УНИВЕРЗИТЕТ У БЕОГРАДУ
МАТЕМАТИЧКИ ФАКУЛТЕТ

Душан Ж. Џамић

НОВЕ МЕТОДЕ КЛАСТЕРОВАЊА НА КОМПЛЕКСНИМ МРЕЖАМА

докторска дисертација

Београд, 2021.

UNIVERSITY OF BELGRADE
FACULTY OF MATHEMATICS

Dušan Ž. Džamić

NOVEL METHODS FOR CLUSTERING IN COMPLEX NETWORKS

Doctoral Dissertation

Belgrade, 2021.

Ментор:

др Мирослав Марић
ванредни професор, Математички факултет Универзитета у Београду

Чланови комисије:

др Гордана Павловић-Лажетић
редовни професор, Математички факултет Универзитета у Београду

др Зорица Станимировић
редовни професор, Математички факултет Универзитета у Београду

др Мирослав Марић
ванредни професор, Математички факултет Универзитета у Београду

др Нина Радојичић Матић
доцент, Математички факултет Универзитета у Београду

др Ненад Младеновић
редовни професор, Khalifa University, Абу Даби, Уједињени Арапски Емирати
научни саветник (у пензији), Математички институт САНУ, Београд

Датум одбране: _____

Посвећено породици

Захвалница

У изради докторске дисертације имао сам велику подршку породице, пријатеља и колега. Овом приликом желим да им се захвалим.

Пре свега, неизмерно се захваљујем ментору професору Мирославу Марићу на указаном поверењу, несебичном ангажовању и подршци током целокупних студија на Математичком факултету, као и на стручним саветима и активном учествовању у свим фазама израде ове дисертације.

Захваљујем се члановима комисије, уваженим професорима Гордани Павловић - Лажетић, Зорици Станимировић, Ненаду Младеновићу и доценту Нини Радојичић Матић, који су детаљним читањем овог рукописа и својим сугестијама значајно допринели коначном обликовању овог рада. Посебно се захваљујем професорки Зорици Станимировић која је поставила чврсте темеље за мој даљи научно-истраживачки рад у области математичке оптимизације. Увек волим да нагласим да сам уз несебичну помоћ професора Мирослава Марића и Зорице Станимировић направио прве кораке у научно-истраживачком раду, и посебно сам им захвалан на томе. Такође, посебно се захваљујем професору Ненаду Младеновићу који ме је заинтересовао за методу променљивих околина и омогућио сарадњу са врхунским научницима из истраживачких области ове тезе.

Захваљујем се свим колегама са Катедре за математику на Факултету организационих наука, колегама из групе за математичку оптимизацију на Математичком факултету и колегама са пројекта „Математички модели и методе оптимизације великих система" који су ми директно или индиректно помогли у процесу академског усавршавања.

На својим првим корацима у математици, у основној школи „Николај Велимировић" у Александровцу, посебно желим да се захвалим изузетном учитељу Мићи Карајовићу и учитељици Надици Карајовић. Захваљујем се и професорки математике Стојки Симчевић, која је кроз моје средњошколско образовање у школи „Свети Трифун" у Александровцу, утврдила и подржала жељу да наставим образовање на Математичком факултету.

Како кроз читав живот, тако и током писања ове дисертације највећу подршку пружала ми је породица којој посвећујем овај докторат. Захваљујем се оцу Живораду, мајци Зорици и брату Горану на бескрајној посвећености, позитивној енергији, личним одрицањима и поверењу које су имали у мене. Највећу захвалност дугујем супрузи Андријани, која ме свих ових година употпуњује неизмерном љубављу и разумевањем и која ми је умногоме помогла да завршим дисертацију. Овај докторат посвећујем и прерано преминулом деди Радојици кога нисам упознао и преминулом деди Петру који је имао важну улогу у мом животу и одрастању.

Наслов дисертације: Нове методе кластеровања на комплексним мрежама

Резиме: Теорија комплексних мрежа показала се као изузетно значајна у проучавању карактеристика и структуре разноврсних комплексних система. У последње две деценије велики број истраживања је усмерен ка развијању метода кластеровања на комплексним мрежама. Центар за дискретну математику и теоријско рачунарство (енгл. *Center for Discrete Mathematics and Theoretical Computer Science* – DIMACS), који представља конзорцијум престижних академских институција (Универзитет Рутгерс, Принстон, Колумбија) и истраживачких лабораторија (Microsoft, IBM, AT&T, NEC), сврстао је 2012. године проблем кластеровања на мрежи на листу најважнијих проблема и изазова у рачунарству за које је неопходно развити и имплементирати ефикасне алгоритме. Како се кластеровање на мрежи може применити у разноврсним контекстима за постизање различитих циљева, не постоји опште прихваћена дефиниција кластера. Из тог разлога, приликом креирања метода користе се различити приступи. Приступ који је привукао највећу пажњу истраживача подразумева дефинисање мере за одређивање квалитета партиције и конструисање метода за проналажење партиције која има максималну вредност дефинисане мере квалитета. Оваквим приступом проблем кластеровања се формулише као проблем комбинаторне оптимизације, а за решавање могу се користити различите методе математичке оптимизације. Од дефинисаних мера квалитета, најчешће се користи модуларност.

Кластеровање максимизацијом модуларности, односно проналажење партиције са максималном вредношћу модуларности, представља NP-тежак проблем, тако да се за решавање развијају хеуристичке методе. У оквиру дисертације предложена је нова метода за максимизацију модуларности заснована на методи променљивих околина. У циљу ефикасне примене на комплексним мрежама великих димензија, развијен је механизам за декомпозицију проблема на мање потпроблеме и побољшан механизам за превазилажење локалних максимума модуларности коришћењем критеријума за повремено прихватање лошијег решења у односу на текуће решење. За тестирање предложене методе коришћене су DIMACS инстанце. Добијени резултати су упоређени са најбољим резултатима презентованим у литератури за разматрани проблем, који су добијени двема методама развијеним у оквиру DIMACS позива 2012. године. Осим тога, добијени резултати су упоређени и са резултатима шест метода

развијених након 2012. године које су се издвојиле у литератури. Компаративна анализа резултата показује да предложена метода надмашује постојеће методе за максимизацију модуларности и поправља најбоља позната решења на 9 од 13 тешких DIMACS инстанци.

Кластеровање максимизацијом модуларности није погодно за откривање малих кластера у мрежама великих димензија, чак и када су очигледни. Из тог разлога, у оквиру дисертације предложена је нова функција за мерење квалитета партиције чијом се максимизацијом могу идентификовати кластери. Кроз три тврђења показано је да нова мера, названа E-функција, превазилази недостатке који карактеришу модуларност и има потенцијал за идентификовање кластера у мрежи. За потребе детаљног тестирања предложене E-функције и поређења са функцијом модуларности, развијена је генеричка метода променљивих околина за оптимизацију било које реалне функције за мерење квалитета партиције. Рачунски експерименти спроведени су на генерисаним и реалним инстанцама из литературе за које је исправна подела на кластере позната. Резултати показују да се максимизацијом E-функције могу идентификовати очекивани кластери како на генерисаним, тако и на реалним инстанцама.

Кључне речи: комплексне мреже, кластеровање, комбинаторна оптимизација, метода променљивих околина, модуларност, e-функција

Научна област: Рачунарство

Ужа научна област: Математичка оптимизација

Dissertation title: Novel methods for clustering in complex networks

Abstract: The theory of complex networks has proven to be very important in the study of the characteristics and structure of various complex systems. In the last two decades, a large number of researches have been directed towards the development of methods for clustering in complex networks. In 2012, Center for Discrete Mathematics and Theoretical Computer Science (DIMACS), which is a well-known consortium of prestigious academic institutions (Rutgers University, Princeton, Colombia) and research laboratories (Microsoft, IBM, AT & T, NEC), included the problem of clustering in complex networks on the list of the most important problems and challenges in computer science. Clustering in complex networks can be applied in a variety of contexts to achieve different goals, and therefore, there is no generally accepted definition of a cluster. For this reason, different approaches are used in developing clustering methods. An approach that has attracted the most attention of researchers involves two subproblems: defining a function to determine the quality of a partition and constructing methods to find a partition that has the maximum value of the defined quality function. In this approach, the problem of clustering is formulated as the problem of combinatorial optimization and various methods of mathematical optimization can be used to solve it. One of the most commonly used quality function is the modularity.

Clustering by modularity maximization, i.e., finding a partition with the maximum value of modularity, is NP-hard problem. Thus, only heuristic algorithms are suitable of processing large datasets. In this dissertation, a novel method for modularity maximization based on the variable neighborhood search heuristic is proposed. For the purpose of efficient application in large-scale complex networks, a procedure for decomposition of the problem into smaller subproblems is developed. In addition, a mechanism for overcoming local maxima of modularity is improved using criteria for occasional acceptance of solution which is worse than the current one. DIMACS instances are used to test the proposed method, and the obtained results are compared with the best ones presented in the literature, obtained by two methods developed in DIMACS implementation challenge in 2012. In addition, the obtained results are compared with the results of six methods developed after 2012, from the literature. A comparative analysis of the results shows that the proposed method outperforms the existing methods for modularity maximization and improves the best known solutions on 9 out of 13 hard instances.

Clustering by modularity maximization is not suitable for detecting small clusters in large networks. For this reason, a new function for measuring the quality of a partition has been proposed in the dissertation. Through three theorems, it is shown that the new measure, called E-function, has the potential to identify clusters in the network and overcome limits of modularity. For testing the proposed E-function and comparison with the modularity function, a generic variable neighborhood method is developed to optimize the considered quality function. Computational experiments are performed on generated and real instances from the literature for which the correct division into clusters is known. The results show that the expected clusters can be identified, both on artificial and real instances, by maximizing the E-function.

Keywords: complex network, clustering, combinatorial optimization, variable neighborhood search, modularity, exponential quality function

Research area: Computer science

Research sub-area: Mathematical optimization

Садржај

Глава 1

Увод

Појам *систем* има широку примену у савременој науци за представљање скупа елемената (реалних или апстрактних), који међусобно интерагују и функционишу као целина ради остварења заједничког циља. Системи који се састоје из скупа елемената између којих постоје изразито јаке нелинеарне интеракције називају се *комплексним системима*. Комплексност ових система није нужно резултат великог броја елемената и интеракција између њих. Управо нелинеарне интеракције између елемената система стварају највеће разлике између комплексних и једноставних система са линеарним интеракцијама. Нелинеарне интеракције између елемената система дефинишу његову унутрашњу структуру, понашање и модел промене, чинећи га тежим за разумевање и управљање [1]. Услед поремећаја механизма негативне повратне спреге, једноставни системи се развијају глатко и континуирано ка једном стању равнотеже. Супротно томе, код комплексних система елемент А утиче на елемент Б, али и елемент Б повратно утиче на елемент А. При томе, промене у једном елементу изазване дејством другог елемента мењају начин његовог деловања на други, и обратно, а заједно на систем у целини. Услед локалних интеракција, у комплексним системима долази до формирања образаца понашања и структурног уређења без планирања и централизоване контроле. Ова појава назива се *самоорганизовано комплексно понашање*.

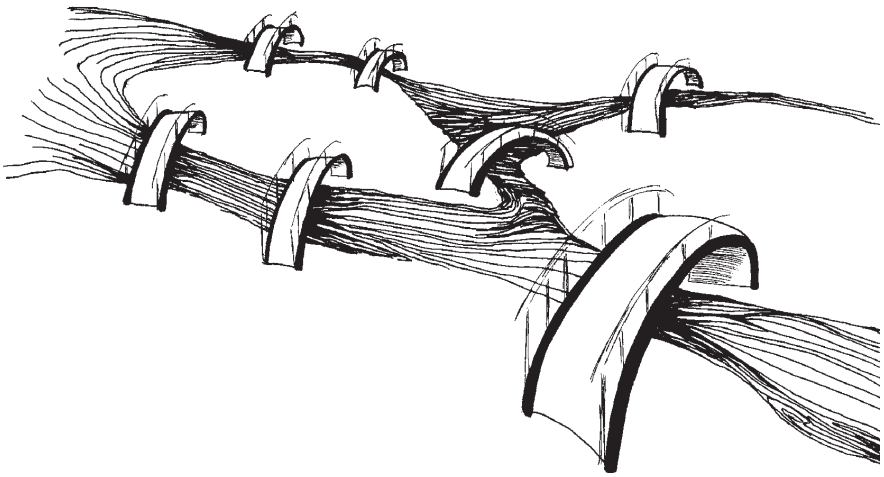
Комплексни системи налазе се свуда око нас, нпр. инфраструктуре као што су енергетска мрежа, транспортни и комуникациони системи, друштвене и економске организације, екосистеми, жива ћелија, људски мозак итд. Уобичајена подела комплексних система врши се према градивном елементу, али без обзира на природу градивног елемента, могу постојати слични принципи

самоорганизације. Штавише, одређене појаве се могу испољити као *универзалне карактеристике* на макроскопском нивоу које не зависе од детаља на микроскопском нивоу [2]. Подела комплексних система се такође може извршити користећи универзалне карактеристике које се описују принципом организације и параметрима.

Због сложених зависности између стања елемената, а често и великог броја елемената и интеракција између њих, за проучавање комплексних система и њихових особина користе се рачунари високих перформанси. Рачунари омогућавају решавање сложених математичких модела који описују комплексне системе и извршавање различитих симулација у којима се посматра велики број елемената и испитује утицај различитих параметара. Занемаривањем специфичности градивних елемената сваки комплексни систем може се представити *комплексном мрежом* и проучавати различитим математичким и рачунарским методама. Наиме, структуру комплексног система осликава комплексна мрежа у којој су елементи система представљени чворовима, док су интеракције између елемената представљене као гране између одговарајућих чворова. У односу на једноставне или случајно генерисане мреже, комплексне мреже испољавају значајну нехомогеност при дистрибуцији грана. Карактеристично је да се јавља висока концентрација грана у оквиру појединих група чворова и истовремено ниска концентрација грана између чворова у различитим групама. Овакве групе чворова се називају *кластери* (модули, заједнице) и често имају заједничке особине и улоге у комплексном систему. Проблем проналажења оваквих група у комплексним мрежама назива се *проблем кластеровања на комплексним мрежама* и представља предмет истраживања ове дисертације. Развијање метода за решавање овог проблема и њихова примена су од великог значаја за разумевање динамике и еволуције комплексних система. Осим тога, могу омогућити бољу визуализацију и пружити неопходне информације о појединачним чворовима и њиховим улогама у мрежи. На пример, поједини чворови у кластеру могу имати улогу у повезивању кластера са остатком мреже, док други чворови могу имати улогу контролисања и стабилизације кластера. Математичку основу за проучавање комплексних мрежа чини *теорија графова*, док за решавање самог проблема кластеровања важну улогу имају методе *математичке оптимизације* и *рачунарске интелигенције*.

1.1 Теорија графова

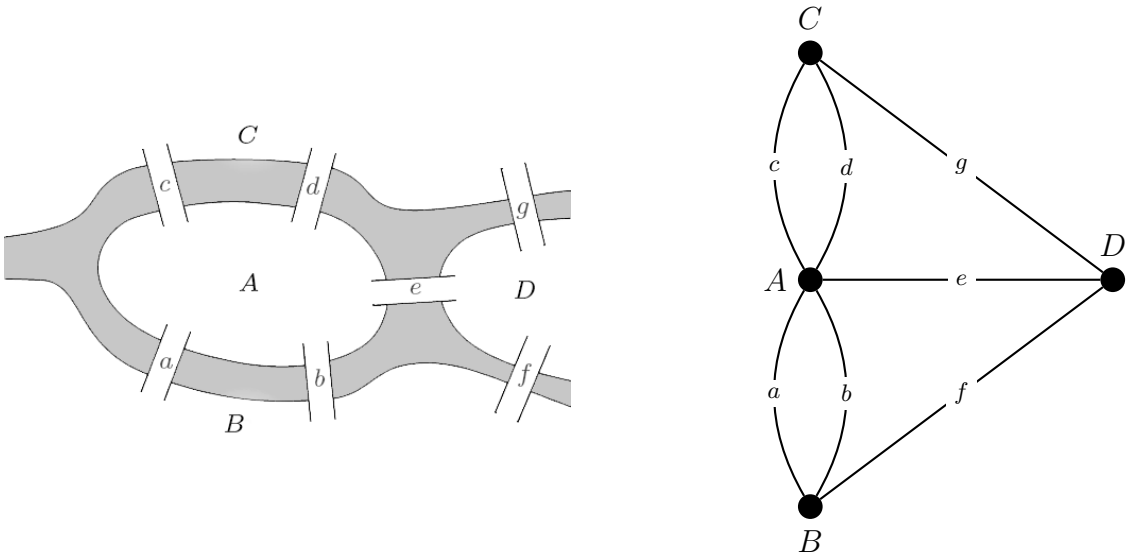
У 18. веку, око 1735. године, појавио се тзв. проблем Кенигсбершких мостова, у руском (тада пруском) граду Калињинграду, односно тадашњем Кенигсбергу. Град је раздвојен реком Прегел (сада Прегоља) и при проласку кроз град на средини реке формирана су два мања острва (Кант и Октобар), а после проласка кроз град река се раздваја на два дела, на Стару Прегољу и Нову Прегољу. Та острва и одговарајуће обале били су спојени са 7 мостова као што је приказано на слици 1.1.



Слика 1.1: Кенигсбершки мостови, 1734. година

Питање које се постављало тада јесте да ли се свих 7 мостова може обићи у једној шетњи, тако да се сваки мост пређе само једном. Овај проблем је привукао пажњу швајцарског математичара Леонарда Ојлера када се налазио на дужности главног председавајућег на Катедри за математику Санкт Петербуршке Академије наука. Августа 1735. године, Ојлер је овај проблем систематски и математички прво изложио Санкт Петербуршкој Академији, да би га затим решио 1736. године, доказујући да такав пут није могуће реализовати, а такође је у истом раду [3] дао и опште решење проблема за произвољан број „копна" и произвољан број „мостова".

Приликом решавања проблема, Ојлер је направио скицу која је одговарала тадашњој конфигурацији мостова и реке у Кенигсбергу која је приказана на слици 1.2.



Слика 1.2: Скица Кенигсберга са мостовима и придружени граф

Ојлер је са A, B, C, D означио копна, а са a, b, c, d, e, f мостове који их спајају. Мостове је разматрао и као одговарајуће прелазе са једног копна на друго, тако да је прелаз са копна А на копно В означио са AB, а ако би требало после преласка са копна А на копно В прећи на копно D, цео пут био би означен са ABD и представљао би прелазак 2 моста и присуство на 3 копна у том случају. На основу тога, Ојлер је закључио да би прелазак преко свих 7 Кенигсбершких мостова у овом проблему захтевао, према уведеним ознакама, 8 великих слова за копна која се у том случају узимају у обзир. У свом раду Ојлер даље развија доказ проблема, описујући ситуацију са појединачним копнима: на пример, ако би се мост a прешао једанпут, копно би према томе било на почетку пута или на крају, односно појавило би се само једанпут. У случају да се мостови a, b, c и d прелазе једанпут, A би се појавило тачно два пута, независно од тога да ли је на почетку или на крају пута. Тако да ако уочимо 5 мостова који воде ка копну A, на основу претходног долазимо до закључка да ће се копно A појавити тачно 3 пута за ту путању (при чему ће се сваки од мостова прелазити тачно једанпут).

