тако да се модуларност повећа након сваког ажурирања. Да бисмо то постигли, докажимо прво следећу лему:

Lema 3.2.1. *За дати граф* $\mathbf{G}$ *ажурирањем* n*-те колоне партиционе матрице* $\mathbf{U}$ *добијене у итерацији* $(k-1)$*, из* $\mathbf{u}_n^{(k-1)}$ *у* $\mathbf{u}_n^{(k)}$*, модуларност се мења на следећи начин:*

$$\Delta Q\Big(\mathbf{u}_n^{(k-1)} \to \mathbf{u}_n^{(k)}\Big) = \frac{1}{\|\mathbf{A}\|}\Bigg[2\Big(\mathbf{u}_n^{(k)} - \mathbf{u}_n^{(k-1)}\Big)^T \tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n + b_{nn}\Big[\Big(\mathbf{u}_n^{(k)}\Big)^T\mathbf{u}_n^{(k)} - \Big(\mathbf{u}_n^{(k-1)}\Big)^T\mathbf{u}_n^{(k-1)}\Big]\Bigg], \tag{3.1}$$

ӣде је $\tilde{\mathbf{U}}^{(k-1)}_{[n]} = [u_1^{(k-1)}, ..., \mathbf{u}_{n-1}^{(k-1)}, \mathbf{0}_{c\times 1}, \mathbf{u}_{n+1}^{(k-1)}, ..., \mathbf{u}_N^{(k-1)}]$ *и* $\mathbf{b}_n$ *n-ӣа колона маӣрице модуларносӣи* $\mathbf{B}$ *дефинисане у одељку 1.1.3.*

Доказ. По дефиницији, важи

$$\Delta Q\Big(\mathbf{u}_n^{(k-1)} \to \mathbf{u}_n^{(k)}\Big) = \frac{tr\Big(\mathbf{U}^{(k)}\mathbf{B}(\mathbf{U}^{(k)})^T - \mathbf{U}^{(k-1)}\mathbf{B}(\mathbf{U}^{(k-1)})^T\Big)}{\|\mathbf{A}\|}.$$

Ако матрицу $\mathbf{U}$ представимо као $\tilde{\mathbf{U}}+\check{\mathbf{U}}$, где је $\tilde{\mathbf{U}} = [\mathbf{u}_1, ..., \mathbf{u}_{n-1}, \mathbf{0}_{c\times 1}, \mathbf{u}_{n+1}, ..., \mathbf{u}_N]$ и $\check{\mathbf{U}} = [\mathbf{0}_{c\times 1}, ..., \mathbf{0}_{c\times 1}, \mathbf{u}_n, \mathbf{0}_{c\times 1}, ..., \mathbf{0}_{c\times 1}]$, онда важи:

$$\begin{aligned}\Delta Q\Big(\mathbf{u}_n^{(k-1)} \to \mathbf{u}_n^{(k)}\Big) = \frac{1}{\|\mathbf{A}\|} tr\Bigg[&\Big(\tilde{\mathbf{U}}^{(k)} + \check{\mathbf{U}}^{(k)}\Big)\mathbf{B}\Big(\tilde{\mathbf{U}}^{(k)} + \check{\mathbf{U}}^{(k)}\Big)^T - \\ &\Big(\tilde{\mathbf{U}}^{(k-1)} + \check{\mathbf{U}}^{(k-1)}\Big)\mathbf{B}\Big(\tilde{\mathbf{U}}^{(k-1)} + \check{\mathbf{U}}^{(k-1)}\Big)^T\Bigg].\end{aligned} \tag{3.2}$$

Пошто се мења само n-та колона матрице $\mathbf{U}$, може се закључити да важи $\tilde{\mathbf{U}}^{(k)} = \tilde{\mathbf{U}}^{(k-1)} = \tilde{\mathbf{U}}$. Ако то заменимо у (3.2), након поједностављивања израза добијамо:

$$\begin{aligned}\Delta Q\Big(\mathbf{u}_n^{(k-1)} \to \mathbf{u}_n^{(k)}\Big) = \frac{1}{\|\mathbf{A}\|} tr\Bigg[&\Big(\check{\mathbf{U}}^{(k)} - \check{\mathbf{U}}^{(k-1)}\Big)B\tilde{\mathbf{U}}^T + \tilde{\mathbf{U}}B\Big(\check{\mathbf{U}}^{(k)} - \check{\mathbf{U}}^{(k-1)}\Big)^T + \\ &\check{\mathbf{U}}^{(k)}B\Big(\check{\mathbf{U}}^{(k)}\Big)^T - \check{\mathbf{U}}^{(k-1)}B\Big(\check{\mathbf{U}}^{(k-1)}\Big)^T\Bigg].\end{aligned}$$

Узимајући у обзир да важи $tr(A) = tr(A^T)$, имамо:

$$\begin{aligned}\Delta Q\Big(\mathbf{u}_n^{(k-1)} \to \mathbf{u}_n^{(k)}\Big) = \frac{1}{\|\mathbf{A}\|} tr\Bigg[&2\Big(\Big(\check{\mathbf{U}}^{(k)} - \check{\mathbf{U}}^{(k-1)}\Big)B\tilde{\mathbf{U}}^T\Big) + \\ &\Big(\check{\mathbf{U}}^{(k)}B\Big(\check{\mathbf{U}}^{(k)}\Big)^T - \check{\mathbf{U}}^{(k-1)}B\Big(\check{\mathbf{U}}^{(k-1)}\Big)^T\Big)\Bigg].\end{aligned} \tag{3.3}$$

Приметимо још да важе следеће једнакости:

$$\begin{aligned}tr\Big(\Big(\check{\mathbf{U}}^{(k)} - \check{\mathbf{U}}^{(k-1)}\Big)B\tilde{\mathbf{U}}^T\Big) &= \sum_{i=1}^{C}\sum_{j=1}^{N}\Big(u_{in}^{(k)} - u_{in}^{(k-1)}\Big)b_{jn}\tilde{u}_{ij} \\ &= \sum_{j=1}^{N}\sum_{i=1}^{C}\Big(u_{in}^{(k)} - u_{in}^{(k-1)}\Big)\tilde{u}_{ij}b_{jn} \\ &= \Big(\mathbf{u}_n^{(k)} - \mathbf{u}^{(k-1)}\Big)^T\tilde{\mathbf{U}}\mathbf{b}_n,\end{aligned} \tag{3.4}$$

док се други део једнакости (3.3), због $tr\Big(\check{\mathbf{U}}^{(k)}B(\check{\mathbf{U}}^{(k)})^T\Big) = \sum_{i=1}^{C}\Big(u_{in}^{(k)}\Big)^2 b_{nn}$, поједностављује на следећи начин:

$$tr\Big(\check{\mathbf{U}}^{(k)}B\Big(\check{\mathbf{U}}^{(k)}\Big)^T - \check{\mathbf{U}}^{(k-1)}B\Big(\check{\mathbf{U}}^{(k-1)}\Big)^T\Big) = \sum_{i=1}^{C}\left[\Big(u_{in}^{(k)}\Big)^2 - \Big(u_{in}^{(k-1)}\Big)^2\right]b_{nn}$$
$$= b_{nn}\left(\Big(\mathbf{u}_n^{(k)}\Big)^T\mathbf{u}_n^{(k)} - \Big(\mathbf{u}_n^{(k-1)}\Big)^T\mathbf{u}_n^{(k-1)}\right). \tag{3.5}$$

Заменом (3.4) и (3.5) у (3.3) добија се тражена једнакост.