Ојлер је уопштио ово тврђење, тако да гласи: уколико је број мостова било који непаран број, и ако се тај број увећа за 1, тада је број појављивања за A једнак половини датог броја. Разматрајући одговарајуће ситуације за преостале случајеве, копна B, C и D морају да се појаве 2 пута, с обзиром на то да према сваком од њих воде по 3 моста. Према томе, за 7 Кенигсбершких

мостова, имамо следеће: 3 појављивања за A и по 2 појављивања за B, C, D, што у збиру даје број 9. Међутим, на почетку свог рада, Ојлер је показао да за 7 мостова мора да постоји само 8 појављивања копна, тј. 8 великих слова, што је контрадикција са чињеницом да их има 9. Према томе, није могуће прећи свих 7 Кенигсбершких мостова тачно једанпут. Овде треба приметити да је начин на који је Ојлер решио проблем, представљајући га апстрактно, помоћу линија и слова, био потпуно нов начин размишљања у то време, тако да се Ојлеров рад сматра почетком модерне теорије графова.

Све до четрдесетих година 19. века, теорија графова је углавном била скуп неповезаних, веома интересантних проблема, а мање целовита математичка теорија. Објављивање Конигове монографије 1936. године [4] покренуло је заснивање теорије графова као самосталне математичке дисциплине, а између осталог, те године је појам *граф* ушао у општу употребу [5]. Касније, након Другог светског рата, општи развој математике и велики захтеви у применама подстакли су развој теорије графова. Више детаља о теорији графова и њеном развоју може се наћи у [6].

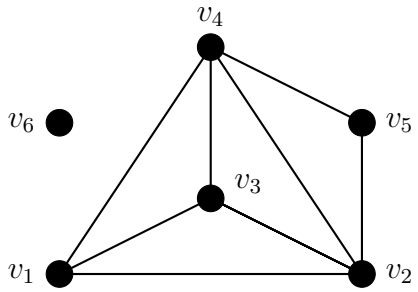
1.1.1 Појам графа

Сваки граф се састоји од *скупа чворова* (енгл. *vertex set*) којим се представљају неки објекти и *скупа грана* (енгл. *edge set*) којим се представљају односи између тих објеката. У зависности од тога да ли је скуп чворова коначан или бесконачан, графови се могу поделити на коначне и бесконачне. У наставку ће бити разматрани само коначни графови који имају примену у анализи и изучавању реалних комплексних система. Са друге стране, у зависности од начина дефинисања скупа грана, постоје различите врсте графова чије су формалне дефиниције дате у наставку.

Граф $G = (V, E)$ се састоји од коначног непразног скупа чворова V и скупа грана $E \subseteq \{\{v_i, v_j\} : v_i, v_j \in V, v_i \neq v_j\}$. Гране овако дефинисаног графа омогућавају представљање искључиво неоријентисаних (неусмерених) односа између елемената из скупа V јер се представљају преко неуређених парова чворова. Такође, није могуће присуство гране која спаја чвор са самим собом (петља). Због тога се овакав граф у литератури често назива *прост граф* или *неоријентисан граф без петљи*. Дефиниција простог графа се врло једноставно може проширити тако да је $E \subseteq V \cup \{\{v_i, v_j\} : v_i, v_j \in V, v_i \neq v_j\}$ чиме се омогућава присуство гране облика $e = v_i$ (где је $v_i \in V$) која чвор v_i спаја са

самим собом. Такав граф се назива *неоријентисан граф са петљама*. У теорији графова и теорији комплексних мрежа најчешће се разматрају графови без петљи.

Број чворова графа G, у ознаци $|V|$ назива се *ред графа*, док се број грана у графу G, одређен као $|E|$ назива величина графа. Код неоријентисаног графа два чвора $v_i, v_j \in V$ су *повезана* граном ако је $\{v_i, v_j\} \in E$. У том случају чворове v_i, v_j називамо *крајњим тачкама* гране $\{v_i, v_j\}$. За грану $e = \{v_i, v_j\} \in E$ кажемо да је *инцидентна* чворовима v_i и v_j. Два чвора v_i, v_j су *суседни чворови* ако представљају крајње тачке исте гране, док су две гране суседне ако су инцидентне истом чвору. Неоријентисан граф се може представити цртежом тако што чворове графа представимо тачкама у равни, а затим непрекидним линијама повежемо суседне чворове. Цртеж графа G_N са скупом чворова $\{v_1, v_2, v_3, v_4, v_5, v_6\}$ и скупом грана $\{\{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_4\}, \{v_2, v_3\}, \{v_2, v_4\}, \{v_2, v_5\}, \{v_3, v_4\}, \{v_4, v_5\}\}$ приказан је на слици 1.3.



Слика 1.3: Неоријентисани граф G_N

Степен и околина чвора

Степен чвора $v_i \in V$ у ознаци $k(v_i)$ представља број грана које су инцидентне са њим. Чвор са степеном нула назива се *изолован чвор*. За граф у коме сви чворови имају степен k каже се да је *k-регуларан*. *Околина чвора* v_i у ознаци $N(v_i)$ представља скуп свих његових суседа, тј. $N(v_i) = \{v_j : \{v_i, v_j\} \in E\}$. У неоријентисаном графу без петљи важи $k(v_i) = |N(v_i)|$. На графу G_N, приказаном на слици 1.3, околина чвора v_3 је $N(v_3) = \{v_1, v_2, v_4\}$ и његов степен $k(v_3) = 3$, док је чвор v_6 изолован чвор.

Теорема 1.1. У неоријентисаном графу $G = (V, E)$ збир степена свих чворова једнак је двоструком броју грана, тј. важи:

$$\sum_{v \in V} k(v) = 2|E|.$$

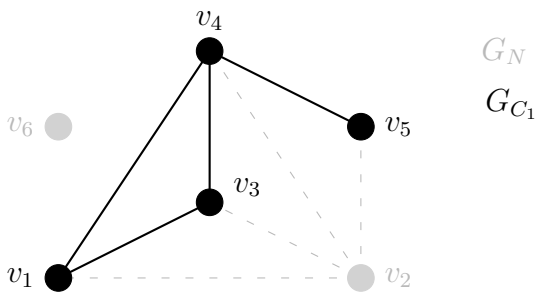
Доказ. Свака грана у неоријентисаном графу $G = (V, E)$ поседује две крајње тачке, а самим тим увећава степен сваке од њих за један. То значи да свака грана укупну суму степена чворова увећава за 2 и она мора да буде једнака двоструком броју грана графа тј. $2|E|$. ■

Теорема 1.2. У неоријентисаном графу $G = (V, E)$ број чворова непарног степена је паран.

Доказ. Скуп чворова V графа G можемо поделити на два скупа: скуп чворова са парним степеном V_0 и скуп чворова са непарним степеном V_1 тако да је $V = V_0 \cup V_1$ и $V_0 \cap V_1 = \emptyset$. Са једне стране, имамо да је укупан степен чворова из V_0 паран број као збир парних бројева. Са друге стране, на основу теореме 1.1, имамо да је укупан степен свих чворова из $V = V_0 \cup V_1$ такође паран број. На основу тога следи да укупан степен чворова из V_1 мора такође да буде паран број. Како су то чворови непарног степена, њихов укупан степен ће бити паран јер је њихов број паран. ■

Индуковани подграф

Граф $G' = (V', E')$ је *подграф* графа $G = (V, E)$ ако је $V' \subseteq V$ и $E' \subseteq E$. У овом случају G је *надграф* од G'. *Индуковани подграф* скупом чворова $C \subseteq V$, у ознаци $G_C = (C, E_C)$ је подграф графа $G = (V, E)$ који се добија издвајањем подскупа чворова $C \subseteq V$ и свих грана између њих тј. $E_C = \{\{v_i, v_j\} \in E : v_i \in C, v_j \in C\}$. Појам *индуковани подграф* биће од изузетне важности при разматрању проблема кластеровања на мрежама у наредним поглављима. У случају графа G_N, приказаног на слици 1.3, индуковани подграф скупом чворова $C_1 = \{v_1, v_3, v_4, v_5\}$ је граф G_{C_1} са скупом чворова C_1 и скупом грана $\{\{v_1, v_3\}, \{v_1, v_4\}, \{v_3, v_4\}, \{v_4, v_5\}\}$. Цртеж подграфа G_{C_1} у оквиру графа G_N приказан је на слици 1.4.



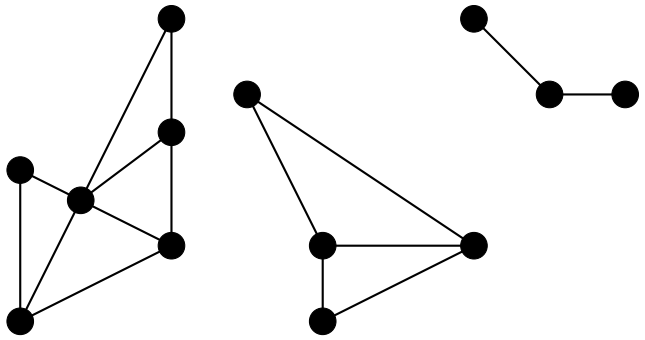
Слика 1.4: Индуковани подграф G_{C_1} издвојен из графа G_N

Компоненте повезаности

Пут дужине k од чвора v_{i_0} до чвора v_{i_k} у графу $G = (V, E)$ представља низ облика $v_{i_0}, e_{j_1}, v_{i_1}, e_{j_2}, v_{i_2}, e_{j_3}, v_{i_3} \ldots, v_{i_{k-1}}, e_{j_k}, v_{i_k}$ где су $v_{i_0}, v_{i_1}, \ldots v_{i_k} \in V$, $e_{j_1}, e_{j_2}, \ldots e_{j_k} \in E$ и $e_{j_t} = \{v_{i_{t-1}}, v_{i_t}\}$. Кажемо да пут пролази кроз чворове $v_{i_0}, v_{i_1}, \ldots, v_{i_{k-1}}, v_{i_k}$ што је уједно и његов краћи запис.

Прост пут од v_{i_0} до v_{i_k} је пут у коме се сваки од чворова $v_{i_1}, v_{i_2} \ldots v_{i_{k-1}}$ јавља тачно једном. Пут који пролази кроз све чворове графа тачно једном назива се *Хамилтонов пут*. Пут који пролази кроз све гране графа тачно једном назива се *Ојлеров пут*. *Затворен пут* је пут у коме је $v_{i_0} = v_{i_k}$, тј. почиње и завршава се у истом чвору. Затворен прост пут се још назива *контура* или *циклус*. Затворен Хамилтонов пут се назива *Хамилтонова контура* и слично затворен Ојлеров пут се назива *Ојлерова контура*.

Два чвора v_i и v_j су *повезана* ако постоји пут од чвора v_i до чвора v_j. Уз то се подразумева да је сваки чвор повезан са самим собом. Граф $G = (V, E)$ је *повезан* ако су свака два чвора из V повезана. У супротном, граф је *неповезан* и у њему се могу издвојити компоненте повезаности.



Слика 1.5: Пример графа који садржи три компоненте повезаности

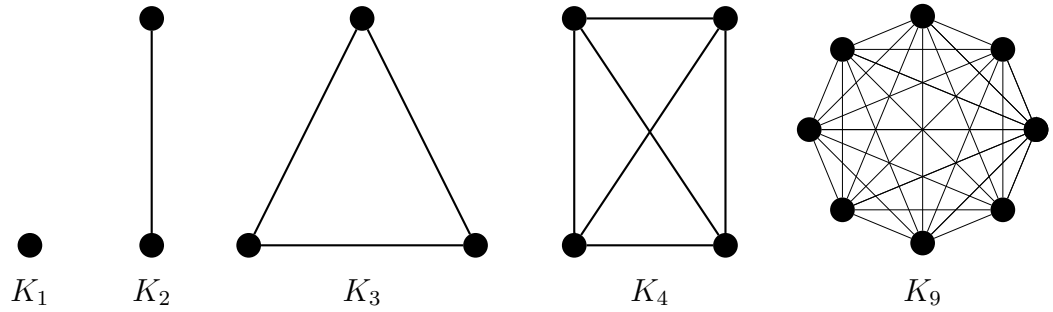
Компонента повезаности графа G је максималан повезан подграф графа G, тј. G' је компонента повезаности графа G ако је G' повезан подграф од G и не постоји други повезан подграф графа G различит од G', који је надграф од G'. Пример графа који има три компоненте повезаности приказан је на слици 1.5.

Теорема 1.3. Сваки затворен пут непарне дужине садржи контуру непарне дужине.

Доказ. Доказаћемо тврђење индукцијом по дужини l затвореног пута. За $l = 1$ затворен пут представља петљу што је уједно контура непарне дужине. Претпоставимо да сваки затворен пут чија је дужина непарна и не већа од $2k - 1$ садржи контуру непарне дужине. Посматрајмо затворен пут $v_{i_0}, v_{i_1}, \ldots v_{2k+1}$, $v_{i_0} = v_{i_k}$ дужине $2k+1$. Ако се сваки од чворова појављује тачно једном, овај пут је по дефиницији истовремено и контура непарне дужине. У супротном, ако се неки чвор понавља, нека је $v_{i_a} = v_{i_b}$, $0 \leq a < b \leq 2k + 1$. Тада се затворени пут који разматрамо може поделити на два затворена пута: $v_{i_0}, \ldots, v_{i_a} = v_{i_b}, \ldots, v_{2k+1}$ и $v_{i_a}, \ldots, v_{i_b}$. Оба пута су затворена, а један од њих мора да буде непарне дужине, не веће од $2k - 1$ и на основу индуктивне хипотезе садржи контуру непарне дужине. ■

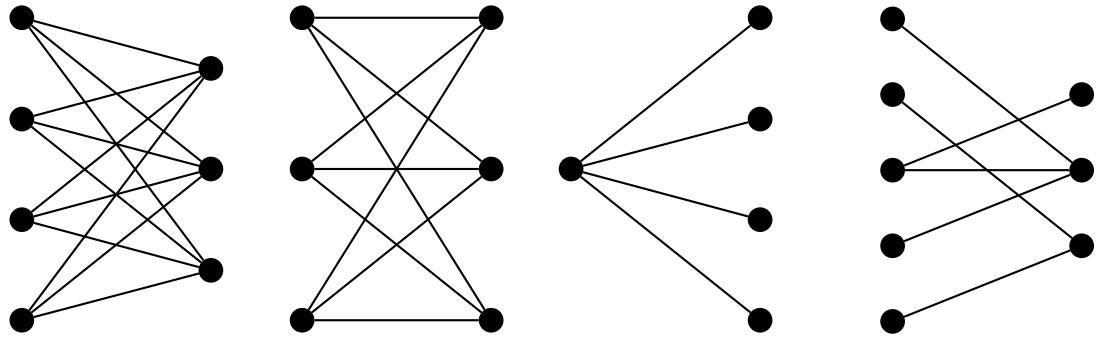
1.1.2 Класе графова

Граф са n чворова у коме постоји грана између било која два различита чвора назива се *комплетан граф* (*клика*) и означава са K_n. Графови K_1, K_2, K_3, K_4 и K_9 приказани су на слици 1.6. Комплемент комплетног графа K_n назива се *празан граф* и означава са N_n.



Слика 1.6: Комплетни графови K_1, K_2, K_3, K_4 и K_9

Комплетан бипартитни граф у ознаци $K_{a,b}$ представља граф у коме је скуп чворова V унија два дисјунктна непразна подскупа V_1 и V_2, са a и b чворова респективно, а скуп грана садржи гране између свих парова чворова из различитих подскупова тј. $E = \{\{v_i, v_j\} : v_i \in V_1, v_j \in V_2\}$. Ако је $a = b$, тада сви чворови имају степен a, односно граф $K_{a,a}$ је a-регуларан. За $a = 1$ граф $K_{1,b}$ се назива *звезда граф*. Било који подграф комплетног бипартитног графа назива се *бипартитни граф*. Графови $K_{4,3}, K_{3,3}, K_{1,4}$, као и један бипартитни граф, приказани су на слици 1.7. Приметимо да је $K_{1,1} = K_2$.



Слика 1.7: Комплетни бипартитни графови $K_{4,3}, K_{3,3}, K_{1,4}$ и један бипартитни граф

Теорема 1.4. Граф је бипартитни ако и само ако не садржи контуру непарне дужине.

Доказ. ($\Rightarrow$): Нека је $G = (V, E)$ бипартитни граф. Доказаћемо да не садржи контуру непарне дужине. Како је G бипартитни, на основу дефиниције, његов скуп чворова V се може поделити на два скупа V_1 и V_2 тако да је $V = V_1 \cup V_2,\ V_1 \cap V_2 = \emptyset$, а све гране $e \in E$ су облика $e = \{v_i, v_j\},\ v_i \in V_1, v_j \in V_2$. Претпоставимо да у графу G постоји контура $v_{i_1}, v_{i_2}, \ldots, v_{i_l}, v_{i_1}$ непарне дужине l. Без губитка општости, претпоставимо да је $v_{i_1} \in V_1$. Тада имамо да је $v_{i_2} \in V_2, v_{i_3} \in V_1$, односно да је

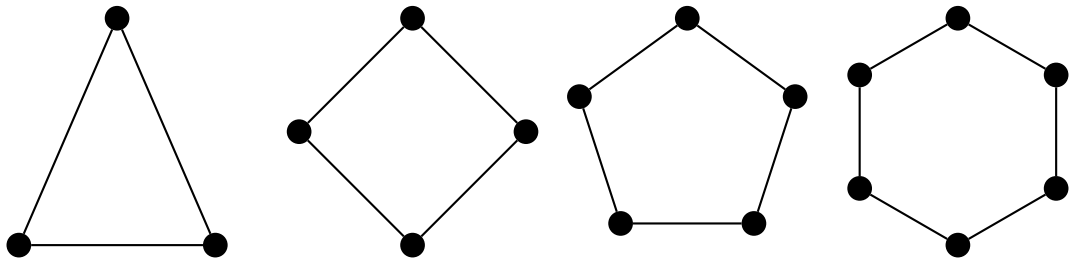
$$v_{i_k} \in \begin{cases} V_1, & \text{ако је } k \text{ непаран број,} \\ V_2, & \text{ако је } k \text{ паран број.} \end{cases} \qquad k = 1, \ldots l$$

Како је l непаран број, имамо да је $v_{i_l} \in V_1$ и да грана $\{v_{i_l}, v_{i_1}\}$, која је део посматране контуре, повезује два чвора из скупа V_1, што је у контрадикцији са претпоставком да је G бипартитни граф. Дакле, претпоставка да у графу G постоји контура непарне дужине је неодржива, чиме је доказано да граф G не садржи контуру непарне дужине.

($\Leftarrow$): Претпоставимо сада да у графу G не постоји контура непарне дужине. Такође претпоставимо, без губитка општости, да је граф $G = (V, E)$ повезан (у супротном можемо посматрати сваку његову компоненту повезаности). Доказаћемо да је G бипартитни граф. Нека је $v \in V$ произвољан чвор. Поделимо скуп чворова V на два скупа, V_1 који садржи чворове за које је најкраћи пут до чвора v непарне дужине и V_2 који садржи чворове за које је најкраћи пут до чвора v парне дужине. Сада имамо да је $v \in V_2$ и $V_1 \cap V_2 = \emptyset$. Претпоставимо да су $v_{i_1} \in V_1$ и $v_{i_2} \in V_1$ суседни чворови. Тада постоји затворени пут $\{v, \ldots, v_{i_1}, v_{i_2} \ldots v\}$ непарне дужине (не обавезно прост). На основу теореме 1.3, у графу G постоји и контура непарне дужине, што је у контрадикцији са полазном претпоставком да у графу G не постоји контура непарне дужине. Дакле, претпоставка да постоје суседни чворови у скупу V_1 је неодржива. Слично, не постоје два суседна чвора у скупу V_2. Коначно имамо да у графу G постоје само гране између чворова из V_1 и V_2, тј. да је граф G бипартитни граф.

■

Циклични граф са $n \geq 3$ чворова у ознаци O_n је повезан граф у коме сваки чвор има степен 2. Дакле, O_n је 2-регуларан. На слици 1.8 приказани су циклични графови O_n за $3 \leq n \leq 6$. Приметимо да је $O_3 = K_3$.



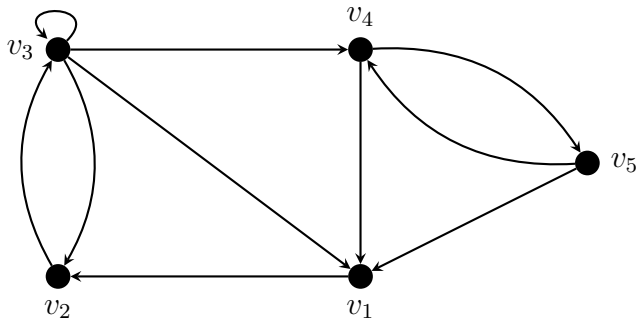
O_3 O_4 O_5 O_6

Слика 1.8: Циклични графови O_3, O_4, O_5, O_6

1.1.3 Оријентисан граф

Неоријентисаним графом можемо представити, на пример, једну мрежу сарадње истраживача у којој сваки истраживач представља чвор и повезан је граном са другим истраживачем ако су заједно објавили научни рад. Међутим, за представљање веб мреже где свака страница представља чвор, а гране представљају хиперлинкове између страница, неоријентисан граф често није погодан. Главни разлог је то што се може десити да страница А садржи хиперлинк до странице Б, док страница Б не садржи хиперлинк до странице А. У том случају потребно је назначити да постоји само грана од А до Б, али не и од Б до А, што се може постићи коришћењем уређених, уместо неуређених парова за гране графа.

Оријентисан граф $G = (V, E)$ се састоји од коначног непразног скупа чворова V и скупа грана $E \subseteq V \times V$. Елементи скупа Е су уређени парови чворова и представљају усмерене гране. За грану $e = (v_i, v_j) \in E$ чвор v_i представља *почетни чвор*, а v_j *завршни чвор*. Кажемо да је чвор v_i суседан ка чвору v_j, односно да је чвор v_j суседан од чвора v_i. На цртежу графа грани се додаје усмерење од v_i ка v_j, најчешће једном стрелицом. Оријентисан граф може да садржи и грану облика (v_i, v_i), тј. петљу. Оријентисан граф G_O са скупом чворова $\{v_1, v_2, v_3, v_4, v_5\}$ и скупом грана $\{(v_1, v_2), (v_2, v_3), (v_3, v_1), (v_3, v_2), (v_3, v_3), (v_3, v_4), (v_4, v_1), (v_4, v_5), (v_5, v_1), (v_5, v_4)\}$ приказан је на слици 1.9.



Слика 1.9: Оријентисан граф G_O

Излазни степен чвора v_i у ознаци $k^+(v_i)$ представља број грана којима је чвор v_i почетни чвор. Са друге стране, *улазни степен* чвора v_i у ознаци $k^-(v_i)$ представља број грана којима је чвор v_i завршни чвор. Чвор чији је улазни степен нула назива се *извор*, док се чвор чији је излазни степен нула назива *понор*.

Теорема 1.5. У оријентисаном графу $G = (V, E)$ важи:

$$\sum_{v \in V} k^-(v) = \sum_{v \in V} k^+(v) = |E|.$$

Доказ. Свака грана $e = (v_i, v_j)$ оријентисаног графа $G = (V, E)$ увећава излазни степен почетног чвора v_i и улазни степен завршног чвора v_j. Дакле, свака грана увећава истовремено суму излазних и суму улазних степена свих чворова за један. ■

Усмерени пут дужине k од чвора v_{i_0} до чвора v_{i_k} у оријентисаном графу $G = (V, E)$ представља низ облика $v_{i_0}, e_{j_1}, v_{i_1}, e_{j_2}, v_{i_2}, e_{j_3}, v_{i_3} \ldots, v_{i_{k-1}}, e_{j_k}, v_{i_k}$ где су $v_{i_0}, v_{i_1}, \ldots v_{i_k} \in V$, $e_{j_1}, e_{j_2}, \ldots e_{j_k} \in E$ и $e_{j_t} = (v_{i_{t-1}}, v_{i_t})$.

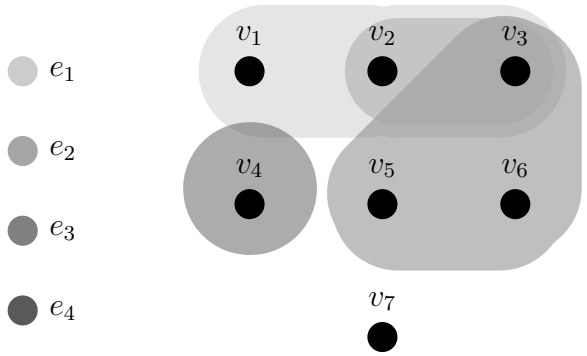
Сваком усмереном графу можемо придружити одговарајући неоријентисан граф тако што све усмерене гране, изузев петљи, посматрамо као неусмерене гране. *Носећи граф* оријентисаног графа $G = (V, E)$ је неоријентисан граф $G^* = (V, E^*)$ где је $E^* = \{\{v_i, v_j\} : (v_i, v_j) \in E, v_i \neq v_j\}$. За оријентисан граф кажемо да је *повезан* ако је његов носећи граф повезан. Често се користи и термин *слабо повезан* јер нам ова особина не гарантује постојање усмереног пута између било која два чвора у графу. Насупрот томе, за оријентисан граф $G = (V, E)$ кажемо да је *јако повезан* ако између било која два различита чвора у графу постоји усмерен пут који их повезује.

Оријентисан граф $G' = (V', E')$ је *подграф* оријентисаног графа $G = (V, E)$ ако је $V' \subseteq V$ и $E' \subseteq E$. У овом случају G је *надграф* од G'.

1.1.4 Уопштења појма графа

Кроз развој теорије графова уведена су различита уопштења графа као што су мултиграф, хиперграф, тежински граф итд. *Мултиграф* $G = (V, E, f)$ састоји се од скупа чворова V, скупа грана E и пресликавања $f : E \to V \times V$ које свакој грани додељује њен почетни и крајњи чвор. За разлику од неоријентисаног и оријентисаног графа, мултиграф може да садржи више различитих грана између два чвора. На пример, проблему Кенигсбершких мостова одговара управо један мултиграф приказан на слици 1.2. *Хиперграф* $G = (V, E)$ се састоји од скупа чворова V и скупа грана $E \subseteq \mathcal{P}(V) \backslash \{\emptyset\}$. Дакле, једна грана хиперграфа може да повезује више чворова. За разлику од графова и

мултиграфова, хиперграфови се тешко представљају цртежом, нарочито ако садрже већи број грана. Хиперграф G_H са чворовима $\{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ и гранама $\{\{v_1, v_2, v_3\}, \{v_2, v_3\}, \{v_3, v_5, v_6\}, \{v_4\}\}$ приказан је на слици 1.10.



Слика 1.10: Хиперграф G_H

Сваки од наведених типова графова може се проширити у *тежински граф*, додавањем тежинске функције $w : E \to R$ која свакој грани графа додељује реалан број који представља тежину грану. Овакви графови се јављају у различитим контекстима тако да тежине грана могу представљати трошкове, дужину, капацитет итд. Приликом решавања проблема најкраћег пута у графу или проблема трговачког путника, тежине грана се морају разматрати јер имају кључну улогу у одређивању решења, док се у проблему кластеровања могу, али и не морају разматрати. Више детаља о свим наведеним терминима, уопштењима појма графа, као и различитим проблемима у теорији графова може се наћи у [6, 7, 8].

1.2 Теорија сложености

Имплементација алгоритама на неком рачунарском моделу троши ресурсе тог модела, првенствено време и меморију. Утрошак времена, односно меморије, мери се у односу на димензију улазних података и назива се *временска*, односно *просторна сложеност алгоритма*. Временска и просторна сложеност алгоритма одређује његову употребљивост за решавање неког проблема, тј. даје могућност да се процени највећа димензија проблема која може бити решена тим алгоритмом са расположивим ресурсима. Временска сложеност алгоритма за решавање проблема димензије n представља се функцијом $t(n)$. Приликом процене користи се анализа најгорег случаја или анализа просечног случаја. Анализа најгорег случаја у пракси омогућава добру процену, док

анализа просечног случаја није увек могућа и подразумева познавање допуштених улазних вредности и вероватноће појављивања. У наставку ће бити подразумевана анализа најгорег случаја – ако није наведено другачије.

1.2.1 Асимптотско време извршавања

Нека је за алгоритам A време извршавања $t(n) = n^3 + 2n^2 + 2n + 30$. Ако је време извршавања елементарне операције на конкретном рачунару једна наносекунда (10^{-9} секунди), тада за улаз димензије $n = 10000$ члан n^3 функције $t(n)$ троши 16.7 минута док сви остали чланови $2n^2 + 2n + 30$ заједно троше свега 0.2 секунде. Како доминантни члан функције $t(n)$ скоро у потпуности одређује време извршавања алгоритма за велику димензију улаза, најчешће се користи асимптотска нотација и одговарајућа функција једноставнијег облика која асимптотски ограничава функцију $t(n)$ одозго.

Асимптотска нотација O

Нека су f и g две функције чији је домен скуп природних бројева, а кодомен скуп реалних бројева тј. $f, g : \mathbb{N} \longrightarrow \mathbb{R}$. Кажемо да функција $g(n)$ *асимптотски ограничава функцију* $f(n)$, у ознаци $f(n) = O(g(n))$, ако постоји реална коначна константа $c > 0$ и цео број $n_0 \in \mathbb{R}$ тако да за све $n \geq n_0$ важи

$$|f(n)| \leq c|g(n)|.$$

У наставку су приказана правила за рачунање са O нотацијом. Више детаља о анализи сложености алгоритама и асимптотској нотацији може се наћи у [8, 9].

Теорема 1.6. Ако су x и y реални бројеви такви да $x \leq y$, тада је $n^x = O(n^y)$.

Доказ. Нека је $n > 1$, тада је $\ln n > 0$ и имамо

$$\begin{aligned} x \leq y &\Leftrightarrow x \ln n \leq y \ln n \\ &\Leftrightarrow \ln n^x \leq \ln n^y \\ &\Leftrightarrow n^x \leq n^y. \end{aligned}$$

Дакле, $n^x \leq n^y$ за $n > 1$ и $x \leq y$, одакле следи да је $n^x = O(n^y)$. ■

Две теореме у наставку показују да је својство припадности $O(g(n))$ затворено у односу на множење скаларом и сабирање.

Теорема 1.7. Ако је $f(n) = O(g(n))$, тада је $c\,f(n) = O(g(n))$.

Доказ. Из $f(n) = O(g(n))$ на основу дефиниције следи да постоји константа $c_1 \in R$ тако да почев од неког $n \geq n_0$ важи

$$|f(n)| \leq c_1\,|g(n)|.$$

Множењем неједнакости са $|c|$ добијамо $|c\,f(n)| \leq c_1|c|\,|g(n)|$. Дакле, постоји реална константа $c_2 = c_1|c|$ таква да за $n \geq n_0$ важи

$$|c\,f(n)| \leq c_2\,|g(n)|,$$

одакле следи да је $c\,f(n) = O(g(n))$. ■

Теорема 1.8. Ако су $f_1(n) = O(g(n))$ и $f_2(n) = O(g(n))$, тада је $(f_1+f_2)(n) = O(g(n))$.

Доказ. Из $f_1(n) = O(g(n))$ на основу дефиниције имамо да постоји константа $c_1 \in R$ тако да почев од неког $n \geq n_1$ важи

$$|f_1(n)| \leq c_1\,|g(n)|.$$

Слично, из $f_2(n) = O(g(n))$ имамо да постоји константа $c_2 \in R$ тако да почев од неког $n \geq n_2$ важи

$$|f_2(n)| \leq c_2\,|g(n)|.$$

Нека је $n_0 = \max n_1, n_2$. Тада за свако $n > n_0$ имамо:

$$\begin{aligned} |f_1(n) + f_2(n)| &\leq |f_1(n)| + |f_2(n)| \\ &\leq c_1\,|g(n)| + c_2\,|g(n)| \\ &= (c_1 + c_2)\,|g(n)|. \end{aligned}$$

Дакле, постоји реална константа $c = c_1 + c_2$ таква да за $n \geq n_0$ важи

$$|f_1(n) + f_2(n)| \leq c\,|g(n)|,$$

одакле следи да је $(f_1 + f_2)(n) = O(g(n))$. ■

Теорема 1.9. Ако су $f_1(n) = O(g_1(n))$ и $f_2(n) = O(g_2(n))$, тада је $(f_1\, f_2)(n) = O((g_1\, g_2)(n))$.

Доказ. Из $f_1(n) = O(g_1(n))$ на основу дефиниције имамо да постоји константа $c_1 \in R$ тако да почев од неког $n \geq n_1$ важи

$$|f_1(n)| \leq c_1\, |g_1(n)|.$$

Слично, из $f_2(n) = O(g_2(n))$ имамо да постоји константа $c_2 \in R$ тако да почев од неког $n \geq n_2$ важи

$$|f_2(n)| \leq c_2\, |g_2(n)|.$$

Нека је $n_0 = \max\{n_1, n_2\}$. Тада за свако $n > n_0$ имамо:

$$\begin{aligned} |f_1(n)\, f_2(n)| &= |f_1(n)|\, |f_2(n)| \\ &\leq c_1\, |g(n)| + c_2\, |g_2(n)| \\ &= (c_1\, c_2)|g_1(n)\, g_2(n)|. \end{aligned}$$

Дакле, постоји реална константа $c = c_1\, c_2$ таква да за $n \geq n_0$ важи

$$|f_1(n)\, f_2(n)| \leq c\, |g_1(n)\, g_2(n)|,$$

одакле следи да је $(f_1\, f_2)(n) = O\left((g_1\, g_2)(n)\right)$. ■

Теорема 1.10. Ако је $p(n) = a_k\, n^k + a_{k-1}\, n^{k-1} + \cdots + a_1\, n + a_0$, тада је $p(n) = O(n^k)$, за $n \geq 1$.

Доказ. Како је

$$\begin{aligned} |p(n)| &= |a_k\, n^k + a_{k-1}\, n^{k-1} + \cdots + a_1\, n + a_0| \\ &\leq |a_k\, n^k| + |a_{k-1}\, n^{k-1}| + \cdots + |a_1\, n| + |a_0| \\ &= |a_k|\, n^k + |a_{k-1}|\, n^{k-1} + \cdots + |a_1|\, n + |a_0| \\ &\leq |a_k|\, n^k + |a_{k-1}|\, n^k + \cdots + |a_1|\, n^k + |a_0|\, n^k \quad (\text{за } n \geq 1) \\ &= (|a_k| + |a_{k-1}| + \cdots + |a_1| + |a_0|)\, n^k, \end{aligned}$$

јасно је да постоји реална константа $c = |a_k| + |a_{k-1}| + \cdots + |a_1| + |a_0|$ таква да за $n \geq 1$ важи

$$|p(n)| \leq c\, |n^k|,$$

одакле следи да је $p(n) = O(n^k)$. ■

Класе сложености алгоритама

За алгоритме константне сложености $O(1)$ карактеристично је да време извршавања не зависи од величине улаза. Пример је алгоритам за рачунање збира првих n природних бројева коришћењем формуле $1 + 2 + 3 + \cdots + n = \frac{n(n-1)}{2}$. Алгоритам се извршава у једном кораку који у овом случају обухвата три операције, а време извршавања је исто за $n = 10$ као и за $n = 10000$.

Алгоритми логаритамске сложености $O(\log n)$ представљају изузетно ефикасне алгоритме, а пример је алгоритам бинарне претраге. Код алгоритама линеарне сложености $O(n)$ време извршавања расте линеарно са порастом димензије проблема. Пример представља алгоритам за одређивање минимума или максимума за дати низ бројева величине n.

Алгоритми полиномијалне сложености $O(n^k)$, где је k реална позитивна константа већа од 1, теоријски спадају у ефикасне алгоритме, међутим у пракси су ефикасни само за релативно мале вредности константе k. На пример, алгоритам сортирања уметањем (енгл. *insertion sort*) је сложености $O(n^2)$, док је алгоритам множења матрица сложености $O(n^3)$.

Алгоритми експоненцијалне сложености $O(a^n)$, где је a реална позитивна константа већа од 1, спадају у неефикасне алгоритме јер време извршавања расте експоненцијално у односу на димензију улаза. Овакви алгоритми су практично неупотребљиви за веће димензије улаза. Алгоритам који испитује све подскупове скупа са n елемената сложености $O(2^n)$ је један представник ове класе.

1.2.2 Класе сложености P и NP

Проблем одлучивања (енгл. *decision problem*) представља проблем чије је решење одговор „да" или „не". Примери два проблема одлучивања у теорији графова су:

- P_1: „Да ли је дати граф G бипартитни?";
- P_2: „Да ли дати граф G садржи Хамилтонову контуру?".

Тест пример или *инстанца* проблема, представља конкретан задатак проблема који је потребно решити. У случају проблема P_1 и P_2 инстанцу представља конкретан граф.

Проблем одлучивања припада класи P ако постоји детерминистички алгоритам полиномијалне временске сложености који га решава. На пример, проблем P_1 припада класи P. Алгоритам који даје решење креће од једног чвора и смешта га у скуп V_1, затим његове суседе смешта у скуп V_2, а затим све њихове суседе поново у скуп V_1 итд. Ако се процес поделе чворова заврши успешно, граф је бипартитни и одговор је „да", док је у случају покушаја смештања једног чвора у оба скупа одговор „не".

Проблем одлучивања припада класи NP ако постоји детерминистички алгоритам полиномијалне временске сложености који за једно потенцијално решење утврђује да ли је оно заиста решење проблема. Алтернативно NP класа се може дефинисати и као скуп проблема одлучивања чије решење може да се добије недетерминистичким алгоритмом у полиномијалном времену.

Примери NP проблема одлучивања су проблеми P_1 и P_2. За проблем P_1 се веома једноставно може утврдити да ли је дата подела скупа чворова V на V_1 и V_2 исправна. У полиномијалном времену се проверава да ли је $V_1 \cap V_2 = \emptyset$ и да ли су све гране облика $e = \{v_1, v_2\}$, где је $v_1 \in V_1$ и $v_2 \in V_2$. Слично, код проблема P_2 за дату контуру може се у полиномијалном времену утврдити да ли пролази кроз све чворове тачно једном.

Једно од отворених питања теоријског рачунарства тиче се односа између ове две класе, тј. да ли је P = NP. Јасно је да је P $\subseteq$ NP јер је сваки детерминистички алгоритам истовремено недетерминистички. Међутим, још увек није познато да ли је NP $\subseteq$ P.