□

Дакле, циљ је да ажурирање сваке колоне партиционе матрице буде такво да се промена модуларности максимизује. Оно што треба одредити је вектор $\mathbf{u}_n^{(k)}$, односно n-та колона фази c-партиције, такав да је промена модуларности максимална. Другим речима, потребан нам је извод израза из (3.1) по $\mathbf{u}_n^{(k)}$. Штавише, потребна нам је тачка у којој тај извод има вредност 0, тј:

$$\frac{\partial\Big[\Delta Q\Big(\mathbf{u}_n^{(k-1)} \to \mathbf{u}_n^{(k)}\Big)\Big]}{\partial\Big(\mathbf{u}_n^{(k)}\Big)^T} = 2\tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n + 2b_{nn}\mathbf{u}_n^{(k)} = 0.$$

Дакле, тражена тачка је вектор:

$$\mathbf{u}_n^{(k)} = \frac{-\tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n}{b_{nn}},\ n = \overline{1,N}. \tag{3.6}$$

Једнакост (3.6) је кључна за алгоритам `FFMM`. Пошто су дијагонални елементи матрице суседства једнаки нули, дијагонални елементи матрице модуларности $\mathbf{B}$ су мањи од нуле, односно $b_{nn} < 0$, за $n = \overline{1,N}$. Због тога имамо:

$$\mathbf{u}_n^{(k)} = \frac{\tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n}{|b_{nn}|},\ n = \overline{1,N}. \tag{3.7}$$

Предложено ажурирање колона може довести до негативних вредности у партиционој матрици, што није дозвољено. Да би партициона матрица остала конзистентна, негативне вредности се замењују нулом, а потом се колона нормализује, тако да за елементе колоне важи $\sum_{i=1}^{C} u_{in} = 1$. Пошто нормализација мора бити примењена, члан $|b_{nn}|$ може бити изостављен из (3.7). Једнакост (3.7) се тада своди на:

$$\mathbf{u}_n^{(k)} = \tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n,\ n = \overline{1,N}. \tag{3.8}$$

Пажљивим посматрањем матрице $\tilde{\mathbf{U}}_{[n]}^{(k-1)}$ и вектора $\mathbf{b}_n$, може се закључити да важи:

$$\mathbf{u}_n^{(k)} = \tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n = \mathbf{U}^{(k-1)}\tilde{\mathbf{b}}_n, \;\; n = \overline{1, N}. \tag{3.9}$$

где је $\mathbf{U}^{(k-1)}$ партициона матрица добијена у итерацији $(k-1)$, а $\tilde{\mathbf{b}}_n$ представља n-ту колону матрице модуларности B, са нулом на n-тој позицији.

Употреба (3.9) уместо (3.8) омогућава истовремено конструисање целе партиционе матрице. Коначна једначина ажурирања партиционе матрице је

$$\mathbf{U}^{(k)} = \mathbf{U}^{(k-1)}\tilde{\mathbf{B}}, \tag{3.10}$$

при чему $\tilde{\mathbf{B}}$ представља матрицу модуларности $\mathbf{B}$ са нулама по дијагонали.

Псеудокод описаног приступа је приказан алгоритмом 1. Максимизација фази модуларности врши се оптимизујући почетну (случајну) партициону матрицу итеративним множењем описаним једначином (3.10). Међутим, за јако велике мреже, израчунавање производа $\mathbf{U}\tilde{\mathbf{B}}$ у реалном времену је практично немогуће. Стога, потребно је додатно оптимизовати израчунавање траженог производа.

Алгоритам 1: FFMM

Улаз: Модификована матрица модуларности: $\tilde{\mathbf{B}}$; Број кластера: C; Број итерација: K
Излаз: Партициона матрица: $\mathbf{U}$

```
Иницијализовати матрицу U димензије C × N вредностима из
 униформне расподеле, при чему се свака колона сумира на 1;
for k = 2, 3, ..., K do
    U^(k) = U^(k−1) B̃;
    Елиминисати негативне елементе матрице U^(k);
    Нормализовати колоне матрице U^(k), тако да се вредности колона
     сумирају на 1;
end
return U^(k)
```

3.2.2 Ефикасно израчунавање производа $\mathbf{U}\tilde{\mathbf{B}}$

Као што је поменуто у извођењу формуле за итеративно израчунавање партиционе матрице, главни разлог велике рачунске сложености алгоритма 1 је рачунање производа $\mathbf{U}\tilde{\mathbf{B}}$. Циљ је елиминисати вишеструко израчунавање

статичких компоненти производа, односно делова производа који су исти у свакој итерацији алгоритма. Према дефиницији матрице модуларности $\mathbf{B}$, имамо:

$$\mathbf{UB} = \mathbf{UA} - \mathbf{U}\frac{\mathbf{k}\mathbf{k}^T}{\|\mathbf{A}\|}, \tag{3.11}$$

па се n-та колона производа $\mathbf{UB}$ рачуна као:

$$\mathbf{Ub}_n = \mathbf{Ua}_n - \frac{k_n \cdot \mathbf{Uk}}{\|\mathbf{A}\|}, \tag{3.12}$$

где је k_n n-ти елемент вектора $\mathbf{k}$. Примећујемо да је израз $\mathbf{Uk}$ статичка компонента за све колоне производа $\mathbf{UB}$, па може бити претходно израчунат за све колоне. Међутим, (3.10) користи $\mathbf{U\tilde{B}}$. Пошто су сви дијагонални елементи матрице суседства једнаки нули, дијагонални елементи матрице модуларности се рачунају по формули $b_{nn} = -\frac{k_n^2}{\|\mathbf{A}\|}$, за $n = \overline{1, N}$, где је $\mathbf{k}$ вектор степена чворова и k_n његов n-ти елемент. То доводи до

$$\mathbf{b}_n = \tilde{\mathbf{b}}_n - \frac{k_n^2 \mathbf{e}_n}{\|\mathbf{A}\|}, \tag{3.13}$$

где $\mathbf{e}_n$ означава вектор димензије N, при чему је $e_i = 1$ за $i = n$ и $e_i = 0$ за $i \neq n$. Заменом (3.13) у (3.12) добијамо

$$\mathbf{U}\tilde{\mathbf{b}}_n = \mathbf{Ua}_n - \frac{k_n \cdot \mathbf{Uk}}{\|\mathbf{A}\|} + \frac{k_n^2 \cdot \mathbf{Ue}_n}{\|\mathbf{A}\|}.$$

Једноставним манипулацијама, претходни израз своди се на

$$\mathbf{U}\tilde{\mathbf{b}}_n = \mathbf{Ua}_n + \frac{(k_n \cdot \mathbf{u}_n - \mathbf{Uk})\, k_n}{\|\mathbf{A}\|}, \tag{3.14}$$

где је члан $\mathbf{Uk}$ статичка компонента, с обзиром на то да не зависи од n, па може бити израчуната једном за све колоне матрице $\mathbf{U\tilde{B}}$. Штавише, i-та координата производа $\mathbf{Ua}_n$ у једнакости (3.14), може се написати као

$$(\mathbf{Ua}_n)_i = \sum_{l \in \mathbb{N}_n} w_{ln} u_{il},\ i = \overline{1, C},\ n = \overline{1, N}, \tag{3.15}$$

где $\mathbb{N}_n$ представља скуп свих суседа чвора n.

На тај начин је сложеност израчунавања матрице $\mathbf{U\tilde{B}}$ смањена. Да би се додатно повећала ефикасност предложеног алгоритма, велике мреже се у циклусима деле у подмреже, на које се у сваком циклусу примењује алгоритам 1. Због начина на који тај алгоритам ради, назван је *вишециклусни* ***FFMM*** (`McFFMM`) и биће детаљније описан у следећем одељку.

3.2.3 Вишециклусни FFMM за комплексне мреже

Алгоритам McFFMM садржи две модификације алгоритма FFMM. Прво, израчунавање производа $\mathbf{U}\tilde{\mathbf{B}}$ је замењено својом рачунски ефикаснијом верзијом, Друго, FFMM се примењује у више циклуса. Идеја је да кластери које алгоритам FFMM детектује у једном циклусу постану подмреже које ће бити обрађене у следећем циклусу.

У првом циклусу, применом алгоритма FFMM открива се неколико суперкластера, од којих сваки садржи више мањих кластера. У следећем циклусу, сваки суперкластер се посматра као једна подмрежа на коју се примењује FFMM да би се открили кластери унутар њега. Та процедура се понавља ради откривања кластера са већом резолуцијом. Псеудокод је дат алгоритмом 2, при чему су кораци описани на следећи начин.

1. McFFMM почиње применом алгоритма FFMM да би се детектовало $c^{(1)}$ подмрежа (линија 2 алгоритма 2). FFMM је приказан алгоритмом 1, при чему је резултат тог алгоритма партициона матрица $\mathbf{U}^{(1)}$.

2. Број детектованих подмрежа се поставља за први циклус ($C^{(1)} = c^{(1)}$), пре почетка другог циклуса (линија 3 алгоритма 2).

3. Конструкција подмрежа примењује се на партициону матрицу добијену у претходном циклусу, тј. $\mathbf{U}^{(l-1)}$ (линија 5 алгоритма 2). Као резултат, добија се матрица суседства за сваку од подмрежа.

4. За сваку подмрежу, генерише се случајна партициона матрица $\mathbf{U}_j^{(l)} \in \mathbb{R}^{c_j^{(l)} \times N_j^{(l)}}$ са нормализованим колонама, при чему $N_j^{(l)}$ означава број чворова j-те подмреже у l-том циклусу, а $c_j^{(l)}$ број кластера који ће бити детектовани унутар j-те подмреже у l-том циклусу (линија 7 алгоритма 2). За вредности $c_j^{(l)}$ су предефинисане вредности $c^{(l)}$ и оне представљају највећи број кластера који може бити детектован унутар неке подмреже у l-том циклусу. Вредности $c_j^{(l)}$ се рачунају по формули:

$$c_j^{(l)} = c^{(l)} \frac{N_j^{(l)}}{\max\limits_i \left(N_i^{(l)} \right)}, \; j = \overline{1, C^{(l-1)}}, \; l \geq 2. \tag{3.16}$$

.

5. FFMM се примењује на сваку доступну подмрежу ради откривања $c_j^{(l)}$ кластера, за $j = \overline{1, C^{(l-1)}}$ (линија 8 алгоритма 2).

6. Партициона матрица $\mathbf{U}^{(l)}$ која одговара читавој мрежи формира се на начин описан у секцији 3.2.5.

7. Укупан број детектованих кластера (односно број подмрежа за наредни циклус) се рачуна по формули $C^{(l)} = \sum_{j=1}^{C^{(l-1)}} c_j^{(l)}$ (линија 11 алгоритма 2), а затим се прелази на следећи циклус.

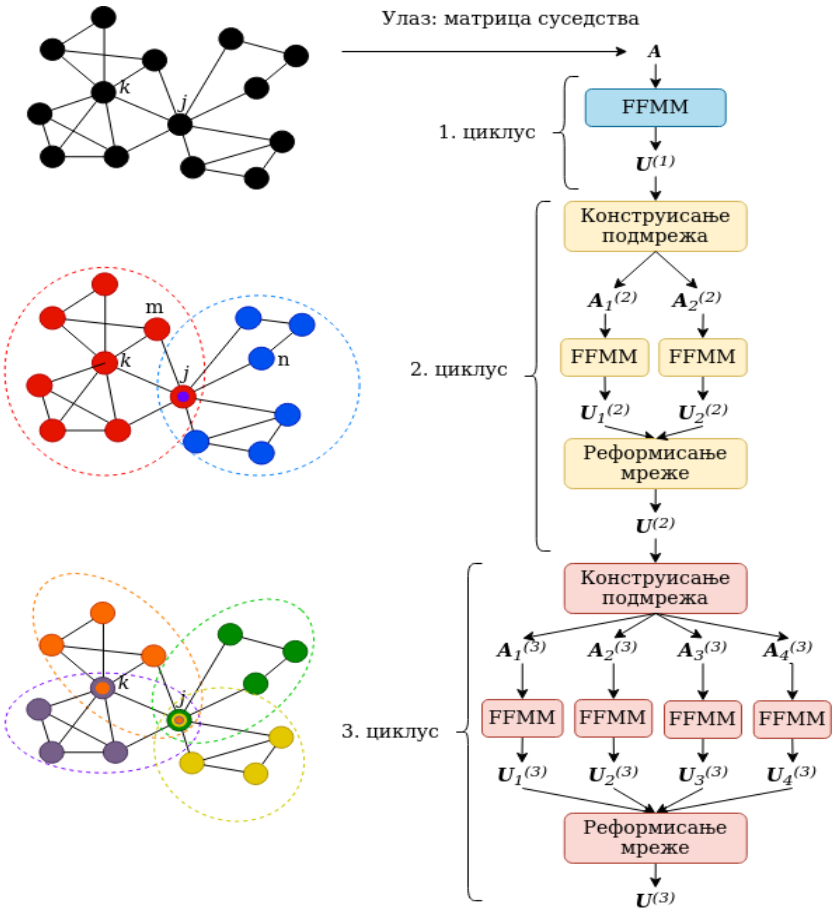
Алгоритам 2: McFFMM

Улаз: Максималне вредности броја кластера: $c^{(1)}, c^{(2)}, ..., c^{(L)}$;
Матрица суседства: $\mathbf{A}$
Излаз: Партициона матрица: $\mathbf{U}^{(L)}$

```
1  Иницијализовати матрицу U^(1) димензије c^(1) × N;
2  Применити FFMM на U^(1) и одредити c^(1) кластера;
3  Поставити C^(1) = c^(1);
4  for l = 2, 3, ..., L do
5  |  Конструисати C^(l−1) подмрежа, тј. A_j^(l) за j = 1, 2, ..., C^(l−1), од
   |   кластера добијених у циклусу (l − 1);
6  |  for j = 1, 2, ..., C^(l−1) do
7  |  |  Иницијализовати U_j^(l) матрицом димензије c_j^(l) × N_j^(l) користећи
   |  |   формулу (3.16);
8  |  |  Применити FFMM на A_j^(l) и тако одредити c_j^(l) кластера, односно
   |  |   U_j^(l);
9  |  end
10 |  Реконструисати U^(l) помоћу U_j^(l), за j = 1, 2, ..., C^(l−1), на начин
   |   описан у секцији 3.2.5;
11 |  Поставити C^(l) = Σ_{j=1}^{C^(l−1)} c_j^(l);
12 end
```

3.2.4 Конструисање подмрежа

Главни циљ конструисања подмрежа јесте да се уклоне ивице између чворова који припадају различитим подмрежама. Тај процес је од велике важности да би FFMM могао бити независно примењен на сваку подмрежу. Као улаз, ова процедура узима партициону матрицу $\mathbf{U}^{(l)}$, а затим генерише матрицу суседства за сваку подмрежу. На слици 3.1, у другом циклусу, то су матрице $\mathbf{A}_1^{(1)}$ и $\mathbf{A}_2^{(1)}$. Да би идентификоване подмреже биле међусобно раздвојене, елиминишу се гране између одговарајућих чворова. На пример, на



Слика 3.1: Дијаграм тока алгоритма `McFFMM`

слици 3.1 чвор j повезан је са чворовима из оба детектована кластера. Стога, ивице које повезују j и чворове из друге подмреже (рецимо n), морају бити уклоњене из матрице суседства прве подмреже, тј. $\mathbf{A}_1^{(1)}$. Аналогно, ивице које повезују j и чворове из прве подмреже (рецимо m), морају бити уклоњене из матрице суседства друге подмреже, тј. $\mathbf{A}_2^{(1)}$.

3.2.5 Реформисање мреже

Процес реформисања мреже подразумева спајање подмрежа, при чему се као резултат формира укупна партициона матрица мреже, која одговара оригиналној мрежи задатој матрицом суседства $\mathbf{A}$. Спајање партиционих матрица које одговарају подмрежама је праволинијски посао у случају дисјунктних кластера, где сваки чвор припада тачно једном кластеру, па је стога само један елемент сваке колоне укупне партиционе матрице различит од

нуле. Међутим, `FFMM` као резултат даје вероватноће припадања кластерима како за преклапајуће, тако и за непреклапајуће чворове.