Важну потклasу NP проблема представљају NP-комплетни проблеми који се могу дефинисати коришћењем дефиниције полиномијалне сводљивости. Проблем R је полиномијално сводљив на проблем Q ако се од сваке инстанце I_R проблема R у полиномијалном времену може конструисати инстанца I_Q за проблем Q тако да је за инстанцу I_R одговор „да" ако и само ако је за инстанцу I_Q одговор „да". Проблем одлучивања је NP-комплетан ако припада класи NP и сви други проблеми класе NP се полиномијално своде на њега. Другим речима, класа NP-комплетних проблема садржи међусобно еквивалентне проблеме тако да је приликом доказивања да неки нови проблем R припада овој класи довољно доказати да је један познат NP-комплетан проблем полиномијално сводљив на R. Дакле, проналажење ефикасног алгоритма за решавање једног NP-комплетног проблема омогућило би ефикасно решавање свих NP-комплетних проблема.

Први проблем за који је доказано да је NP-комплетан је проблем исказне задовољивости (енгл. *Boolean Satisfiability Problem* – SAT). Доказ је дао Кук 1971. године [10], а након тога Карп је за још 21 проблем доказао NP-комплетност [11]. Ови резултати поставили су темељ целе теорије и омогућили једноставније доказивање да су многи други проблеми NP-комплетни свођењем на проблем SAT или неки други проблем за који је доказано да је NP-комплетан.

Проблем одлучивања је NP-тежак ако се сваки NP проблем своди на њега или еквивалентно, ако постоји NP-комплетан проблем који је полиномијално сводљив на њега. За проблеме из ове класе важи да су тешки бар колико и NP-комплетни проблеми. Проблем P_2 представља један NP-тежак проблем, а истовремено и NP-комплетан проблем. Ипак, постоје проблеми који су NP-тешки, али не и NP-комплетни. Типичан представник је проблем испитивања заустављања програма (енгл. *halting problem*). Више детаља о теорији сложености може се наћи у [9, 12].

1.3 Математичка оптимизација

За разумевање појава у свету који нас окружује веома су значајни проблеми минимума и максимума одређених функција. Леонард Ојлер, швајцарски математичар 18. века, још 1744. године је говорио да се ништа у свету не дешава, а да нема смисао неког максимума или минимума. Често се наводи оригинална изјава на латинском језику у литератури:

„Cum enim mundi universi fabrica sit perfectissima atque a Creatore sapientissimo absoluta, nihil omnino in mundo contingit, in quo non maximi minimive ratio quaepiam eluceat; quamobrem dubium prorsus est nullum, quin omnes mundi effectus ex causis finalibus ope methodi maximorum & minimorum aeque feliciter determinari queant, atque ex ipsis causis efficientibus."

— L.Euler (1707–1783)

Заиста, светлост се простире тако да растојање између две тачке пређе за најкраће време, балон сапунице тежи да заузме облик који има најмању могућу површину за дату запремину, пчеле изграђују саће тако да дати простор

поделе на једнаке делове уз минималну потрошњу материјала. Са проблемима минимума и максимума се сусрећемо и у свакодневном животу, на пример, приликом транспорта од једног до другог места тражимо пут који ће нам омогућити минимално време путовања, у продавници бирамо ред тако да чекање буде минимално итд.

Проблем *математичке оптимизације* представља проблем у коме се захтева максимизација или минимизација дате реалне функције бирањем вредности променљивих из подскупа домена који је одређен датим ограничењима. Формално, за дату реалну функцију $f : S \to R$ и непразан скуп $X \subseteq S$, потребно је пронаћи минимум функције f на скупу X, тј. решити задатак

$$\min_{x \in X} f(x) \tag{1.1}$$

Сваком проблему математичке оптимизације се може придружити одговарајући проблем одлучивања увођењем границе L за вредност функције циља. У том случају, одговарајући проблем одлучивања гласи: „Да ли постоји $x \in X$ тако да је $f(x) \leq L$?".

Функција f мери квалитет решења x и назива се *функција циља* или *критеријумска функција*. Како се проблем максимизације функције f своди на проблем минимизације функције $-f$, тј. $\max_{x \in X} f(x) = -\min_{x \in X}(-f(x))$, довољно је разматрати само један од њих.

Скуп S представља домен функције циља, док скуп X представља *допустив скуп* решења. Ограничења која одређују скуп X записују се на различите начине у зависности од њихове математичке природе. Тачка $x \in X$ се назива *допустива тачка*, односно *допустиво решење*. Слично, свака тачка $x \in S \backslash X$ је *недопустива*. Допустиво решење x' се још назива и *локални минимум* у некој околини $N(x') \subseteq X$ ако важи $(\forall x \in N(x'))\,(f(x') \leq f(x))$. Допустиво решење x^* у коме се достиже минимум функције f на скупу X, тј. за које важи $(\forall x \in X)\,(f(x^*) \leq f(x))$, назива се *глобални минимум*, а у случају проблема минимизације још и *оптимално решење*. Оптимално решење може бити јединствено, а може се десити и да постоји више различитих решења у којима се достиже иста оптимална вредност функције f. Приликом решавања проблема оптимизације захтева се проналажење само једног оптималног решења – ређе се захтева проналажење свих оптималних решења.

1.3.1 Типови оптимизационих проблема

У зависности од типа функције циља f и функција које дефинишу допустиви скуп X, постоје различите класификације проблема математичке оптимизације. Ако је функција циља конвексна као и допустив скуп X, тада се ради о проблему *конвексне оптимизације*. У супротном, реч је о *неконвексној оптимизацији*. Са друге стране, ако су функција циља и функције које дефинишу допустиви скуп линеарне, кажемо да се ради о проблему *линеарне оптимизације*, а у супротном о проблему *нелинеарне оптимизације*. Са треће стране, ако промењиве које учествују у оптимизацији узимају вредности из коначног или пребројивог скупа, ради се о *проблему дискретне оптимизације*, а уколико узимају вредности из непребројивог скупа, реч је о проблему *континуалне (глобалне) оптимизације*.

Користећи претходне три поделе, оптимизациони проблеми се деле на неколико класа, а затим се развијају различите групе општих метода за њихово решавање у оквиру посебне дисциплине која повезује математику са рачунарским наукама и назива се *математичко програмирање*. Подела на типове проблема математичког програмирања приказана је на слици 1.11.

Слика 1.11: Класификација проблема математичког програмирања

Како је проблем конвексне оптимизације увек проблем непрекидне оптимизације, у зависности од линеарности функције циља и функција које дефи-

нишу допустиви скуп, постоје проблеми *линеарног програмирања* (енгл. *Linear Programming* – LP) и *конвексног нелинеарног програмирања* (енгл. *Convex Nonlinear Programming* – Convex NLP). Са друге стране, проблем неконвексне оптимизације може бити континуалан (а самим тим и нелинеаран - *Nonconvex Nonlinear Programming* – NCNLP) или дискретан. Неконвексни дискретни проблеми оптимизације се могу даље поделити на линеарне и нелинеарне. Код линеарних проблема се понекад разликују две класе у зависности од тога да ли су све променљиве целобројне – *целобројно линеарно програмирање* (енгл. *Integer Linear Programming* – ILP) или су неке целобројне, а неке реалне – *мешовито целобројно линеарно програмирање* (енгл. *Mixed Integer Linear Programming* – MILP). Код нелинеарних проблема (енгл. *Mixed Integer Nonlinear Programming* – MINLP) разматра се да ли релаксација скупа, из кога променљиве узимају вредност, са дискретног на континуални даје конвексни или неконвексни проблем.

За решавање проблема линеарног програмирања Данцинг је 1947. развио симплекс методу [13]. Временска сложеност предложеног алгоритма у најгорем случају је $O(m^{n/2})$, где је n број променљивих, а m број ограничења. Ипак, алгоритам је нашао широку индустријску примену, а користи се и данас јер је сложеност $O(n+m)$ у просечном случају. Тек 1979. године Качијан је доказао да је проблем линеарног програмирања решив у полиномијалном времену [14]. Први полиномијални алгоритам који је имао практични значај развио је Кармаркар 1984. године [15] коришћењем методе унутрашњих тачака (енгл. *interior-point method*). За решавање проблема нелинеарног програмирања развијене су градијентне методе, Њутнова метода, метода казнених фунцкија и друге. Више детаља о овим методама може се пронаћи у [16, 17].

Проблем мешовитог целобројног линеарног програмирања у општем облику је NP-тежак и не постоји ефикасан егзактни алгоритам за његово решавање. За мале инстанце проблема могу се користити методе посредног претраживања, попут гранања и ограничавања (енгл. *Branch and Bound* – BnB), гранања и одсецања (енгл. *Branch and Cut* – BnC). Ове методе претражују само део допустивог скупа и посредно доказују да се оптимално решење не налази у делу који није претражен. Одређивање горњих и доњих граница и њихово међусобно приближавање током претраживања назива се *ограничавање*, а елиминација делова допустивог скупа се назива *одсецање*. Наиме, идеја је да се уместо претраживања експоненцијално великог допустивог ску-

па полазног проблема P, пронађе оптимално решење његове полиномијалне релаксације R (добијене уклањањем неких ограничења). У случају да је то решење допустиво за P, проблем је решен. У супротном, проблему R се додају ограничења (тако да остане полиномијалан) која ће добијено решење учинити недопустивим за нови релаксирани проблем. Додавањем ограничења, део допустивог скупа проблема R који није допустив за P, се искључује из даљег претраживања (одсецање), а остатак дели на више делова (гранање) чија унија садржи сва допустива решења проблема P. Више детаља о методама посредног претраживања може се пронаћи у [18, 19, 20]. За решавање проблема целобројног линеарног програмирања са великим бројем променљивих често се користи и метода гранања и оцењивања (енгл. *Branch and Price* – BnP) која представља хибрид BnB методе и методе генерација колона (енгл. *column generation method*). Више детаља може се наћи у [21].

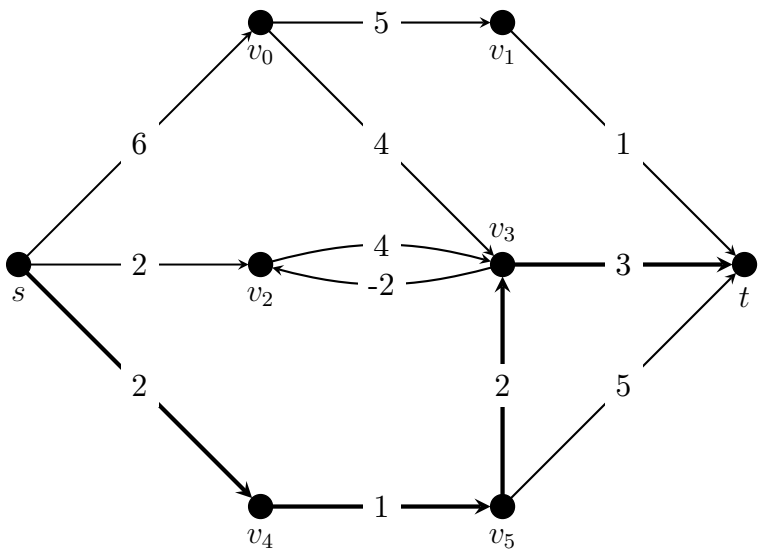
1.3.2 Комбинаторна оптимизација

Посебно важну грану дискретне оптимизације представља *комбинаторна оптимизација*. Код проблема комбинаторне оптимизације простор решења је коначан, а решења се могу изразити коришћењем концепата из комбинаторике попут скупова, комбинација, пермутација или концепата из теорије графова попут чворова, грана, контура, клика итд.

Пример једног проблема комбинаторне оптимизације је проблем одређивања најкраћег пута (енгл. *Shortest Path Problem* – SPP) између два чвора у тежинском графу. Формално, проблем најкраћег пута се може формулисати на следећи начин: Дат је тежински оријентисан граф $G = (V, E)$, почетни чвор $s \in V$ и крајњи чвор $t \in V$. Дужина пута $p = v_0, v_1, v_2, \ldots, v_k$ представља збир тежина грана на том путу, односно $w(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i)$. Из скупа свих путева P_{st} од чвора s до чвора t потребно је одредити пут најмање дужине:

$$\min_{p \in P_{st}} w(p).$$

Граф који представља једну инстанцу проблема најкраћег пута приказан је на слици 1.12. Оптимално решење, односно најкраћи пут од s до t је $P^* = s, v_4, v_5, v_3, t$ дужине 8.



Слика 1.12: Пример једне инстанце проблема најкраћег пута

За проналажење најкраћег пута у тежинском графу са ненегативним коефицијентима, Дајкстра је 1956. развио ефикасан алгоритам полиномијалне сложености $O(|V|^2)$, а који је објављен три године касније [22]. Белмано је 1958. унапредио алгоритам тако да решава инстанце проблема са негативним коефицијентима, али под условом да не постоји контура негативне дужине [23]. Сложеност Белмановог алгоритма је $O(|E||V|)$. Ови алгоритми имају велику практичну примену јер су услови које захтевају најчешће испуњени. Међутим, у општем случају проблем проналажења најкраћег простог пута, односно одговарајући проблем одлучивања је NP-тежак. Такође, велики број других проблема комбинаторне оптимизације, односно њима одговарајући проблеми одлучивања, припадају класи NP-тешких проблема, а развијање метода за њихово решавање представља изазов који привлачи многобројне истраживаче. Пример таквог проблема је проблем трговачког путника (енгл. *Travelling Salesman Problem* – TSP) у коме је дат одређен број градова и трошкови путовања између свака два града, а потребно је одредити руту са најмањим трошковима којом се обилази сваки град тачно једном и враћа у почетни град. У терминима теорије графова проблем се може формулисати као проблем одређивања Хамилтонове контуре најмање тежине у комплетном тежинском графу. Чворови графа представљају градове, гране представљају путеве, а тежине грана представљају трошкове путовања. До данас није пронађен ефикасан алгоритам који одређује оптимално решење за сваку ин-

станцу овог проблема. Генерално, код проблема комбинаторне оптимизације број допустивих решења, иако коначан, изузетно је велик и захтева велику количину ресурса у процесу претраживања. За TSP проблем са n градова постоји $(n-1)!$ потенцијалних рута. Претраживање целог простора (израчунавање трошкова свих рута и одређивање најповољније) је сложености $O(n!)$, и није га могуће спровести у реалном времену већ за $n = 20$.

1.4 Метахеуристичке методе

У првој фази развоја метода за решавање проблема математичке оптимизације истраживачи су се углавном бавили креирањем *егзактних метода*, тј. метода које на основу одговарајућих теоријских разматрања гарантују проналажење оптималног решења у коначном броју корака или конвергенцију низа апроксимација ка оптималном решењу. При креирању оваквих метода увек се полази од прецизно формулисаног математичког модела који одражава природу проблема који се решава. Међутим, проблеми који се јављају у научним и индустријским применама имају комплексну структуру са великим бројем ограничења која се не могу у потпуности математички формализовати, а самим тим није могуће креирати прецизан математички модел. Осим тога, код неких прецизно дефинисаних проблема може се десити да егзактне методе, у случају инстанци великих димензија, не могу да пронађу оптимално решење користећи расположиве временске и меморијске ресурсе.

У циљу ефикасног решавања таквих проблема, истраживачи су средином шездесетих година прошлог века почели да конструишу *хеуристичке методе* (хеуристике) које проналазе довољно добра решења у датим временским и меморијским оквирима. Термин „хеуристика" потиче од старогрчке речи „heuriskein" што значи „пронаћи" или „открити". Како ове методе не користе класичне формализоване математичке поступке засноване на теоријским резултатима, њихова примена не гарантује проналажење оптималног решења разматраног проблема. Са друге стране, хеуристике дају могућност имплементације правила која често имитирају процес људског размишљања. На тај начин добро конципиране хеуристике обезбеђују проналажење решења која су врло блиска оптималном.

Према начину проналаска решења хеуристике се деле на конструктивне и итеративне. Код конструктивних хеуристика формира се решење блиско

оптималном, тако што се користе принципи попут прождрљивости (енгл. *greedy*) или гледања унапред. Конструктивне хеуристике се обично користе за решавање проблема код којих је тешко формирати неко допустиво решење. Итеративне хеуристике формирају читав низ решења, односно у свакој итерацији покушавају да поправе претходно решење. Према принципу који се користи приликом доношења одлука, хеуристике се могу поделити на стохастичке и детерминистичке. Код детерминистичке хеуристике све одлуке су у потпуности детерминисане, а применом више пута на истој инстанци увек се добија исто решење. Код стохастичке хеуристике, барем једна одлука се доноси случајним одабиром. Због тога се приликом сваке примене могу добити различита крајња решења једне инстанце проблема.

Средином осамдесетих година прошлог века почиње интензиван развој општих хеуристика које се могу прилагодити великом броју различитих проблема математичке оптимизације. За ове опште хеуристике данас се користи назив *метахеуристике* који је увео Фред Гловер 1986. године [24] извођењем из две грчке речи „metá" што значи „изнад" или „на вишем нивоу" и „heuriskein" што значи „пронаћи" или „открити". Дакле, метахеуристика представља скуп уопштених, али јасно дефинисаних правила, чијом применом се долази до квалитетних решења разноврсних NP-тешких проблема математичке оптимизације. Све метахеуристике садрже механизме за диверсификацију и интензификацију процеса претраге простора решења. Механизми диверсификације су задужени за ширење претраге кроз простор решења са циљем проналажења делова простора у којима се налазе квалитетна решења. Са друге стране, механизми интензификације су задужени за детаљније истраживање делова простора у којима се очекују квалитетна решења. Баланс између диверсификације и интензификације је кључан за ефикасну претрагу простора решења.

Процес претраживања може бити вођен једним решењем или популацијом која садржи више решења. Од метахеуристика вођених једним решењем (енгл. *single solution based*) посебно се издвајају метода симулираног каљења [25] (енгл. *Simulated Annealing* – SA), табу претрага [26, 27] (енгл. *Tabu Search* – TS), метода променљивих околина [28] (енгл. *Variable Neighborhood Search* – VNS) и насумично похлепна адаптивна претрага [29] (енгл. *Greedy Randomized Adaptive Search Procedure* – GRASP). Већина метахеуристика вођених једним решењем за механизам интензификације користи неки облик локалне

претраге која се састоји из малих померања текућег решења док се не пронађе локални минимум или док се не задовољи задати услов. Разлику између ових метахеуристика прави механизам диверсификације који не дозвољава заглављивање у локалним минимумима.

Од популационих метахеуристика (енгл. *population based*) издвајају се еволутивни алгоритми [30, 31] (енгл. *Evolutionary Algorithm* – EA), алгоритми интелигенције ројева [32] (енгл. *Swarm Intelligence* – SI), оптимизација колонијом мрава [33] (енгл. *Ant Colony Optimization* – ACO) и метода расуте претраге [34, 35] (енгл. *Scatter Search* – SS). Основна идеја је да скуп решења тј. популација напредује кроз итерације применом дефинисаних правила саме метахеуристике. Правила углавном имитирају оптимизационе процесе и механизме који се јављају у природи. На пример, еволутивни алгоритми имитирају процес еволуције јединки кроз селекцију, укрштање и мутацију док алгоритми интелигенције ројева имитирају понашање и организацију скупа јединки у природи (птице, пчеле, мрави). Популационе метахеуристике представљају важан део метода рачунарске интелигенције (енгл. *Computational Intelligence* – CI) поред неуронских мрежа и фази система [36].

У новијим истраживањима често се врши хибридизација, односно комбиновање метахеуристике са неком другом метахеуристиком или методом математичке оптимизације. Хибиридизацијом две метахеуристике у пракси се често долази до бољих решења него када се примењују појединачно [37, 38]. Хибридизација се врши на различите начине, на пример уместо локалне претраге у једној метахеуристици користи се нека друга метахеуристика, или се у оквиру једне популационе метахеуристике на одређен број решења примењује друга метахеуристика вођена једним решењем. У литератури су се посебно издвојили меметски алгоритми [39, 40, 41] (енгл. *Memetic Algorithm* – MA) који представљају хибрид еволутивних алгоритама и локалне претраге. Осим хибрида две метахеуристике у литератури се могу пронаћи и *математичке*, односно методе настале комбинацијом метахеуристика и егзактних метода математичке оптимизације [42, 43, 44]. Више детаља о хибридним методама може се пронаћи у прегледним радовима [45, 46].

Глава 2

Кластеровање на комплексним мрежама

У овом поглављу описан је проблем кластеровања на комплексним мрежама који се разматра у поглављима 3 и 4. У секцији 2.1 описане су комплексне мреже, наведене њихове универзалне карактеристике као и структуре података погодне за представљање у рачунару. Приликом поређења, анализе и избора структуре која ће бити коришћена у наставку узете су у обзир универзалне карактеристике које се испољавају у комплексним мрежама. У секцији 2.2 формално је описан проблем кластеровања на мрежи док су у секцији 2.3 приказани досадашњи приступи за његово решавање који се налазе у литератури.

2.1 Комплексне мреже

Структура комплексног система представља се комплексном мрежом тако што се елементи система представљају чворовима, а интеракције представљају гранама између одговарајућих чворова. Математички концепт који одговара комплексној мрежи је граф. Термини „мрежа" и „граф" се користе као синоними у зависности од контекста. На пример, термин „мрежа" се чешће примењује када се говори о реалним системима, а термин „граф" када се говори само о математичкој репрезентацији. У енглеском језику се уз термин „*network*" најчешће користе „*node*" и „*link*" за појмове „*чвор*" односно „*грана*", док се уз термин „*graph*" чешће користе термини „*vertex*" и „*edge*" из теорије графова.

Кључно својство сваког чвора у комплексној мрежи представља његов степен дефинисан у секцији 1.1.1, док средњи степен чвора представља важну карактеристику мреже. Нека је n број чворова у мрежи, m број грана, и нека су чворови нумерисани $v_1, v_2, \ldots v_n$. Степен чвора v_i у ознаци $k(v_i)$ представља број грана које су инцидентне са њим. Средњи степен чвора у мрежи дефинисан је на следећи начин

$$\langle k \rangle = \frac{1}{n} \sum_{i=1}^{n} k(v_i).$$

На основу теореме 1.1 имамо да је $\sum_{i=1}^{n} k(v_i) = 2m$, па је $\langle k \rangle = \frac{2m}{n}$. Средњи степен чвора показује каква је просечна густина повезаности мреже и омогућава поређење две мреже различитих величина. Густина мреже је

$$d = \frac{2m}{n\,(n-1)} = \frac{\langle k \rangle}{n-1},$$

и представља однос између броја постојећих грана у мрежи и броја грана у потпуно повезаној мрежи са истим бројем чворова.

Друга важна карактеристика мреже је расподела степена чворова која се означава са $P(k)$ и представља вероватноћу да случајно изабран чвор у графу има степен k. Дакле,

$$P(k) = \frac{|\{v_i : k_{v_i} = k\}|}{n}.$$

У случајним мрежама са n чворова код којих се чворови насумично повезују са вероватноћом p (Ердош-Ренји модел [47, 48]) расподела степена чворова има биномну расподелу са параметрима p и n, односно

$$P(k) = \binom{n-1}{k} p^k\,(1-p)^{n-1-k}.$$

Ако је n довољно велико и просечни степен чвора фиксиран, расподела степена чворова се може апроксимирати Пуасоновом расподелом са параметром $\langle k \rangle$, тј.

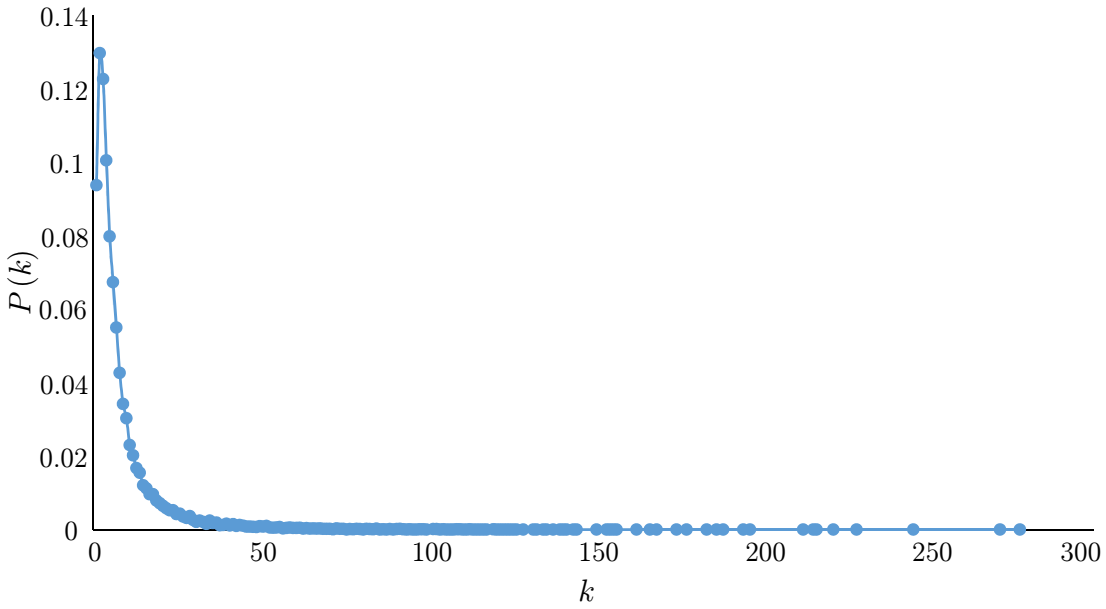
$$P(k) = e^{-\langle k \rangle} \frac{\langle k \rangle^k}{k!}.$$

У реалним комплексним мрежама гране између чворова се не успостављају насумично, а емпиријском анализом утврђено је да расподела степена чворова у већини реалних мрежа има степену расподелу (енгл. *power law distribution*)

$$P(k) \approx k^{-\gamma}.$$

Штавише, у већини комплексних мрежа важи да је $2 < \gamma < 3$ (видети [49, 50, 51]). Како за степену расподелу важи да након скалирања задржава исти облик, тј. $P(ak) = F(a)\,P(k)$, мреже у којима се јавља оваква расподела се називају мреже без скале (енгл. *scale free network*).

Расподела степена чворова у мрежи сарадње између истраживача на пољу физике кондензоване материје [52] приказана је на слици 2.1. Мрежа садржи више од 40000 чворова и 175000 грана, а просечни степен чвора $\langle k \rangle$ је осам.



Слика 2.1: Расподела степена чворова у мрежи сарадње између истраживача на пољу физике кондензоване материје [52]

За разлику од случајних мрежа у којима су гране хомогено распоређене, у мрежама без скале гране су хетерогено распоређене, односно највећи број чворова има мали број суседа, док свега неколико чворова има изузетно велики број суседа. Чворови чији је степен значајно већи од просечног степена чворова називају се хабови (енгл. *hubs*). Услед њиховог присуства растојања између чворова су јако мала и у просеку износе $\ln(\ln n)$ (видети [53]).

Још једно важно својство чвора у мрежи представља *локални коефицијенш ӣрӯйисања* који описује повезаност његових суседа [54]. За чвор v_i локални коефицијент груписања, у ознаци $\widehat{C_{v_i}}$, представља вероватноћу да су два суседа

чвора v_i међусобно повезана тј.

$$\widehat{C_{v_i}} = \frac{2\left|E_{N_{v_i}}\right|}{k_{v_i}\left(k_{v_i}-1\right)},$$

где је $E_{N_{v_i}} = \{\{v_j, v_l\} : v_j, v_l \in N(v_i)\}$ скуп свих грана између суседа чвора v_i. Јасно је да ће $\widehat{C_{v_i}}$ бити један ако су сви суседи чвора v_i међусобно повезани гранама, и нула ако између њих нема грана. На нивоу мреже понекад се посматра средњи коефицијент груписања

$$\langle\widehat{C}\rangle = \frac{1}{n}\sum_{i=1}^{n}\widehat{C_{v_i}}.$$

Ипак, више информација пружа глобални коефицијент груписања дефинисан на следећи начин

$$\widehat{C} = \frac{\sum_{i=1}^{n} k_{v_i}\left(k_{v_i}-1\right)\widehat{C_{v_i}}}{\sum_{i=1}^{n} k_{v_i}\left(k_{v_i}-1\right)}.$$

Овако дефинисан глобални коефицијент груписања представља вероватноћу да су два произвољна суседа неког чвора повезани и може се изразити коришћењем броја триплета у мрежи [54]. Триплет представља скуп од три чвора који су повезани са две гране (отворен триплет) или три гране (затворен триплет). У овим терминима коефицијент груписања представља однос између броја затворених триплета и броја свих триплета у мрежи, односно

$$\widehat{C} = \frac{t_{\triangleleft}}{t_{\triangleleft} + t_{<}},$$

где је $t_{\triangleleft}$ број затворених триплета и $t_{<}$ број отворених триплета.

Последњих десет година теорија комплексних мрежа је у великој експанзији услед развоја рачунарства, а почела је интензивно да се развија крајем деведесетих година прошлог века када су упоредно са емпиријским истраживањима развијени и теоријски модели помоћу којих се може конструисати мрежа са карактеристичним особинама [55, 56, 57]. Доприноси истраживања комплексних мрежа имају широк спектар примена у различитим областима, почев од рачунарских наука, економије и организационих наука, па до социологије, биологије, неуронауке и медицине.

2.1.1 Структуре података за рад са мрежама

Приликом анализирања комплексних мрежа рачунарским методама важно је користити ефикасне структуре података. Структура података за рад

са мрежом односно графом, осим представљања оријентисаних и неоријентисаних грана треба да омогући основне операције као што су:

- `provera`(v_i, v_j) – врши проверу да ли постоји грана између два чвора;
- `dodaj_cvor`(v_i) – додаје чвор у граф;
- `ukloni_cvor`(v_i) – уклања чвор из графа;
- `dodaj_granu`(v_i, v_j) – додаје грану између чворова;
- `ukloni_granu`(v_i, v_j) – уклања грану између чворова.

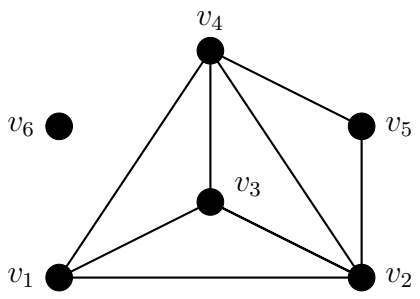
У наставку су описане и упоређене три најчешће коришћене структуре за рад са графовима. Више детаља о структурама података за рад са графовима може се наћи у [9, 8, 58].

Матрица суседства

Нека су чворови оријентисаног графа $G = (V, E)$ нумерисани тако да је $V = \{v_1, v_2, \ldots, v_n\}$. *Матрица суседства* графа G је матрица $A = [a_{ij}]$ димензије $n \times n$ у којој је

$$a_{ij} = \begin{cases} 1, & \text{ако постоји грана од чвора } v_i \text{ до чвора } v_j, \\ 0, & \text{у супротном.} \end{cases}$$

За неоријентисане графове матрица суседства A је увек симетрична, тј. важи $a_{ij} = a_{ji}$, па је евентуално могуће чувати само вредности елемената a_{ij} са и испод главне дијагонале тј. за $i \geq j$. На левој страни слике 2.2 приказан је неоријентисан граф G_N из секције 1.1.1, док је на десној страни исте слике дата одговарајућа матрица суседства.

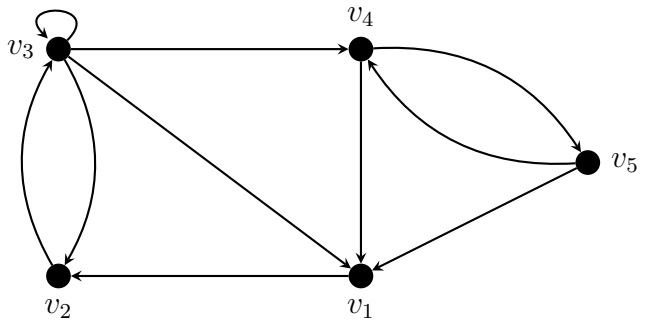


Граф G_N

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	1	1	1	0	0
v_2	1	0	1	1	1	0
v_3	1	1	0	1	0	0
v_4	1	1	1	0	1	0
v_5	0	1	0	1	0	0
v_6	0	0	0	0	0	0

Слика 2.2: Матрица суседства графа G_N

На левој страни слике 2.3 приказан је оријентисан граф G_O из секције 1.1.3, док је на десној страни исте слике дата одговарајућа матрица суседства.



Граф G_O

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

Слика 2.3: Матрица суседства графа G_O

Дакле, i-та врста матрице A је вектор дужине n чија j-та координата чува информацију да ли из чвора v_i постоји грана до чвора v_j. Из тог разлога операције `provera`(v_i, v_j), `dodaj_granu`(v_i, v_j) и `ukloni_granu`(v_i, v_j) се извршавају у константном времену $O(1)$, тј. не зависе од броја чворова и грана у графу. Са друге стране, операције за додавање и уклањање чвора `dodaj_cvor`(v_i), `ukloni_cvor`(v_i, v_j) могу захтевати чак $O(n^2)$ корака уколико проширивање матрице захтева алокацију новог дела меморије и преписивање текућег садржаја меморијског простора.

Из матрице A могу се веома једноставно одредити неке карактеристике графа и његових чворова. За сваки чвор v_i који поседује петљу, елемент a_{ii} у матрици A има вредност 1. Стога, траг матрице суседства $\mathrm{tr}(A) = \sum_{i=1}^{n} a_{ii}$, одређује број петљи у графу G. Степен чвора v_i у простом графу се може одредити као збир елемената у i-тој врсти, тј. важи $k(v_i) = \sum_{j=1}^{n} a_{ij}$. Слично, излазни односно улазни степен чвора у оријентисаном графу може се одредити као збир елемената у врсти односно колони, тј. важи

$$k^+(v_i) = \sum_{j=1}^{n} a_{ij} \quad \text{и} \quad k^-(v_j) = \sum_{i=1}^{n} a_{ij}\,.$$

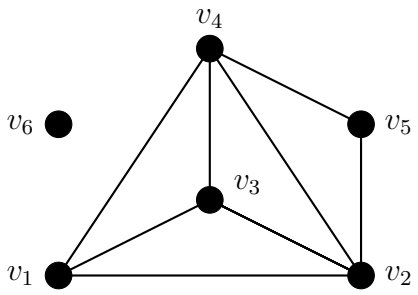
Главни недостатак матрице суседства је то што она захтева простор величине $O(n^2)$, чак и када у графу постоји јако мали број грана.

Листа суседства

У случају графа са великим бројем чворова и малим бројем грана, недостатак матрице суседства се може превазићи тако што се евидентирају само постојеће гране, односно њен почетни и крајњи чвор. За сваки чвор v_i графа $G = (V, E)$ довољно је сачувати следећи скуп чворова

$$l_{v_i} = \begin{cases} \{v_j : \{v_i, v_j\} \in E\}, & \text{ако је } G \text{ неоријентисан граф,} \\ \{v_j : (v_i, v_j) \in E\}, & \text{ако је } G \text{ оријентисан граф.} \end{cases}$$

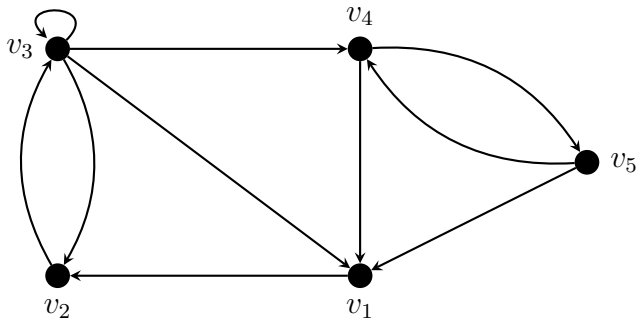
Скуповне листе чворова за неоријентисан граф G_N и оријентисан граф G_O приказане су на сликама 2.4 и 2.5.



$l_{v_1} = \{v_2, v_3, v_4\}$
$l_{v_2} = \{v_1, v_3, v_4, v_5\}$
$l_{v_3} = \{v_1, v_2, v_4\}$
$l_{v_4} = \{v_1, v_2, v_3, v_5\}$
$l_{v_5} = \{v_2, v_4\}$
$l_{v_6} = \emptyset$

Граф G_N

Слика 2.4: Скуповне листе графа G_N

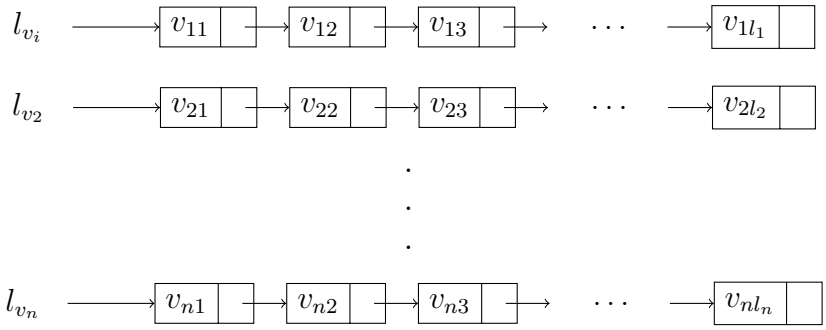


$l_{v_1} = \{v_2\}$
$l_{v_2} = \{v_3\}$
$l_{v_3} = \{v_1, v_2, v_3, v_4\}$
$l_{v_4} = \{v_1, v_5\}$
$l_{v_5} = \{v_1, v_4\}$

Граф G_O

Слика 2.5: Скуповне листе графа G_O

Елементи скупа l_{v_i} се најчешће чувају у једноструко повезаној листи која се назива *листа суседства*. Сваки елемент листе суседства садржи два поља, у првом један елемент скупа l_{v_1} и у другом показивач на следећи елемент у листи (слика 2.6)



Слика 2.6: Листе суседства

На овај начин свака грана је евидентирана помоћу почетног и завршног чвора па је укупан меморијски простор у овом случају $O(n + m)$. Такође, како се дужина листе може динамички мењати, операције `dodaj_cvor(`v_i`)` и `dodaj_granu(`v_i, v_j`)` се извршавају у константном времену $O(1)$, без обзира на број чворова и грана у графу. За операцију `ukloni_cvor(`v_i`)` потребно је обрисати комплетну листу l_{v_i}, али и уклонити чвор v_i из свих преосталих листа суседства, што захтева $O(m)$ корака. За уклањање гране операцијом `ukloni_granu(`v_i, v_j`)` потребно је проналажење и уклањање чвора v_j из листе l_{v_i} што у најгорем случају захтева $O(n)$ корака. Највећи недостатак ове структуре представља сложеност операције `provera(`v_i, v_j`)` која захтева у најгорем случају $O(n)$ корака при провери да ли се чвор v_j налази у листи l_{v_i}. Из листе суседства неоријентисаног графа веома једноставно се може одредити степен и околина произвољног чвора јер важи $N(v_i) = l_{v_i}$ и $k(v_i) = |l_{v_i}|$. Слично, из листе суседства оријентисаног графа може се одредити излазни степен чвора јер је $k^+(v_i) = |l_{v_i}|$.