Посматрајмо слику 3.1. Нека је $\mathbf{u}_{1,j}^{(1)} = \left[u_{1,1j}^{(1)}, u_{1,2j}^{(1)}\right]^T$ вектор припадања j-тог чвора у првом циклусу (односно j-та колона матрице $\mathbf{U}_1^{(1)}$). У другом циклусу, `FFMM` се примењује на пронађене подмреже. Као резултат, прва (црвена) и друга (плава) подмрежа су додатно подељене у две подмреже. Стога, у свакој подмрежи, додељен је вектор припадања чвору j. Пошто чвор j у добијеним подмрежама може имати различит редни број од оног у полазној мрежи, одговарајући вектори ће бити $\mathbf{u}_{1,k}^{(2)}$ (k-колона матрице $\mathbf{U}_1^{(1)}$) и $\mathbf{u}_{2,m}^{(2)}$ (m-та колона матрице $\mathbf{U}_2^{(1)}$), где k и m представљају редне бројеве чвора j у првој и другој подмрежи. Да би се добили степени припадања у тренутном циклусу, степени припадања из претходног циклуса се множе векторима припадања тренутног циклуса. На тај начин је j-та колона укупне партиционе матрице нормализована. Стога, на крају другог циклуса, j-та колона укупне партиционе матрице има четири елемента и рачуна се на следећи начин:

$$\mathbf{u}_j^{(2)} = \left[u_{1,1j}^{(1)}\left(\mathbf{u}_{1,k}^{(2)}\right)^T, u_{1,2j}^{(1)}\left(\mathbf{u}_{2,m}^{(2)}\right)^T\right]^T.$$

Уопштено говорећи, ако чвор n припада кластерима i_1 и i_2 унутар j-те подмреже у циклусу $(l-1)$, онда важи $u_{j,i_1n}^{(l-1)} \neq 0$ и $u_{j,i_2n}^{(l-1)} \neq 0$. У циклусу l, n-та колона $\mathbf{u}_n^{(l)}$ укупне партиционе матрице $\mathbf{U}^{(l)}$ је:

$$\mathbf{u}_n^{(l)} = \left[0, ..., 0, u_{j,i_1n}^{(l-1)}\left(\mathbf{u}_{i_1,k}^{(l)}\right)^T, 0, ..., 0, u_{j,i_2n}^{(l-1)}\left(\mathbf{u}_{i_2,m}^{(l)}\right)^T, 0, ..., 0\right]^T,$$

при чему су $\mathbf{u}_{i_1,k}^{(l)}$ и $\mathbf{u}_{i_2,m}^{(l)}$ k-та и m-та колона партиционих матрица $\mathbf{U}_{i_1}^{(l)}$ и $\mathbf{U}_{i_2}^{(l)}$, редом.

3.2.6 Конвергенција степена припадања

Због природе алгоритма `FFMM`, велики број чворова заврши са вредностима степена припадања које су близу строгих непреклапајућих вредности (0 или 1). То се дешава при великом броју итерација (број K у алгоритму 1). Да би се то превазишло, степен припадања се поставља на најближу непреклапајућу вредност када пређе предефинисани праг τ. Овај приступ гарантује да ће преклапајући чворови бити они који се мање истичу од осталих (односно

имају мању стандардну девијацију степена припадања). У литератури, за τ се обично узима вредност 0.1.

3.2.7 Анализа сложености

Као што је приказано алгоритмима 1 и 2, највећи удео у укупној сложености алгоритма има израчунавање производа $\mathbf{U}\tilde{\mathbf{B}}$. Једнакост 3.14 показује да се главни део рачунских операција користи за израчунавање производа $\mathbf{Uk}$, што захтева CN множења и $C(N-1)$ сабирања по итерацији. Разлог томе је независност компоненте $\mathbf{Uk}$ од колоне за коју се рачуна, па се може израчунати једном за сваку итерацију. Међутим, све остале компоненте се рачунају за сваки чвор и сваку итерацију посебно. То захтева $|\mathbb{N}_n| - 1$ сабирања за $\sum_{l\in\mathbb{N}_n} a_{ln}\mathbf{u}_{il}$, C множења за $k_n\mathbf{u}_n$, $C-1$ сабирања за $(k_n\mathbf{u}_n - \mathbf{Uk})$ и још по једно множење и сабирање за коначну вредност. Стога, за укупан број операција потребних за алгоритам `FFMM` износи $K(CN + N(C+1))$ множења и $K(C(N-1) + 2m + N(C-1))$ сабирања. У О нотацији, пошто K и C не зависе од n и m, сложеност алгортима `FFMM` износи $\mathrm{O}(n+m)$. На исти начин, долази се до сличних израза за алгоритам `McFFMM`:

$$\begin{aligned} &\textit{Број сабирања:} && K\sum_{l=1}^{L}\sum_{j=1}^{C^{(l-1)}}\left[c_j^{(l)}(N_j^{(l)}-1) + 2m_j^{(l)} + N_j^{(l)}(c_j^{(l)}-1)\right] \\ &\textit{Број множења:} && K\sum_{l=1}^{L}\sum_{j=1}^{C^{(l-1)}}\left[c_j^{(l)}N_j^{(l)} + N_j^{(l)}(c_j^{(l)}+1)\right] \end{aligned}$$

С обзиром да је L фиксирано, сложеност алгоритма `McFFMM` је иста као сложеност алгоритма `FFMM`, односно $\mathrm{O}(n+m)$.

3.2.8 Експериментални резултати

Скупови података на којима је примењен алгоритам `McFFMM` приказани су у табели 3.1.

Вредности N и K означавају редом број чворова мреже и број итерација алгоритма 1. Трећа колона табеле означава број подмрежа на које се граф дели у сваком циклусу. Може се приметити да је у случају малих и средњих мрежа примењен алгоритам `FFMM`, односно само једна итерација алгоритма `McFFMM`.

У табели 3.2 приказан је део компаративне анализе, која је представљена у раду [27], алгоритма `McFFMM` и алгоритама: фази агломеративно кластеровање

Табела 3.1: Скупови података и параметри коришћени за тестирање алгоритма `McFFMM`

`Мрежа`	Скр.	N	$[c^{(1)}, c^{(2)}, ...]$	K	Величина
Делфини [15]	D	62	5	50	мала
Фудбал [10]	F	115	10	50	мала
Фејсбук [17]	Fb	4039	15	50	средња
Имејл	E	36692	10, 100	50	велика
Јутјуб	Y	1134890	20, 100	75	огромна

(скр. `FuzAg`) [1], спектрални `FCM` (скр. `FCM/H2`) [25], факторизација Бајесове ненегативне матрице (скр. `NMF`) [23] и фази C средина (скр. `FCM`) [29]. Такође, приказани су и резултати алгоритма `Louvain` [2], који се добро показао као алгоритам заснован на максимизацији модуларности.

Табела 3.2: Просечна модуларност Q_{avg} и просечно време извршавања t_{avg} алгоритама у 100 покретања, у формату Q_{avg}/t_{avg}.

`Мрежа`	`FuzAg`	`FCM/H2`	`NMF`	`FCM`	`Louvain`	`McFFMM`
`D`	0.664/323	0.528/35	0.387/0.294	9.95e-04/0.031	0.519/0.102	0.500/0.002
`F`	0.691/715	0.605/77	0.417/0.538	191e-04/0.031	0.604/0.105	0.592/0.004
`Fb`	-	-	0.72/499	2.22e-04/0.102	0.832/4.06	0.825/2.65
`E`	-	-	-	-	0.584/54.58	0.405/17
`Y`	-	-	-	-	0.712/1038	0.525/686

Видимо да алгоритам `FFMM` надмашује `FCM/H2` и `NMF` у терминима модуларности, за доста краће време извршавања. Штавише, због рачунске сложености, за велике мреже конвергенција ових метода није могућа у реалном времену. `FuzAg` достиже незнатно веће вредности модуларности, али је време извршавања превелико, па је практично неупотребљив за велике мреже. Иако је `FCM` веома брз алгоритам, његово решење није реално због мале вредности модуларности.

Од поменутих приступа, посебно се истиче `Louvain`, као један од најбоље оцењених алгоритама у литератури за фази кластеровање заснованих на модуларности. Иако `Louvain` постиже већу модуларност на свим типовима мрежа, извршавање алгоритма `McFFMM` траје знатно краће.