Матрица инциденције

Нека су чворови и гране графа $G = (V, E)$ нумерисани тако да је $V = \{v_1, v_2, \ldots, v_n\}$ и $E = \{e_1, e_2, \ldots, e_m\}$. *Матрица инциденције* графа G је ма-

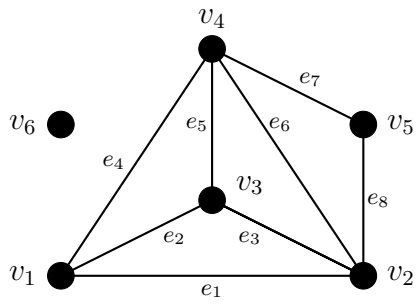
трица $B = [b_{ij}]$ димензије $n \times m$ дефинисана у случају неоријентисаног графа са

$$b_{ij} = \begin{cases} 1, & \text{ако је чвор } v_i \text{ инцидентан са граном } e_j, \\ 0, & \text{у супротном,} \end{cases}$$

а случају оријентисаног графа без петљи са

$$b_{ij} = \begin{cases} 1, & \text{ако је чвор } v_i \text{ почетни чвор гране } e_j, \\ -1, & \text{ако је чвор } v_i \text{ завршни чвор гране } e_j, \\ 0, & \text{у осталим случајевима.} \end{cases}$$

Матрица инциденције за неусмерен граф G_N приказана је на десној страни слике 2.7.



Граф G_N

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
v_1	1	1	0	1	0	0	0	0
v_2	1	0	1	0	0	1	0	1
v_3	0	1	1	0	1	0	0	0
v_4	0	0	0	1	1	1	1	0
v_5	0	0	0	0	0	0	1	1
v_6	0	0	0	0	0	0	0	0

Слика 2.7: Матрица инциденције графа G_N

Матрица инциденције захтева меморијски простор величине $O(n\,m)$. Операција `provera`(v_i, v_j) захтева пролазак кроз све гране, тј. колоне матрице, па је њена сложеност $O(m)$. Све остале операције су сложености $O(n\,m)$ јер захтевају промену величине матрице.

Избор структура

У табели 2.1 приказане су сложености основних операција у различитим структурама за представљање графа G са n чворова и m грана. У последњем реду табеле за сваку структуру наведен је меморијски простор потребан за представљање графа.

Табела 2.1: Сложеност операција у различитим репрезентацијама графа са n чворова и m грана

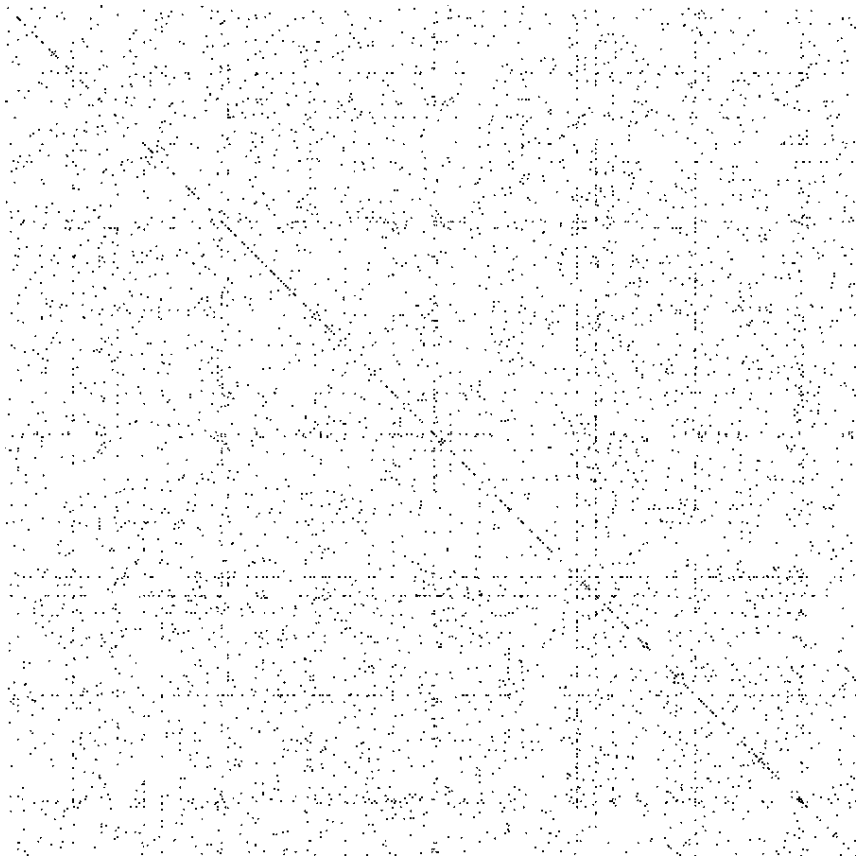
Операција	Матрица инциденције	Матрица суседства	Листа повезаности
`provera`(v_i, v_j)	$O(m)$	$O(1)$	$O(n)$
`dodaj_cvor`(v_i)	$O(n\,m)$	$O(n^2)$	$O(1)$
`dodaj_granu`(v_i, v_j)	$O(n\,m)$	$O(1)$	$O(1)$
`ukloni_cvor`(v_i)	$O(n\,m)$	$O(n^2)$	$O(m)$
`ukloni_granu`(v_i, v_j)	$O(n\,m)$	$O(1)$	$O(n)$
Меморијски простор	$O(n\,m)$	$O(n^2)$	$O(n+m)$

Од наведених структура у практичним применама најчешће се користе матрица суседства и листа повезаности. Наиме, у реалним мрежама број грана m је готово увек већи од броја чворова n и важи $n\,m > n^2$. Стога је матрица суседства у том случају ефикаснија структура од матрице инциденције.

Приликом избора између матрице суседства и листа повезаности потребно је размотрити карактеристике графа са којима се ради и могуће имплементације у програмском језику. На пример, матрица суседства се може имплементирати тако да сваки елемент користи само један бит па се може представити на делу меморије од свега $\frac{n^2}{8}$ бајтова. Са друге стране, наивна имплементација листе суседства на 64-битном рачунару захтева око $12m$ бајтова (у сваком чвору 4 бајта за елемент и 8 бајтова за показивач). Листа суседства ће заузимати више простора од матрице суседства када је

$$12m > \frac{n^2}{8} \Rightarrow \frac{m}{n^2} > \frac{1}{96}.$$

Израз $\frac{m}{n^2}$ је приближно једнак густини графа јер прост граф може имати највише $\frac{n^2-n}{2} = O(n^2)$ грана. Дакле, у оваквој имплементацији листа суседства може да заузме више меморијског простора за представљање графа чија је густина већа од $\frac{1}{96}$. Према томе, графови који се обрађују треба да имају довољно малу густину како би било оправдано коришћење листе суседства, што је управо случај са комплексним мрежама. На слици 2.8 приказана је матрица суседства за интерактом квасца. Мрежу чини 2018 протеина квасца и интеракција између њих. Тачке су постављене на позиције у матрици суседства где је $a_{ij} = 1$, у супротном нема тачке.



Слика 2.8: Илустрација густине матрице суседства – интерактом квасца [54]

Дакле, већина комплексних мрежа има малу густину и велики број чворова [54] тако да је најпожељније представљати их листом суседства са меморијског аспекта. Осим утрошене меморије важно је размотрити и које операције ће најчешће бити примењиване на графу приликом његове обраде. Уколико се често врши провера да ли су два чвора повезана и евентуално се додају или уклањају гране, тада је погодна матрица суседства. У случају када се често ажурира граф додавањем нових чворова и грана, или се често захтева одређивање околине неког чвора, погодна је листа суседства.

2.2 Кластеровање

Кластеровање (енгл. *clustering*) представља процес груписања података тако да степен сличности између елемента буде максималан ако припадају истој групи и минималан ако припадају различитој групи. Групе настале кластеровањем називају се *кластери*, а њихов број и карактеристике нису по-

знате пре спровођења самог поступка. 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.* У литератури на енглеском језику често се користи термин *детекција заједница* (енгл. *community detection*). Кластер (заједница) у једној мрежи представља скуп елемената који су међусобно густо повезани, а ретко са остатком мреже јер интеракције у систему тј. повезаност елемената у мрежи одређује степен сличности међу њима.

Потреба за кластеровањем се може јавити у различитим областима почев од анализе социјалних мрежа и идентификације заједница у њима па до анализе метаболичких мрежа и одређивања функције протеина. Осим директне примене кластеровање може представљати само један корак у обради и припреми података за друге технике истраживања. На пример, приликом истраживања велике количине података, а услед ограничења у рачунарским ресурсима, скуп података се може редуковати избором једног или више представника из сваког кластера. Слично, одударајући подаци (енгл. *outliers*) који се често одстрањују из скупа података за обучавање техникама машинског учења, могу бити идентификовани кластеровањем.

Нека је $G = (V, E)$ граф са n чворова и m грана. Нека је I коначан скуп индекса и $\mathcal{P} = \{C_i : C_i \subset V,\ i \in I\}$ *партиција* графа G, тј. колекција подскупова скупа V таква да:

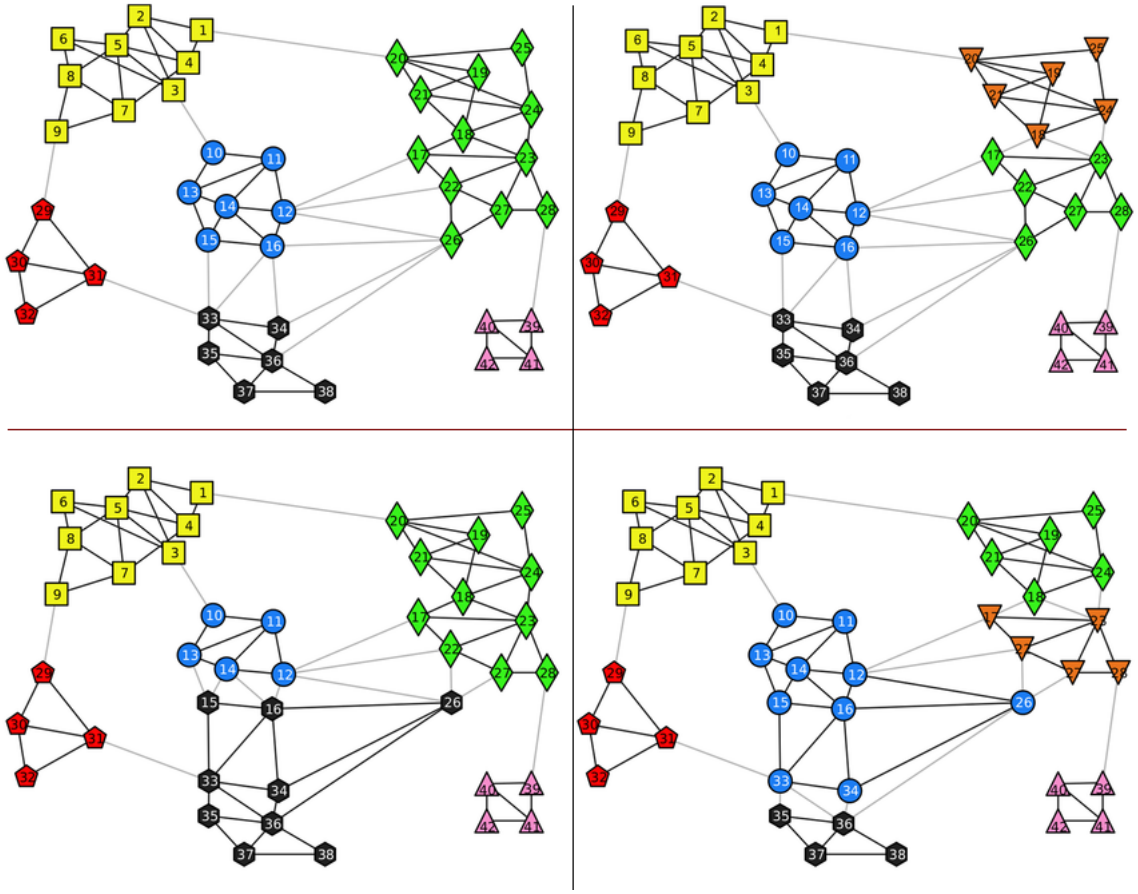
1) $\forall i \in I, \quad C_i \neq \emptyset$;

2) $\forall i, j \in I,\ i \neq j, \quad C_i \cap C_j = \emptyset$;

3) $\bigcup_{i \in I} C_i = V$.

Резултат кластеровања на мрежи представља партиција $\mathcal{P}$ чији елементи C_i индукују подграфове G_{C_i}, $\forall i \in I$, који представљају кластере. Често и саме подскупове C_i називамо кластерима јер на једнозначан начин одређују G_{C_i}. Број кластера у графу G није унапред познат, тј. одређивање партиције није једнозначно дефинисано у односу на кардиналност скупа I. Такође, одређивање партиције није једнозначно дефинисано ни у односу на идеју шта један подграф индукован скупом чворова $C \in \mathcal{P}$ чини кластером. Ово није последица недовољно промишљене дефиниције проблема, већ разноврсности

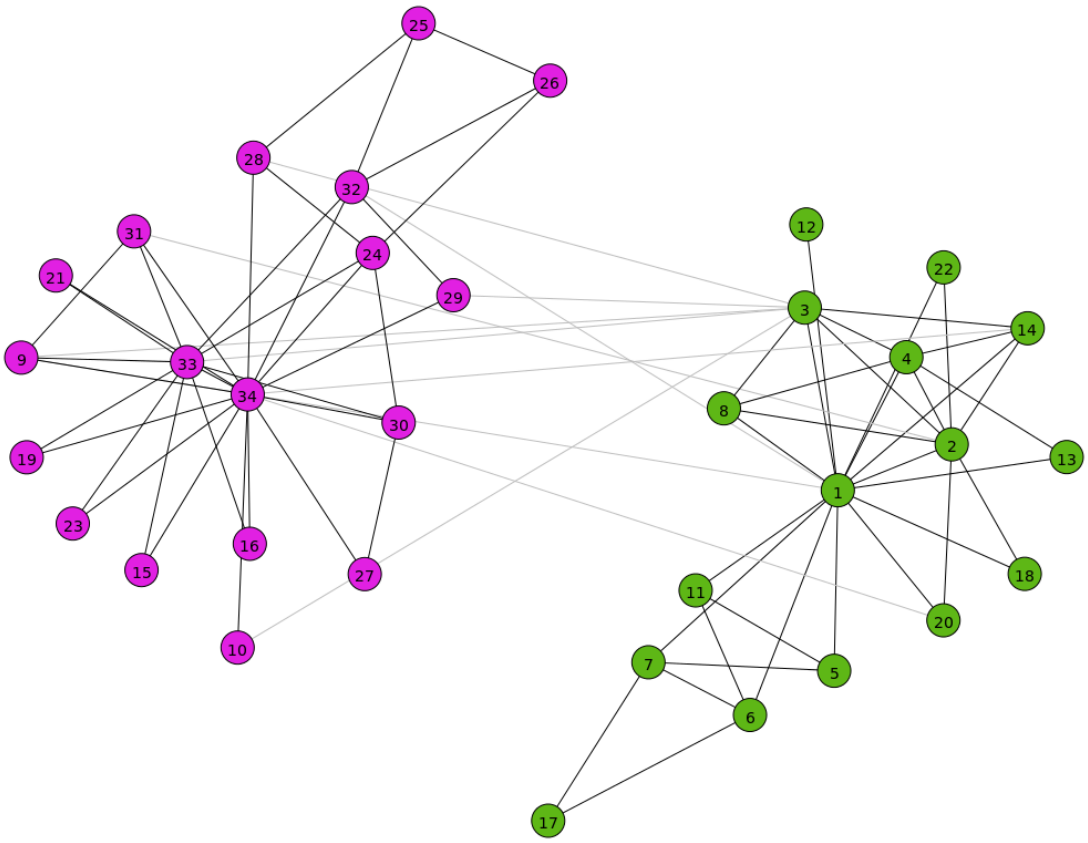
контекста у којима се може вршити кластеровање и циљева који се кластеровањем желе постићи. Пример на слици 2.9 показује да у једној мрежи може постојати више различитих подела у кластере.



Слика 2.9: Различити начини кластеровања једне мреже

За кластеровање на мрежама истраживачи покушавају да креирају опште технике које се могу користити и прилагодити у зависности од контекста и доступних информација о самој мрежи. Приликом креирања метода користе се различити приступи засновани на оптимизацији, спектралној теорији графова, статистичком закључивању и динамичким процесима. Развијање метода за решавање проблема кластеровања и њихова примена су од великог значаја за разумевање динамике и еволуције комплексних система [59, 60, 61, 62, 63, 64, 65]. Осим тога, могу омогућити бољу визуализацију и пружити неопходне информације о појединим чворовима и њиховим улогама у мрежи. На пример, поједини чворови у кластеру могу имати улогу у повезивању кластера са остатком мреже док други чворови могу имати улогу

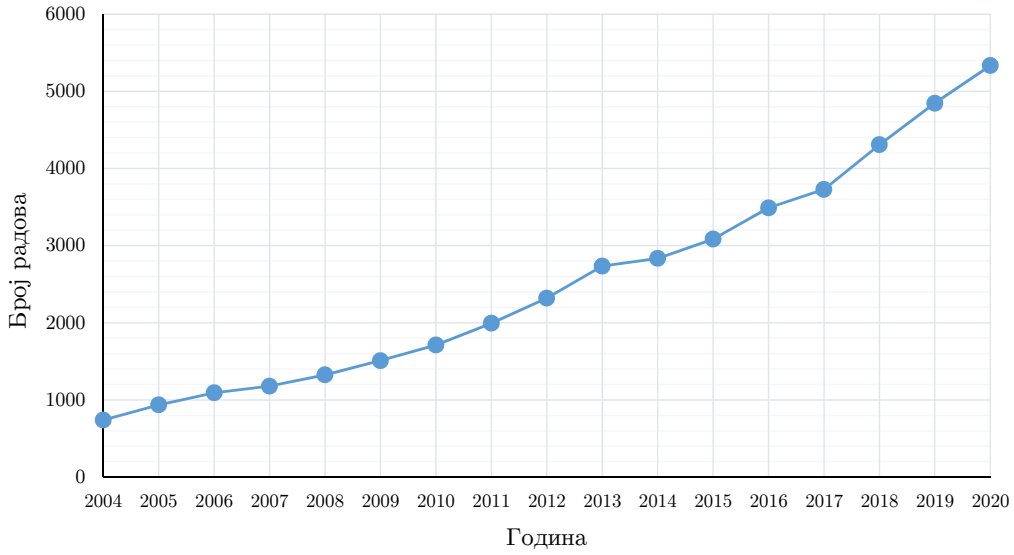
контролисања и стабилизације кластера. Једну од најчешће коришћених мрежа за илустрацију кластеровања представља мрежа под називом Захаријев карате клуб (енгл. *Zachary's karate club*). Мрежа је први пут описана у раду [66], а настала је као резултат Захаријевог надгледања интеракција између чланова једног универзитетског карате клуба у периоду од 1970. до 1972. године. Тада је између 34 члана уочено 78 интеракција ван карате клуба (слика 2.10). Сукоб између тренера (чвор 1) и председника (чвор 34) проузроковао је поделу чланова клуба у две групе. Један део чланова формирао је нови клуб са тренером, док су остали чланови окупљени око председника клуба ангажовали новог тренера. На основу прикупљених података Захарије је коректно предвидео одлуке свих чланова, осим члана број 9 који се придружио групи окупљеној око тренера уместо групи око председника.