3.3 Алгоритми засновани на пропагацији ознака

Методе засноване на пропагацији ознака су веома развијене и широко распрострањене. Главна идеја алгоритама заснованих на пропагацији ознака јесте да се сваком чвору додели ознака која ће представљати његов кластер, а затим да се пропагирају одговарајуће ознаке према структури мреже и дистрибуцији ознака, док не дође до конвергенције. Након поменуте пропагације, чворови у истој заједници ће имати исту ознаку. Класични алгоритам за пропагацију ознака (LPA) је првобитно развијен за детекцију непреклапајућих заједница, након чега је проширен и за случај преклапајућих заједница.

Уопштено говорећи, методе засноване на пропагацији ознака у случају фази кластеровања, састоје се од четири дела: иницијализација, пропагирање ознака, партиционисање и идентификација преклапајућих чворова (слика 3.2). У првом делу, сваком чвору се додели бафер за складиштење информација о ознакама. Величина бафера не мора бити фиксирана, али мора имати максимум. При пропагирању ознака, методе углавном итерирају све док ознаке не буду пропагиране кроз целу мрежу, при чему долази до конвергенције. Да би се утврдило да ли је дошло до конвергенције, потребно је одредити разлику између тренутне и мреже након последње итерације. При партиционисању, најмање једна ознака мора бити додељена сваком чвору. Неке методе деле чворове у групе на основу информација чворова у сопственим баферима, док остале праве поделу на основу информација у баферима суседних чворова. У последњој фази алгоритма, чвор ће бити преклапајући уколико је садржан у две или више заједница. Различите методе одређују овакве чворове на различите начине.

Слика 3.2: Фазе алгоритама заснованих на пропагацији ознака

Један од алгоритама базираних на пропагацији ознака је *алгоритам за пропагацију преклапајућих заједница* (*Community Overlap Propagation Algorithm* - COPRA) [11], који омогућава додељивање више ознака сваком чвору, повезаних

са коефицијентима припадности различитим кластерима. Са друге стране, алгоритам `SpeakEasy` [7], поред локалних информација, користи и очекивану фреквенцију ознака, коју формира на основу целокупне мреже. Овај алгоритам комбинује „одоздо-нагоре" приступ кластеровању (користећи ознаке суседних чворова) са „одозго-надоле" приступом (користећи ознаке у читавој мрежи). На тај начин је укључен како локални, тако и глобални распоред ознака. Дијаграм тока алгоритма SpeakEasy приказан је на слици 3.3, при чему је опис корака дат у наставку.

1. `Иницијализација` – Да би се иницијализовали бафери чворова целе мреже, редни бројеви свих чворова се почетно постављају као потенцијалне ознаке кластера (у неким примерима су искоришћена слова уместо редних бројева чворова, ради јасније репрезентације чворова). На почетку, редни број сваког чвора смешта се у бафер тог чвора, а затим се од редних бројева суседних чворова бројеви насумично бирају, све док се бафери не попуне.

2. `Пропагација ознака` – Да би се ознаке итеративно пропагирале, потребно је да сваки чвор ажурира свој бафер, смештајући своју „најзначајнију" ознаку у бафер суседа. Пошто су бафери иницијално пуни, неопходно је направити места за ознаку која се додаје. Стога, поставља се питање како одредити значај ознаке, али и коју ознаку из бафера суседа је најбоље избацити. Значај ознаке одређује се на основу разлике између њене локалне расподеле (унутар бафера суседа) и глобалне расподеле (унутар бафера свих чворова). Другим речима, што се више појављује у локалном скупу бафера, а мање у глобалном скупу бафера, то је ознака значајнија. Да би број ознака у баферу остао исти након додавања најзначајније ознаке, из бафера се избацује прва ознака. Поступак се понавља итеративно, све до конвергенције.

3. `Једнократно партиционисање` – Након конвергенције претходно описаног поступка, сваком чвору додељује се ознака која је најфреквентнија у баферима суседних чворова. Та ознака представља кластер коме чвор припада. Резултат једнократног партиционисања је партиција P.

4. `Заједничко партиционисање` – Кораци 1-3 се понављају N пута, ради добијања N партиција кандидата $\{P_1, P_2, ..., P_N\}$. Партиција која је нај-

сличнија осталим партицијама бира се као коначна непреклапајућа (дисјунктна) партиција, у ознаци $P^* = \{C_1^*, C_2^*, ..., C_K^*\}$, при чему је C_i^* i-ти кластер у партицији P^*. Уколико је циљ налажење дисјунктних кластера, алгоритам може стати овде. У супротном, наставља се у кораку 5.

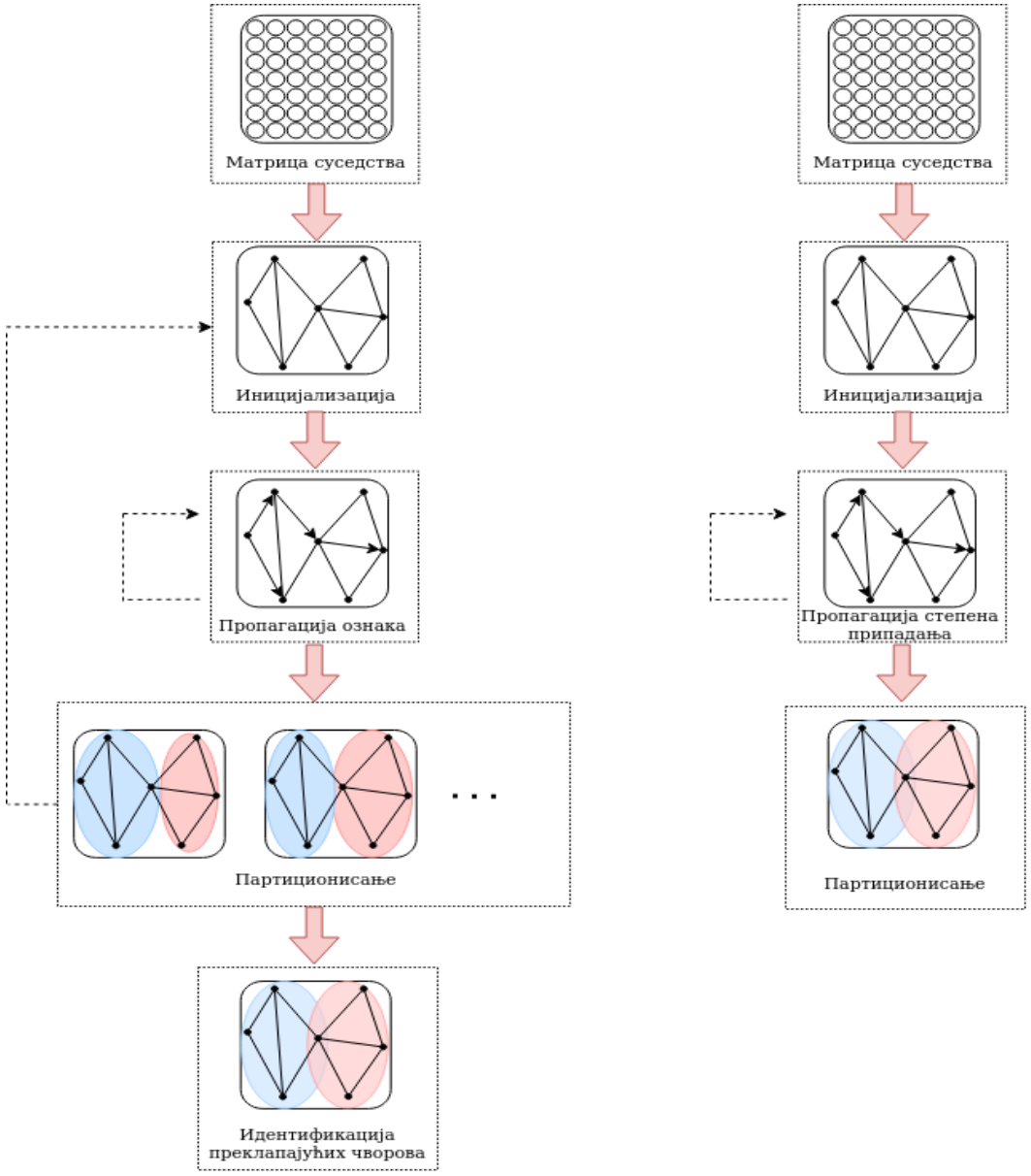
5. `Идентификација преклапајућих чворова` – Ако са a_{ij} означимо број партиција у којима су v_i и v_j припали истом кластеру, при чему је $a_{ii} = 0$, онда је матрица истовременог припадања A конструисана. Под претпоставком да чвор v_i не припада кластеру C_j^* у партицији P^*, тежина припадања чвора v_i кластеру C_j^* дефинише се са:

$$w_{v_i}^{C_j^*} = \frac{\sum_{u \in C_j^*} a_{u,v_j}}{|C_j^*| \cdot N},$$

где је N број партиција, а $|\cdot|$ величина кластера. Уколико је тежина довољно велика, чвор v_i се сматра преклапајућим чвором кластера C_j^*. У раду [8], за праг се узима вредност $1/K_{max}$, где је K_{max} највећи број кластера од N партиција. Међутим, у пракси се показало да такав праг доводи до идентификовања превише преклапајућих чворова, због чега је јако тешко прилагодити праг. Осим тога, `SpeakEasy` захтева генерисање партиција много пута да би се добио добар резултат, што може бити рачунски захтевно. Да би се превазишле наведене слабости, развијен је *алгоритам пропагације степена припадања* (енгл. *Membership Degree Propagation Algorithm* - `MDPA`) [9]. Наиме, уместо пропагације ознака, пропагира се степен, односно вероватноћа, са којом чвор припада неком кластеру. `MDPA` је довео до значајног смањења рачунске сложености и не захтева поновно покретање алгоритма ради постизања преклапајућег партиционисања мреже.

3.3.1 Алгоритам пропагације степена припадања

У алгоритму `SpeakEasy` потребно је поновити процес пропагације ознака N пута, што увелико смањује ефикасност када је мрежа велика. Још важнији проблем је избор прага за идентификацију преклапајућих чворова, што често доводи до њиховог погрешног удела у укупном броју чворова. Стога је развијен алгоритам `MDPA`, који превазилази наведене проблеме. Главна идеја алгоритма јесте дефинисање вектора степена припадања за сваки чвор у мрежи,



Слика 3.3: Дијаграм тока алгоритма `SpeakEasy` са леве стране, и `MDPA` са десне стране. У алгоритму `MDPA`, због пропагације степена припадања уместо ознака, није потребна спољашња петља, а идентификација преклапајућих чворова се своди на читање добијених степена припадања

чији елементи представљају вероватноћу припадања потенцијалним кластерима. За разлику од већине пропагационих метода, врши се пропагација тог вектора уместо ознака. Отуда се алгоритам назива алгоритмом пропагације степена припадања. Алгоритам се грубо састоји из три корака: иницијализација, пропагација степена припадања и партионисање. Дијаграм тока је

приказан на слици 3.3. Важно је нагласити да алгоритам `MDPA` не садржи спољашњу петљу као алгоритам `SpeakEasy`, а идентификација преклапајућих чворова је имплицитно обављена у кораку партиционисања.

Иницијализација

Нека је $V = \{v_1, v_2, ..., v_n\}$ скуп чворова, а E скуп ивица графа $G = (V, E)$. Као и код осталих метода заснованих на пропагацији ознака, конструишемо бафер за сваки чвор мреже. Међутим, разлика је у томе што бафер чува не само информације о ознакама, већ и степен припадања чвора кластеру на који се та ознака односи. Другим речима, степен припадања представља вероватноћу са којом чвор припада потенцијалном кластеру. Стога, бафер чвора v_i означавамо са:

$$b_i = \left\{ \left(l_1^{(i)}, m_1^{(i)}\right), \left(l_2^{(i)}, m_2^{(i)}\right), \cdots, \left(l_{B^{(i)}}^{(i)}, m_{B^{(i)}}^{(i)}\right) \right\}, \ \left(l_j^{(i)} \in \{1, 2, \cdots, n\}, B^{(i)} \leq B\right), \tag{3.17}$$

при чему $l_j^{(i)}$ представља потенцијални кластер чвора v_i, а $m_j^{(i)}$ одговарајући степен припадања чвора v_i кластеру l_j за који важи:

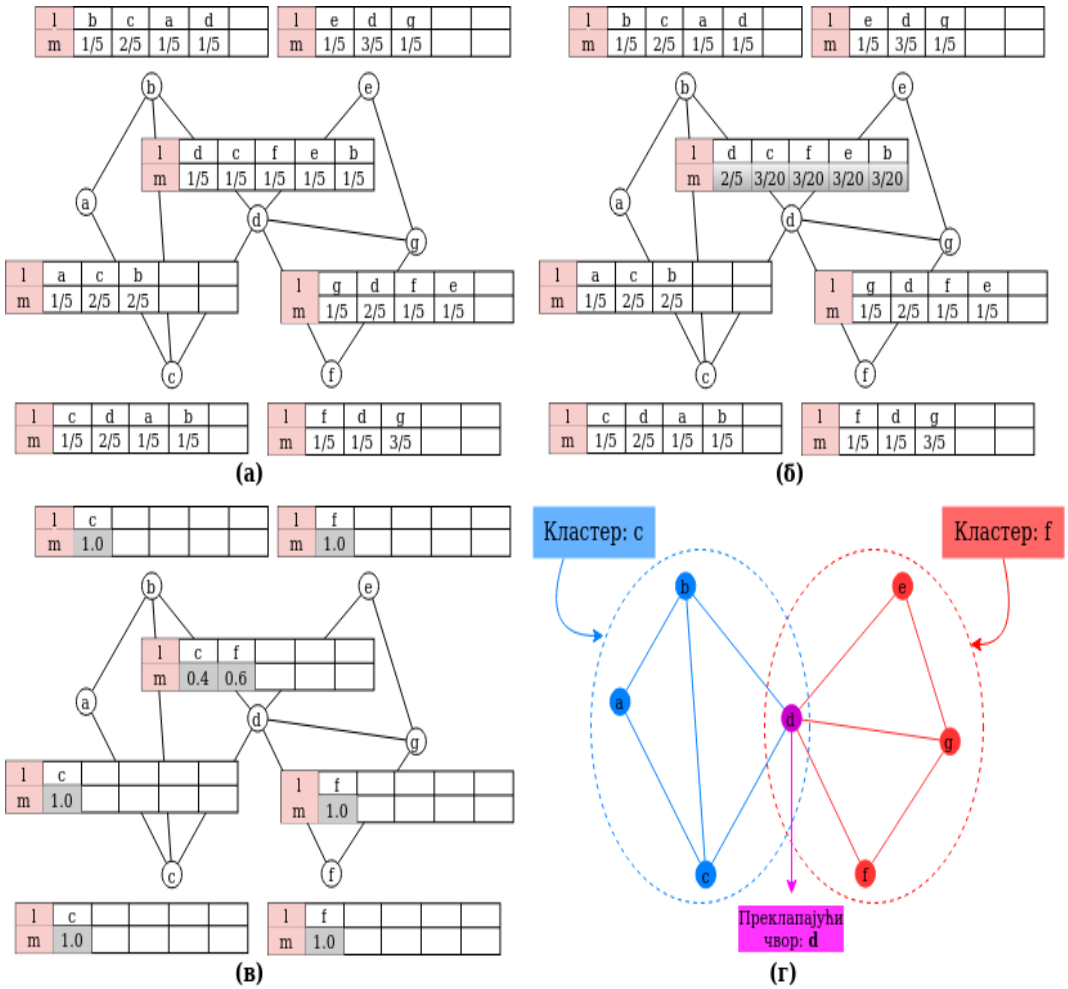
$$\sum_j m_j^{(i)} = 1.$$

Константа B у једнакости (3.17) представља највећи број потенцијалних кластера за сваки чвор, који је постављен на вредност три пута већу од просечног степена чворова. Та вредност је произвољна, и може бити другачија, у зависности од жељених резултата.

На почетку, редни бројеви чворова се узимају за иницијалне ознаке кластера. За сваки чвор, његов редни број се додаје на почетак бафера, заједно са степеном припадања, који се поставља на $1/B$. Након тога, од редних бројева суседа, насумично се бира $B - 1$ бројева, који се са степеном припадања $1/B$ додају у бафер. Уколико се редни број већ налази у баферу, број се не додаје поново, већ се одговарајућа вероватноћа повећава за $1/B$. Због тога, дужина бафера може бити мања од B, али ће у том случају степен припадања бити већи од $1/B$.

Пропагација степена припадања

Циљ пропагације степена припадања јесте повећање степена припадања кластерима са већом локалном дистрибуцијом и мањом глобалном дистрибу-



Слика 3.4: Шематски приказ алгоритма `MDPA`. **(а)** иницијализација, **(б)** пропагација степена припадања, **(в)** стање након пропагације степена припадања, и **(г)** резултат партиционисања. У црвеним колонама, l представља ознаку кластера, а m степен припадања одговарајућим кластерима

цијом. Да би то било могуће, потребно је прецизно дефинисати локалну и глобалну дистрибуцију кластера.

За кластер c, *глобална дистрибуција представља* његову фреквенцију у свим баферима мреже, која се рачуна на следећи начин:

$$g_c = \frac{\sum_j m_c^{(j)}}{n \cdot B}.$$

Локална дистрибуција кластера c у односу на чвор v_i је фреквенција појављивања ознаке c у баферима суседних чворова, која се рачуна по формули:

$$f_c^{(i)} = \frac{\sum_{j \in \mathbb{N}_i} m_c^{(j)}}{|\mathbb{N}_i| \cdot B}.$$

У табели 3.3 приказана је глобална дистрибуција кластера из примера 4а, а у табели 3.4 одговарајућа локална дистрибуција у односу на чвор d.

Чим те вредности буду познате за сваки чвор, може се израчунати разлика између локалне и глобалне дистрибуције за сваку ознаку (кластер). Да би те вредности имале смисла, оне морају бити нормализоване, односно представљене на истој скали (у овом раду та скала је $[0, 5]$). Дакле, *нормализована разлика* за кластер c и чвор v_i рачуна се на следећи начин:

$$d_c^{(i)} = \alpha \cdot \frac{\left(f_c^{(i)} - g_c\right) - \min\limits_j \left(f_j^{(i)} - g_j\right)}{\max\limits_j \left(f_j^{(i)} - g_j\right) - \min\limits_j \left(f_j^{(i)} - g_j\right)},$$

при чему је α параметар за подешавање скале (постављен на 5 у овом раду). Након тога, бира се кластер c чији ће се степен ажурирати унутар бафера чвора v_i, према вероватноћи:

$$p_c^{(i)} = \frac{e^{d_c^{(i)}}}{\sum\limits_{c \in \mathbb{N}_i} e^{d_c^{(i)}}}. \tag{3.18}$$

Табела 3.3: Глобална дистрибуција свих кластера

Ознака	a	b	c	d	e	f	g
Глобална дистрибуција	3/35	5/35	6/35	10/35	3/35	3/35	5/35

Табела 3.4: Локална дистрибуција кластера у односу на чвор d

Ознака	a	b	c	d	e	f	g
Локална дистрибуција	2/25	2/25	3/25	9/25	2/25	2/25	5/25

Поново, узмимо чвор d са слике 3.4а као пример. Оригинална разлика, нормализована разлика и одговарајуће вероватноће за сваки кластер су приказане у табели 3.5.

Дакле, на случајан начин бирамо кластер према вероватноћама из једнакости (3.18). Ако изабрани кластер већ постоји у баферу b_i, одговарајући степен припадања увећава се за $1/B$, при чему се степени припадања осталим кластерима подешавају тако да збир свих степена припадања тог чвора

Табела 3.5: Оригинална разлика, нормализована разлика и одговарајуће вероватноће за сваки кластер у баферима суседа чвора d.

Ознака	a	b	c	d	e	f	g
Оригинална разлика	$-1/175$	$-11/175$	$-9/175$	$13/175$	$-1/175$	$-1/175$	$10/175$
Нормализована разлика	$50/24$	0	$10/24$	5	$50/24$	$50/24$	$105/24$
Одговарајућа вероватноћа	0.0316	0.0039	0.0060	0.5831	0.0316	0.0316	0.3122

остане 1. У супротном, изабрани кластер се додаје у бафер b_i са степеном припадања $1/B$. Ако дужина бафера постане већа од B, уклања се кластер са најмањим степеном припадања. Међутим, може се догодити да постоји више кластера са најмањим степеном припадања. У том случају, кластер за избацивање треба изабрати на случајан начин и подесити бафер тако да сума степена припадања буде 1. У свакој итерацији, сви чворови у мрежи морају да ажурирају свој бафер на описан начин. Алгоритам се завршава када дође до конвергенције или број итерација постане довољно велики. Псеудокод алгоритма пропагације степена припадања приказан је алгоритмом 3.

Алгоритам 3: Пропагација степена припадања

Улаз: Величина бафера: B; Број итерација: NUM; Иницијализовани граф: G'

Излаз: Граф након конвергенције при пропагацији степена припадања: G''

```
1  i = 0;
2  for i < NUM do
3      for v ∈ G' do
4          g ← Израчунати глобалну дистрибуцију свих кластера у
             тренутној мрежи;
5          f^(v) ← Израчунати локалну дистрибуцију свих кластера у
             околини чвора v;
6          L ← Изабрати ознаку (кластер) L према вероватноћи која је
             дата једнакошћу (3.18);
7          G'' ← Повећати степен припадања који одговара кластеру L и
             ажурирати бафер чвора v;
8      end
9      i = i + 1;
10 end
```

У примеру са слике 3.4а, претпостављамо да је случајно изабрани чвор d,

према вероватноћама израчунатим у табели 3.5. Резултат након ажурирања бафера чвора d је приказан на слици 3.4б, а коначан резултат на слици3.4в. Резултат пропагације степена припадања јесте мрежа са израчунатим баферима за сваки чвор. Сваки бафер садржи највише B уређених парова, при чему један уређени пар садржи потенцијални кластер и степен припадања том кластеру.

Партиционисање

Партиционисање се састоји из два дела.

1. Сваком чвору доделити кластер са највећим степеном припадања као *примарни кластер*, односно:

$$v_i \leftarrow \underset{c}{\operatorname{argmax}}\, m_c^{(i)},$$

   где је v_i i-ти чвор, а $m_c^{(i)}$ степен припадања чвора v_i кластеру c.

2. Идентификовати *секундарне кластере* за сваки чвор. За чвор v_i, кластери се додељују на следећи начин:

$$v_i \leftarrow c \quad if \;\; m_c^{(i)} > r, \tag{3.19}$$

   где је r праг припадања, чија је вредност $1/N1$, при чему је $N1$ број примарних кластера додељених чворовима у кораку 1. Стога, $N1$ се може сматрати иницијалним бројем кластера додељеним свим чворовима. Према једнакости (3.19), ако је $m_c^{(i)}$ веће од прага r, чвор v_i се сматра чланом кластера c.

Према томе, можемо једноставно препознати преклапајуће чворове. Уколико је чвору додељено више од једног кластера, чвор се може сматрати преклапајућим. Стога, алгоритам `MDPA` не садржи спољашњу петљу (иницијализација и пропагација), што умногоме смањује рачунску сложеност у односу на алгоритам `SpeakEasy`. Псеудокод корака партиционисања је приказан алгоритмом 4. Коначан резултат кластеровања је приказан на слици 3.4г.