Слика 2.10: Захаријев карате клуб

2.3 Преглед литературе

Центар за дискретну математику и теоријско рачунарство (енгл. *Center for Discrete Mathematics and Theoretical Computer Science* – DIMACS), који представља конзорцијум престижних академских институција (Универзитет Рутгерс, Принстон, Колумбија) и истраживачких лабораторија (Microsoft, IBM, AT&T, NEC), сврстао је 2012. године проблем кластеровања на мрежи на листу најважнијих проблема и изазова у рачунарству [1] за које је неопходно развити и имплементирати ефикасне алгоритме (енгл. *implementation challenges*). Од тада број научних радова посвећен овом проблему сваке године се увећава (слика 2.11) и тренутно достиже преко 10000 радова.



Слика 2.11: Број научних радова о проблему кластеровања на мрежи у периоду од 2004. до 2020. године

2.3.1 Приступи за решавање проблема кластеровања на мрежи

Приликом креирања метода за кластеровање на мрежи пожељно је формално дефинисати појам кластера. Филип Радичи и сарадници у раду [67] покушали су да дефинишу шта један индуковани подграф G_C чини кластером у јаком и слабом смислу. По њиховој дефиницији индуковани подграф

[1] http://dimacs.rutgers.edu/programs/challenge/

G_C представља кластер у јаком смислу (енгл. *cluster in a strong sense*) ако сваки чвор има више грана према чворовима из кластера, него према чворовима из остатка мреже, то јест:

$$k_v^+(C) > k_v^-(C),\ \forall v \in C.$$

Индуковани подграф G_C представља кластер у слабом смислу (енгл. *cluster in a weak sense*) ако је укупан број грана између чворова из C већи од укупног броја грана између чворова из C и остатка мреже, тј. ако важи:

$$\sum_{v \in C} k_v^+(C) > \sum_{v \in C} k_v^-(C)\,.$$

Са друге стране, многи истраживачи [68, 69, 70, 71] сматрају да кластери треба да представљају финални резултат алгоритма без прецизне априорне дефиниције. Алгоритам пропагације ознака (енгл. *Label Propagation Algorithm* – LPA) представљен у раду [70] је најчешће коришћен представник ове класе алгоритама. Идеја алгоритма је веома једноставна и састоји се у итеративном процесу кроз који се чворови групишу у кластере. На почетку сви чворови имају јединствене ознаке које представљају кластере. У свакој итерацији врши се пропагација тако што сваки чвор ажурира своју ознаку на ознаку која је најфреквентнија међу његовим суседима. Формално правило ажурирања ознака за чвор v_i гласи:

$$l_{v_i}^* = \underset{l}{\operatorname{argmax}} \left(\sum_{j=1}^{n} a_{ij}\, \delta(l_{v_j}, l) \right),$$

где је l_{v_i} текућа, а $l_{v_i}^*$ нова ознака чвора v_i. Кронекер делта функција δ има вредност један само ако су аргументи једнаки, у супротном има вредност нула, то јест:

$$\delta(i,j) = \begin{cases} 1, & \text{ако је } i = j, \\ 0, & \text{у супротном.} \end{cases}$$

Уколико постоји више ознака са максималном фреквенцијом, бира се једна случајна, а поступак се понавља све док сваки чвор не добије ознаку која је најчешћа међу ознакама његових суседа. На крају, повезане групе чворова које имају исту ознаку представљају кластере.

Временска сложеност LPA алгоритма је приближно линеарна. Иницијализација свих чворова са јединственом ознаком захтева $O(n)$ времена. Итерација пропагације ознака за један чвор v састоји се од два корака: у првом кораку

суседи се групишу по њиховим ознакама, што захтева $O(k_v)$ корака, а затим се у другом кораку бира ознака групе са највећим бројем чворова (у најгорем случају неопходно је $O(k_v)$ корака). Како се пропагација ознака врши за све чворове, на основу теореме 1.1 добијамо да је сложеност једне итерације $O(m)$. Није могуће прецизно утврдити колико ће се итерација извршити до испуњења услова заустављања, али експериментални резултати спроведени у [70] показују да је број итерација врло мали (око 5) и да не зависи од броја чворова у графу. Важно је напоменути да се понекад као резултат могу добити две или више група чворова са истим ознакама, а које су међусобно повезане преко чворова са другим ознакама. Ово се дешава када два или више суседа добију ознаку од једног чвора, проследе је ка различитим групама чворова, а затим промене своју ознаку. Из тог разлога, након добијања резултата, потребно је применити алгоритам претраге у ширину (енгл. *Breadth-First Search* – BFS) на сваки индуковани подграф скупом чворова који носе исту ознаку како би се добиле компоненте повезаности, односно финални кластери. Временска сложеност овог корака је $O(n + m)$.

Приступ који је привукао највећу пажњу истраживача подразумева дефинисање мере за одређивање квалитета партиције и конструисање метода за проналажење партиције која има максималну вредност дефинисане мере квалитета. Оваквим приступом проблем кластеровања се формулише као проблем комбинаторне оптимизације, а за решавање се могу користити различите методе математичке оптимизације. Формално, мера квалитета (енгл. *quality function*) партиције представља функцију $f : \mathcal{P} \longrightarrow \mathbb{R}$ која свакој партицији додељује коначан реални број. Након дефинисања функције f проблем кластеровања на мрежи која је представљена графом G може се формално записати на следећи начин

$$\max_{\mathcal{P} \in \mathcal{P}_G} f(\mathcal{P}),$$

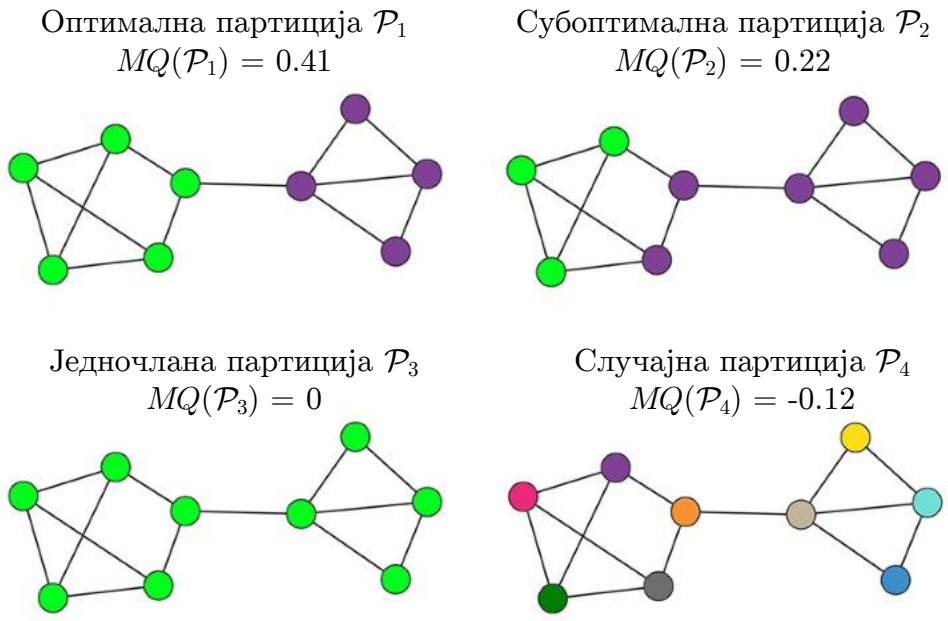
где је $\mathcal{P}_G$ скуп свих партиција графа G.

Од дефинисаних мера квалитета, у литератури се најчешће разматра модуларност (енгл. *modularity*), коју су 2004. године дефинисали Марк Њуман и Мишел Гирван [72]. Модуларност партиције $\mathcal{P}$ је дефинисана са

$$MQ(\mathcal{P}) = \sum_{C \in \mathcal{P}} \left(\frac{m_C}{m} - \left(\frac{k_C^*}{2m} \right)^2 \right),$$

где је m_C број грана између чворова из скупа C и $k_C^* = \sum_{v \in C} k_v$ збир степена

чворова из скупа C. Израз $\frac{m_C}{m}$ представља однос између броја грана у кластеру C и укупног броја грана у графу, док је $\frac{k_C^*}{2m}$ очекивана вредност истог односа у мрежи са чворовима истог степена, а случајно постављеним гранама. Вредности модуларности за различите партиције мреже приказане су на слици 2.12.



Слика 2.12: Вредност модуларности за различите партиције у мрежи [54]

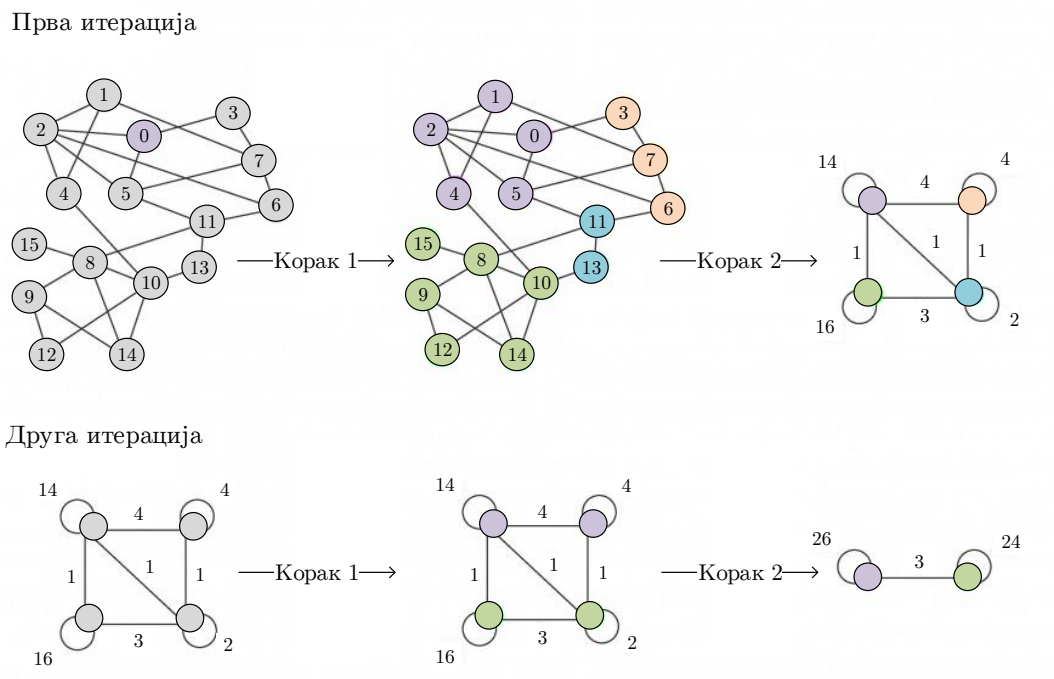
Дакле, у контексту модуларности индуковани подграф скупом чворова C представља кластер уколико је укупан број грана у њему већи од очекиваног броја. Очекивани број грана се израчунава у нултом моделу (енгл. *null model*) који у оригиналној дефиницији представља граф у коме су гране случајно расподељене између чворова уз услов да сваки чвор задржи свој степен. Овакве случајне графове који су одређени степенима чворова разматрао је Болобас још 1980. године [73]. Ретко се користе други нулти модели, попут Ердош-Ренјевог модела, Чунг-Луовог модела [74] или коригованог блок модела [75].

Поред наведених приступа, у литератури се могу пронаћи методе засноване на динамичким процесима и статистичком закључивању [76, 77, 78]. Више детаља о свим овим приступима може се наћи у прегледним радовима [79, 80].

2.3.2 Методе кластеровања засноване на максимизацији модуларности

Максимизација модуларности, односно одређивање партиције са највећом вредношћу модуларности представља NP-тежак [81] проблем, па се за решавање најчешће развијају хеуристичке методе. Развијене егзактне методе за максимизацију модуларности [82, 83] могу у разумном времену извршити кластеровање мреже са свега пар стотина чворова.

Луванова метода [84] (енгл. *Louvain method*) представља похлепну методу засновану на оптимизацији модуларности. Метода се састоји из два корака која се итеративно понављају све док се MQ може повећати. У првом кораку се идентификују кластери оптимизацијом модуларности на локалном нивоу, а затим се у другом кораку креира нова мрежа агрегацијом чворова који припадају истом кластеру у један чвор. Локална оптимизација модуларности почиње од партиције у којој се сваки чвор придружује у засебан кластер, а затим се покушава са повећањем модуларности померањем сваког чвора у кластере којима припадају његови суседи. Процес се завршава када се достигне локални максимум, а затим се прелази на други корак. Пример извршавања методе приказан је слици 2.13.



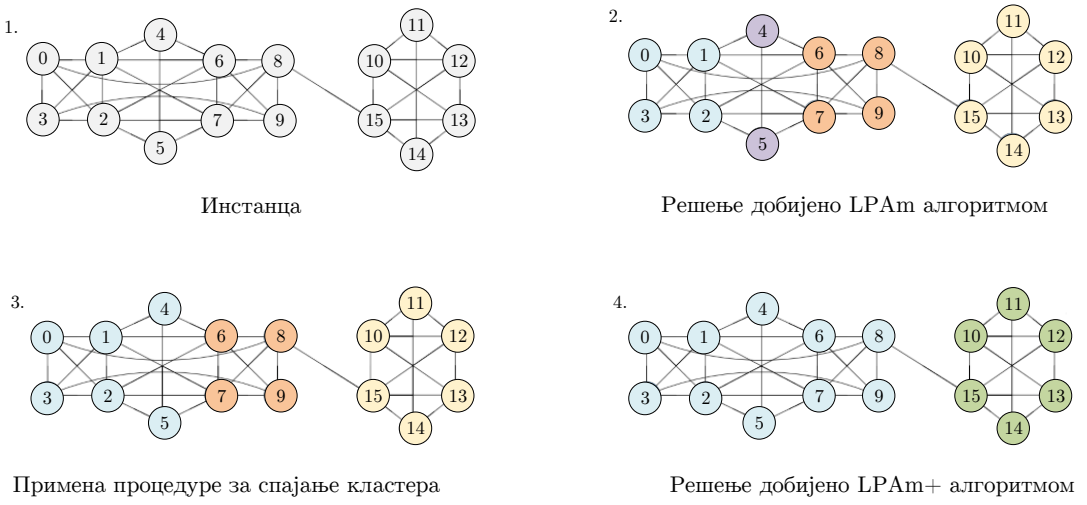
Слика 2.13: Максимизација модуларности Луванoвом методом

Често се врши хибридизација метода које имају различит приступ проблему и на тај начин се користе предности сваког приступа. Пример таквог алгоритма је специјализован алгоритам пропагације ознака за оптимизацију модуларности (енгл. *Modularity-specialized Label Propagation Algorithm* – LPAm) предложен у раду [85]. Барбер и Кларк, аутори LPAm методе, надоградили су правило за ажурирање ознака тако да се приликом избора нове ознаке не бира најфреквентнија ознака међу суседима, већ она која даје највећи допринос повећању MQ функције:

$$l^*_{v_i} = \underset{l}{\operatorname{argmax}} \left(\sum_{j=1}^{n} \left(a_{ij} - \frac{k_{v_i} k_{v_j}}{m} \right) \delta(l_{v_j}, l) \right).$$

Овом хибридизацијом задржана је ефикасност као код LPA алгоритма, а уз то поправљен је квалитет резултујућих партиција у контексту модуларности. Лоша страна LPAm алгоритма је да фаворизује поделе на кластере сличног укупног степена чворова, па стога завршава са радом након достизања неког лошег локалног максимума MQ функције.

Лиу и Мурата су у раду [86] предложили напредну верзију LPAm алгоритма (енгл. *Advanced modularity-specialized label propagation algorithm* – LPAm+) са процедуром за превазилажење локалних максимума удаљених од глобалног максимума. Пример који илуструје недостатак LPAm алгоритма и идеју за превазилажење локалног максимума MQ функције приказан је на слици 2.14.



Слика 2.14: Максимизација модуларности LPAm+ методом

LPAm+ алгоритам почиње применом LPAm алгоритма на дату мрежу. Након добијеног решења примењује се похлепни алгоритам за спајање добијених кластера у циљу превазилажења локалног максимума. Прво се за све парове кластера (C_i, C_j), $i, j \in I$ израчунава промена вредности MQ функције $(\Delta MQ_{C_i C_j})$ која би настала спајањем ових кластера. Затим се врше спајања која ће највише увећавати вредност MQ функције. Уколико је било успешних спајања кластера, примењује се поново LPAm алгоритам и цео поступак спајања се понавља. Алгоритам престаје са радом уколико не постоји пар кластера чијим се спајањем модуларност повећава. Псеудокод LPAm+ методе приказан је алгоритмом 2.1.

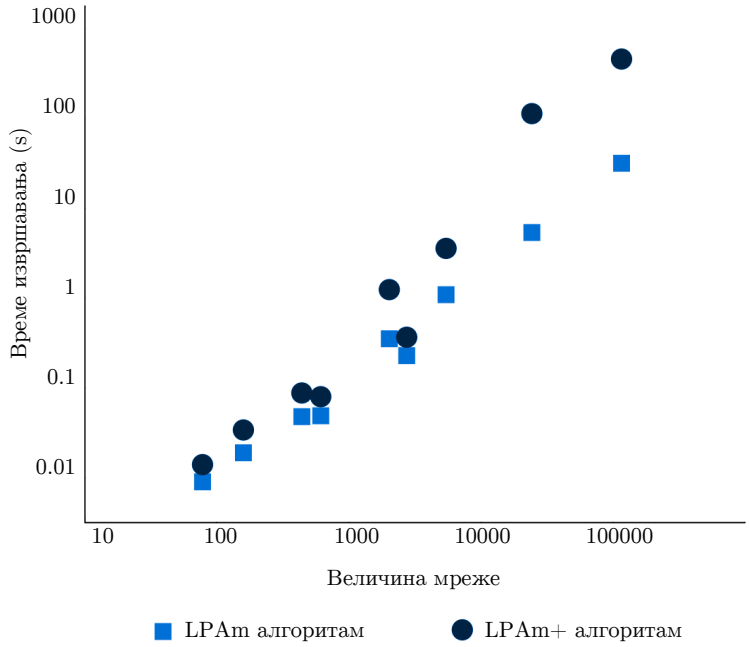
Алгоритам 2.1: Псеудокод LPAm+ алгоритма

улазни подаци: граф G

излазни подаци: партиција $\mathcal{P}$

```
1 P ← Сваки чвор добија јединствену ознаку
2 P ← Максимизација модуларности LPAm алгоритмом
3 while ∃(C1, C2) ∈ P² : ΔMQ_{C1 C2} > 0 do
4     for (∀(C1, C2)) ∈ P² : (ΔMQ_{C1 C2} > 0 ∧ (∄C ∈ P)(ΔMQ_{C C1} >
        ΔMQ_{C1 C2} ∨ ΔMQ_{C C2} > ΔMQ_{C1 C2})) do
5         P ← Спајање кластера C1 и C2
6     P ← Максимизација модуларности LPAm алгоритмом
```

Одређивање прецизне временске сложености LPAm+ алгоритма није могуће јер зависи од резултата добијених применом LPAm. Експериментална тестирања која су аутори методе спровели у [86], показују да LPAm+ алгоритам проналази квалитетнија решења у односу на LPAm алгоритам, али да такође има и дуже време извршавања, нарочито на мрежама великих димензија. Поређење брзине извршавања LPAm и LPAm+ алгоритама у мрежама различите величине приказано је на слици 2.15.



Слика 2.15: Поређење брзине извршавања LPAm и LPAm+ алгоритама [86]

Претходно описане методе имају одређено време извршавања на које се не може утицати са циљем да се добије квалитетније решење. То је један од разлога зашто се у литератури предлажу многобројне метахеуристике код којих се може утицати на време извршавања. Повећавањем границе максималног времена извршавања очекује се добијање квалитетнијих решења. До сада су развијене различите методе засноване на еволутивним алгоритмима [87, 88, 89], табу претрази [90, 91], симулираном каљењу [92, 93], оптимизацијом колонијом мрава [94, 95] и другим метахеуристикама [83, 96, 97].