Анализа сложености

У фази иницијализације, да би се бафери попунили, мора се посетити сваки чвор. Пошто су бафери дужине B, потребно је nB операција, односно:

$$N_i = nB.$$

Алгоритам 4: Партиционисање

Улаз: Праг: r; Величина бафера: B; Граф након конвергенције корака пропагације степена припадања: G''
Излаз: Резултат кластеровања: C

```
for v ∈ G'' do
    flag = 0;
    for (l^(v), m^(v)) ∈ b_v do
        // l^(v) је потенцијални кластер чвора v;
        // m^(v) је степен припадања чвора v кластеру l^(v);
        if m^(v) > r then
            l^(v) ← l^(v) ∪ {v};
            C ← C ∪ {l^(v)};
            flag = 1 ;
        end
    end
    // m^(v) ≤ r важи за све m^(v); чвор v је преклапајући и припада
     свим кластерима у баферу b_v ;
    if flag = 0 then
        for (l^(v), m^(v)) ∈ b_v do
            l^(v) ← l^(v) ∪ {v};
            C ← C ∪ {l^(v)};
        end
    end
end
```

У фази пропагације, потребно је N итерација, док подешавања бафера захтевају фиксиран број операција. Означимо тај број са A. При обилажењу чворова, за тренутни чвор v прво се рачуна глобална дистрибуција која захтева nB операција. Потом се рачуна локална дистрибуција у околини чвора v, која захтева највише $|\mathbb{N}_v| \cdot B$ операција, где је $|\mathbb{N}_v|$ број суседа чвора v. Коначно, бафер се подешава помоћу A операција. Дакле, укупан број операција потребан за корак пропагације је:

$$N_p = N \cdot \sum_v (nB + |\mathbb{N}_v|B + A).$$

У јако великим мрежама, n је знатно веће од $|\mathbb{N}_v|$, при чему је промена глобалне дистрибуције проузрокована ажурирањем бафера само једног чвора. Због тога, нова глобална дистрибуција се може ажурирати помоћу неколико операција. Пошто је тај број операција ограничен, можемо претпоставити да

је константан и да је његова вредност Ag. На основу тога, након оптимизације алгоритма, највећи број операција потребан за фазу пропагације јесте:

$$N_p = nB + N \cdot \sum_v (A_g + |\mathbb{N}_v|B + A) = nB + 2mNB + nNA + nNA_g.$$

Другим речима, глобалну дистрибуцију рачунамо једном на почетку, а касније вршимо њено ажурирање након сваке модификације бафера тренутног чвора.

У последњој фази, неопходно је обићи сваки чвор и упоредити сваки степен припадања унутар бафера са прагом r, за шта је потребно nB операција:

$$N_0 = nB.$$

Дакле, укупан број операција који алгоритам `MDPA` захтева је:

$$N_{total} = N_i + N_p + N_0 = 3nB + 2mNB + nNA + nNA_g.$$

С обзиром да сваки бафер садржи највише B потенцијалних кластера, меморијска сложеност алгоритма `MDPA` износи nB меморијских локација.

Детаљна анализа перформанси алгоритма `MDPA` може се видети у [9].

Глава 4

Закључак

Због комплексности великих мрежа, развијање ефикасних метода за фази кластеровање је од велике важности. Поред самих метода, важни су и механизми за мерење квалитета и поређење различитих партиција, који су често уграђени у сам алгоритам. С обзиром на разноврсност реалних примера мрежа, проблем кластеровања и оцене његовог квалитета може бити дефинисан на разне начине.

У оквиру овог рада предложена је модификација E-функције за оцену квалитета фази кластеровања на комплексним мрежама. Резултати постигнути на познатим скуповима података, Захаријевом карате клубу и PGP мрежи, показују да предложена функција има велики потенцијал у одређивању квалитетних фази партиција комплексних мрежа. За разлику од многих мера квалитета фази кластеровања, E-функција не зависи од додатних параметара, већ се директно примењује на фази партицију графа. Такође, у раду је дат општи преглед алгоритама за фази кластеровање на комплексним мрежама, који може служити као почетна тачка у решавању описаног проблема.

Даљи рад биће заснован на идеји директне максимизације E-функције, као и модификације предложене функције за кластеровање над тежинским и усмереним графовима. Да би се повећала вредност E-функције, може се посматрати њена промена при промени тежина припадања чворова потенцијалним кластерима. Са друге стране, због примене експоненцијалне функције, велика осетљивост E-функције на мале промене у кластерима омогућава ефикасно откривање преклапајућих чворова. Стога има смисла дефинисати околине текућег решења, које се састоје од c-партиција са блиским вредностима припадања чворова потенцијалним кластерима. У сваком кораку се прво ге-

нерише скуп c-партиција које чине околину текућег решења, а потом бира најбоље решење у тој околини. На тај начин се проблем фази кластеровања оптимизацијом E-функције може представити као проблем променљивих околина.

Библиографија

[1] Anupam Biswas and Bhaskar Biswas. Fuzag: Fuzzy agglomerative community detection by exploring the notion of self-membership. *IEEE Transactions on Fuzzy Systems*, PP:1–1, 01 2018.

[2] Vincent Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics Theory and Experiment*, 2008, 04 2008.

[3] Qi Chen and Lingwei Wei. Overlapping community detection of complex network: A survey. pages 513–516, 12 2019.

[4] Dušan Džamić. *Nove metode klasterovanja na kompleksnim mrežama*. PhD thesis, Matematički fakultet, Univerzitet u Beogradu, 2021.

[5] Dušan Džamić, Jun Pei, Miroslav Marić, Nenad Mladenovic, and P. Pardalos. Exponential quality function for community detection in complex networks. *International Transactions in Operational Research*, 27, 03 2018.

[6] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486, 06 2009.

[7] Chris Gaiteri, Mingming Chen, Boleslaw Szymanski, Konstantin Kuzmin, Jierui Xie, Changkyu Lee, Tim Blanche, Elias Chaibub Neto, Su-Chun Huang, Thomas Grabowski, Tara Madhyastha, and Vitalina Komashko. Identifying robust communities and multi-community nodes by combining top-down and bottom-up approaches to clustering. *Scientific Reports*, 5:16361, 11 2015.

[8] Esther Galbrun, Aristides Gionis, and Nikolaj Tatti. Overlapping community detection in labeled graphs. *Data Mining and Knowledge Discovery*, 28:1586–1610, 09 2014.

[9] Rui Gao, Shoufeng Li, Xiaohu Shi, Yanchun Liang, and Dong Xu. Overlapping community detection based on membership degree propagation. *Entropy*, 23:15, 12 2020.

[10] Michelle Girvan and Mark Newman. Community structure in social and biological networks. *proc natl acad sci*, 99:7821–7826, 11 2001.

[11] Steve Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12, 10 2009.

[12] Timothy Havens, James Bezdek, Christopher Leckie, Kotagiri Ramamohanarao, and Marimuthu Palaniswami. A soft modularity function for detecting fuzzy communities in social networks. *Fuzzy Systems, IEEE Transactions on*, 21, 12 2013.

[13] Hongjie Jia, Shifei Ding, Xinzheng Xu, and Nie ru. The latest research progress on spectral clustering. *Neural Computing and Applications*, 24, 06 2014.

[14] Manish Kumar, Anurag Singh, and Hocine Cherifi. An efficient immunization strategy using overlapping nodes and its neighborhoods. pages 1269–1275, 04 2018.

[15] David Lusseau, Karsten Schneider, Oliver Boisseau, Patti Haase, Elisabeth Slooten, and Stephen Dawson. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54:396–405, 01 2003.

[16] Lovász László. *Combinatorial Problems and Exercises*. 01 1993.

[17] Jim Mcauley and Jure Leskovec. Learning to discover social circles in ego networks. *NIPS*, 1:539–547, 01 2012.

[18] M. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 74 3 Pt 2:036104, 2006.

[19] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2), Feb 2004.

[20] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, 07 2005.

[21] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 03 2014.

[22] Alex Pothen, Horst Simon, and Kang-Pu Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11, 08 1990.

[23] Ioannis Psorakis, Stephen Roberts, Mark Ebden, and Ben Sheldon. Overlapping community detection using bayesian non-negative matrix factorization. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 83:066114, 06 2011.

[24] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

[25] Jianhai Su and Timothy Havens. Quadratic program-based modularity maximization for fuzzy community detection in social networks. *Fuzzy Systems, IEEE Transactions on*, 23:1356–1371, 10 2015.

[26] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. *Line: Large-Scale Information Network Embedding*, 03 2015.

[27] Audrey Yazdanparast, Timothy Havens, and Mohsen Jamalabdollahi. Soft overlapping community detection in large-scale networks via fast fuzzy modularity maximization. *IEEE Transactions on Fuzzy Systems*, PP, 03 2020.

[28] Mohammed Zaki and Wagner Meira Jr. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. 05 2014.

[29] Shihua Zhang, Rui-Sheng Wang, and Xiang Zhang. Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A: Statistical Mechanics and its Applications*, 374:483–490, 01 2007.