2.3.3 Недостаци MQ функције

Максимизацијом модуларности у мрежама великих димензија често се не могу открити кластери са малим бројем чворова, чак и када су они очигледни. У литератури се овај недостатак назива *проблем ограничене резолуције* (енгл. *resolution limit problem*), а уочили су га Фортунато и Бартелеми [98] у раду из 2007. године кроз анализу која је представљена у наставку.

Посматрајмо два подграфа G_1 и G_2 графа G који има n чворова и m грана. Нека је $k^*_{G_1}$ укупан степен чворова за G_1 и $k^*_{G_2}$ за G_2. Израчунајмо очекивани број грана између G_1 и G_2 у референтној мрежи одабраној за модуларност, тј.

мрежи са n чворова и m грана које су случајно успостављене, али тако да сви чворови имају исти степен као у графу G. Приметимо да управо због услова о преносу степена сваког чвора, у референтној мрежи укупан степен чворова се неће променити за подграфове G_1 и G_2. Свака грана је на јединствен начин одређена са два чвора која повезује. Вероватноћа да је један чвор из G_1 је $p_1 = \dfrac{k^*_{G_1}}{2m}$ и слично вероватноћа да је један чвор из G_2 је $p_2 = \dfrac{k^*_{G_2}}{2m}$. Стога, имамо да је вероватноћа гране између G_1 и G_2:

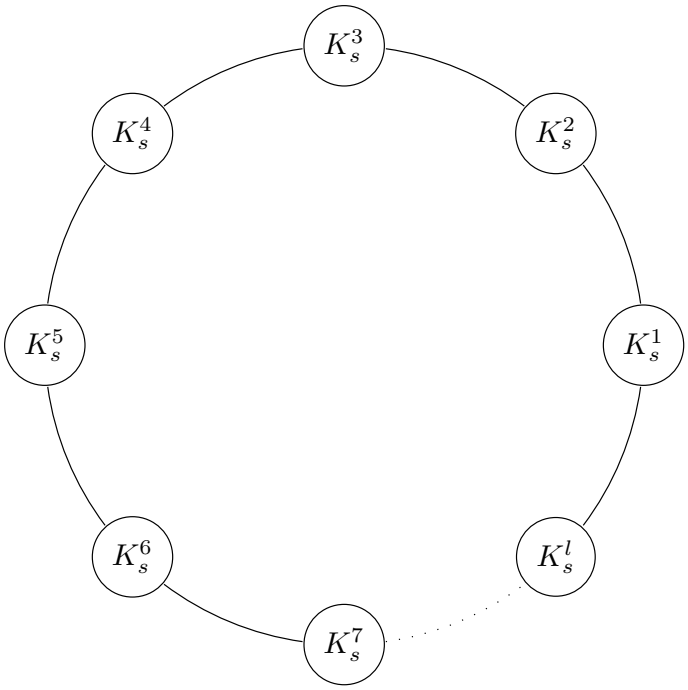
$$p = 2\,p_1\,p_2 = \frac{k^*_{G_1}\,k^*_{G_2}}{2m^2}.$$

Како имамо m грана у референтној случајној мрежи, очекивани број грана између G_1 и G_2 је:

$$\widehat{E} = m\,p = \frac{k^*_{G_1}\,k^*_{G_2}}{2m}.$$

У претходној анализи параметри $k^*_{G_1}$ и $k^*_{G_2}$ не подлежу ниједном ограничењу, осим да не могу бити већи од укупног степена целе мреже ($k^*_{G_1} < 2m$ и $k^*_{G_2} < 2m$). То значи да могу бити доста мањи од m чиме би очекивани број грана био мањи од 1. Прецизније, ако поједноставимо анализу и претпоставимо да је $k^*_{G_1} = k^*_{G_1} = k^*$, онда ће $\widehat{E} < 1$ за $k^* < \sqrt{2m}$. Дакле, максимизацијом модуларности у мрежи са m грана, подграфови који имају мање од $\sqrt{m}$ грана не могу бити идентификовани као кластери, тј. такви подграфови у оптималној партицији ће увек бити само део неког кластера. Овај закључак важи и за потпуне подграфове са мање од $\sqrt{m}$ грана, па као резултат кластеровања максимизацијом модуларности можемо добити кластер који се састоји од два потпуна подграфа која су повезана само једном граном.

Изложени недостатак модуларности као мере квалитета партиције се може сагледати и на конкретном примеру. Нека је K_s клика са s чворова. Посматрајмо граф $G(l, s)$ који је састављен од l клика K_s у низу, тако да су сваке две суседне повезане једном граном преко заједничког чвора. Такође, прва и последња клика су повезане једном граном. Укупан број чворова графа G је $n = s\,l$, док је укупан број грана $m = l\,\left(\frac{s\,(s-1)}{2} + 1\right)$.



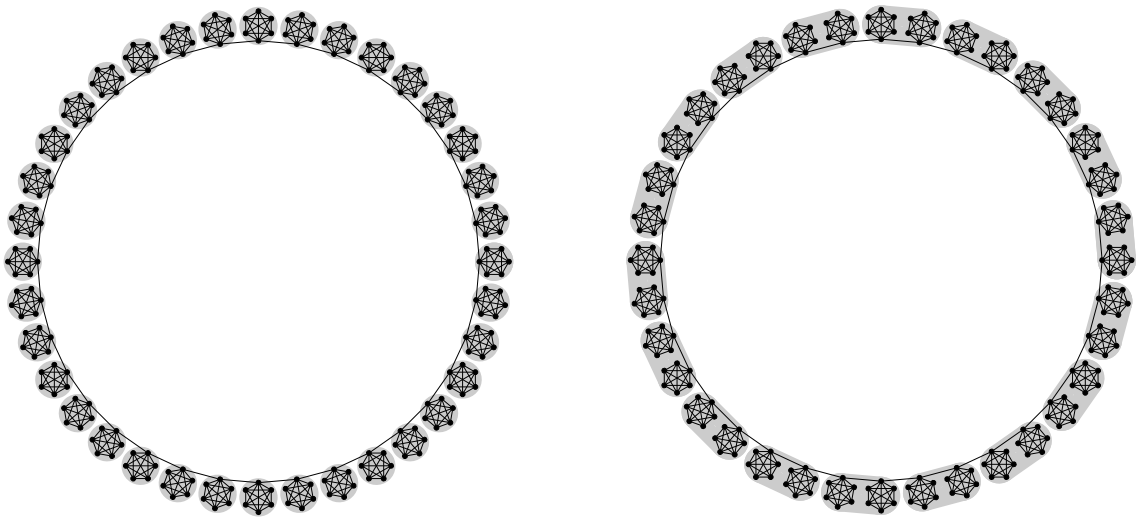
Слика 2.16: Структура графа $G(l, s)$

Очекивано је да кластеровање на графу $G(l, s)$ резултира партицијом $\mathcal{P}_1$ у којој кластери C_i одговарају свакој од појединачних клика K_i, $i = 1, \ldots, l$ (слика 2.17). У складу са тим, природно је очекивати да функција модуларности достиже највећу вредност управо на партицији $\mathcal{P}_1$, али то није увек случај. Број грана у сваком кластеру C_i партиције $\mathcal{P}_1$ је $m_{C_i} = \frac{s\,(s-1)}{2}$, док је збир степена свих чворова $d^*_{C_i} = s\,(s-1) + 2$, одакле следи да је модуларност партиције $\mathcal{P}_1$

$$MQ(\mathcal{P}_1) = 1 - \frac{2}{s\,(s-1)+2} - \frac{1}{l}.$$

Нека је $\mathcal{P}_2$ партиција графа $G(l, s)$ у којој сваки кластер одговара паровима суседних клика (слика 2.17). Без губитка општости, а у циљу поједностављења рачуна, претпоставимо да је l паран број. Како је у сваком кластеру C_i, $i = 1, \ldots, \frac{l}{2}$ партиције $\mathcal{P}_2$ број грана $m_{C_i} = s\,(s-1)+1$, а збир степена свих чворова $d^*_{C_i} = 2\,(s\,(s-1)+2)$, следи да је модуларност партиције $\mathcal{P}_2$

$$MQ(\mathcal{P}_2) = 1 - \frac{1}{s\,(s-1)+2} - \frac{2}{l}.$$



Слика 2.17: Партиције $\mathcal{P}_1$ и $\mathcal{P}_2$ у графу $G(36, 6)$

Вредност $MQ(\mathcal{P}_1)$ ће бити већа од вредности $MQ(\mathcal{P}_2)$ ако и само ако важи

$$1 - \frac{2}{s\,(s-1)+2} - \frac{1}{l} > 1 - \frac{1}{s\,(s-1)+2} - \frac{2}{l}$$

што је еквивалентно неједнакости

$$s\,(s-1)+2 > l. \tag{2.1}$$

Неједнакост (2.1) неће увек бити тачна, на пример за $l = 36$ и $s = 6$ тј. у графу $G(36, 6)$ је

$$0.9097 = MQ(\mathcal{P}_1) < MQ(\mathcal{P}_2) = 0.9131,$$

што значи да исправна партиција $\mathcal{P}_1$ не може бити добијена максимизацијом модуларности.

Структура комплексних мрежа из реалног света се значајно разликује од посматраног графа G, али генерално очекивани број грана између подграфова опада како величина мреже расте. Дакле, ако мрежа има довољно велики број грана, очекивани број грана може бити мањи од један. Ако се то догоди, само једна грана између два подграфа у контексту модуларности се тумачи као знак снажне корелације а максимизација модуларности доводи до спајања та два подграфа у један кластер, независно од њихових карактеристика.

2.3.4 Остале функције за мерење квалитета партиције

Након што су Фортунато и Бартелеми идентификовали проблем ограничене резолуције модуларности у раду [98], појавило се неколико различитих идеја за поправку саме модуларности, али и за дефинисање нове мере. Неколико приступа је засновано на увођењу параметра резолуције који омогућава идентификацију кластера у различитим резолуцијама мреже и називају се *више-резолуцијски приступи* (енгл. *multi-resolution approaches*).

На пример, Аренас и сарадници у раду [99] предлажу додавање петље сваком чвору са тежином $r > 0$ и оптимизацију модуларности за различите вредности параметра r у тако добијеном тежинском графу. На овај начин се укупна тежина мреже може скалирати без промене топологије, односно могу се идентификовати мањи кластери за веће вредности параметра r и већи кластери за мање вредности r.

Понс и Латапи у раду [100] предлажу увођење параметра $r \in (0, 1)$ за регулисање разлике између стварног и очекиваног односа грана у дефиницији модуларности чиме се добија модуларност са више скала:

$$MQ_r(\mathcal{P}) = \sum_{C \in \mathcal{P}} (r\, a_C - (1 - r)\, e_C).$$

Максимизацијом MQ_r за различите вредности параметра r партиције ће садржати веће или мање кластере, а неки кластери ће бити присутни кроз низ узастопних партиција. Као добар индикатор стабилности кластера C, у [100] предлаже се дужина сегмента $[r_{min}(C), r_{max}(C)]$ за који се кластер C појављује у партицији добијеној максимизацијом MQ_r. Иако више-резолуцијске верзије модуларности омогућавају препознавање мањих или већих кластера, оне не отклањају проблем ограничене резолуције. Сада се истовремено јавља тенденција спајања мањих кластера и тенденција дељења већих кластера, а подешавање параметра резолуције тако да се оне истовремено заобиђу је веома тешко а често и немогуће [101].

Са друге стране, Ли и сарадници су 2008. године у раду [102] предложили нову меру, названу модуларност густине, која не укључује параметре у својој дефиницији. Идеја функције је да се укључи информација о броју чворова у кластеру како би се избегло занемаривање малих, а густих кластера. Сваки кластер C окарактерисан је просечним степеном модуларности $k_{avg}(C)$ који је дефинисан као разлика између унутрашњег $k^{+}_{avg}(C)$ и спољашњег просечног степена $k^{-}_{avg}(C)$. Просечни унутрашњи степен подграфа C представља

просечан степен чворова из подграфа C, односно $\frac{2m_c}{n_c}$. Слично, просечни спољашњи степен представља просечни степен чворова одређен само узимајући у обзир гране које подграф C повезују са остатком мреже тј. $\frac{l_c}{n_c}$. Модуларност густине партиције $\mathcal{P}$ представља збир просечних степена модуларности свих кластера $C \in \mathcal{P}$, то јест:

$$MDQ_1(\mathcal{P}) = \sum_{C \in \mathcal{P}} k_{avg}(C) = \sum_{C \in \mathcal{P}} \left(k^+_{avg}(C) - k^-_{avg}(C) \right) = \sum_{C \in \mathcal{P}} \left(\frac{2m_c - l_c}{n_c} \right).$$

Испитивање својства MDQ_1 функције и експерименталним тестирањем, аутори су у раду [102] показали да предложена функција мере при максимизацији не испољава нежељене ефекте као оригинална модуларност.

Пет година касније, Чен и сарадници су у раду [103] предложили суштински другачију функцију мере, али са истим називом:

$$MDQ_2(\mathcal{P}) = \sum_{C \in \mathcal{P}} \left[\frac{m_c}{m} d_C - \left(\frac{k^*_C}{2m} d_C \right)^2 - \left(\sum_{C' \in \mathcal{P}/C} \frac{m_{CC'}}{2m} d_{CC'} \right) \right],$$

где је $d_C = \frac{2m_C}{n_C(n_C-1)}$ густина грана у кластеру C, и $d_{CC'} = \frac{m_{CC'}}{n_C\, n_{C'}}$ густина грана између кластера C и C'. Ова функција доноси две суштинске промене у односу на оригиналну модуларност. Прва промена се огледа у увођењу експлицитне казнене функције за гране између чворова у различитим кластерима. На овај начин се донекле ограничава и регулише подела великих кластера у мање кластере јер се сваком поделом повећава број грана на које се примењује казнена функција. Друга промена се огледа у томе што су сви изрази, па и казнена функција, пондерисани изразима који зависе од броја чворова у кластерима. Из тог разлога, одређен број грана између два мала кластера има већу вредност казнене функције, него исти број грана између два велика кластера. Дакле, MDQ_2 превазилази недостатке оригиналне модуларности коришћењем локалних особина кластера као што је густина.

Миаучи и Кавеса су 2016. године у раду [104] предложили нову функцију мере названу Z-модуларност. Два кластера C_1 и C_2 за које важи да је $a_{C_1} = 0.2$, $a_{C_2} = 0.6$ и $e_{C_1} = 0.6$, $e_{C_2} = 0.5$ у контексту модуларности имају исти допринос $a_{C_1} - e_{C_1} = a_{C_2} - e_{C_2} = 0.1$ у укупном квалитету партиције. Аутори Z-модуларности [104] предлажу да се узима у обзир статистичка реткост кластера тако да у овом конкретном случају кластер C_1 има већи допринос. Полазећи од идеје референтног модела коришћеног у оригиналној дефиници-

ји модуларности, аутори су квантификовали статистичку реткост партиције (Z-модуларност) користећи нормализовано одступање (Z-скор):

$$ZMQ_2(\mathcal{P}) = \frac{\sum_{C\in\mathcal{P}} \frac{m_c}{m} - \sum_{C\in\mathcal{P}} \left(\frac{k_C^*}{2m}\right)^2}{\sqrt{\sum_{C\in\mathcal{P}} \left(\frac{k_C^*}{2m}\right)^2 \left(1 - \sum_{C\in\mathcal{P}} \left(\frac{k_C^*}{2m}\right)^2\right)}}.$$

Поред наведених, у литератури се могу пронаћи и друге фунцкије за мерење квалитета партиција. Више детаља може се наћи у [105, 106], као и у прегледним радовима [79, 80].

Глава 3

Метода променљивих околина за кластеровање на мрежи

У овом поглављу предложена је нова метода за кластеровање на комплексним мрежама максимизацијом модуларности. Метода је заснована на методи променљивих околина која је успешно примењена на велики број проблема оптимизације [107, 108]. Предложена метода је пажљиво имплементирана и унапређена у складу са актуелним истраживањима проблема максимизације модуларности и методе променљивих околина. Поглавље 3 је организовано кроз четири секције. У секцији 3.1 представљена је математичка формулација проблема максимизације модуларности. У секцији 3.2 описана је метода променљивих околина и њене варијанте, док је у секцији 3.3 представљена метода променљивих околина за максимизацију модуларности. У циљу ефикасне примене на комплексним мрежама великих димензија развијен је механизам за декомпозицију проблема на мање потпроблеме и побољшан механизам за превазилажење локалних максимума модуларности коришћењем критеријума за повремено прихватање лошијег решења од тренутно разматраног. У секцији 3.4 приказани су експериментални резултати добијени на DIMACS инстанцама. Добијени резултати су упоређени са најбољим резултатима презентованим у литератури за разматрани проблем, који су добијени двема методама развијеним у оквиру DIMACS позива „*Graph Partitioning and Graph Clustering*" 2012. године. Осим тога, добијени резултати су упоређени и са резултатима шест метода развијених након 2012. године које су се издвојиле у литератури. Резултати представљени у овој глави су такође приказани у раду [109].

3.1 Математичка формулација проблема максимизације модуларности

Нека је дат граф $G = (V, E)$ са n чворова и m грана, нека је $A = [a_{ij}]$ одговарајућа матрица суседства и c_{v_i} ознака кластера коме је придружен чвор v_i. За дату партицију $\mathcal{P}$ графа G, модуларност је дефинисана као:

$$MQ(\mathcal{P}) = \sum_{C \in \mathcal{P}} \left(\frac{m_C}{m} - \left(\frac{k_C^*}{2m} \right)^2 \right).$$

Број грана у кластеру C може се изразити коришћењем матрице суседства, Кронекер делта функције (видети секцију 2.3.1) и ознака кластера, на следећи начин

$$m_c = \frac{1}{2} \sum_{v_i, v_j \in C} a_{ij}, \quad \sum_{C \in \mathcal{P}} \frac{m_C}{m} = \frac{1}{2m} \sum_{C \in \mathcal{P}} \sum_{v_i, v_j \in C} a_{ij} = \frac{1}{2m} \sum_{v_i, v_j \in V} a_{ij}\, \delta(c_{v_i}, c_{v_j}).$$

Слично, вредност квадрата укупног степена свих чворова из C се може записати на следећи начин

$$(k_C^*)^2 = \left(\sum_{v_i \in C} k_{v_i} \right)^2 = \sum_{v_i, v_j \in C} k_{v_i}\, k_{v_j},$$

$$\sum_{C \in \mathcal{P}} \left(\frac{k_C^*}{2m} \right)^2 = \frac{1}{(2m)^2} \sum_{C \in \mathcal{P}} \sum_{v_i, v_j \in C} k_{v_i}\, k_{v_j} = \frac{1}{2m} \sum_{v_i, v_j \in V} \frac{k_{v_i}\, k_{v_j}}{2m}\, \delta(c_{v_i}, c_{v_j}).$$

Заменом у дефиницију MQ функције добија се модуларност као сума вредности по свим гранама комплетног графа K_n:

$$MQ(\mathcal{P}) = \frac{1}{2m} \sum_{v_i, v_j \in V} \left(a_{ij} - \frac{k_{v_i}\, k_{v_j}}{2m} \right) \delta(c_{v_i}, c_{v_j}).$$

Како је K_n комплетан граф (клика), сваки његов индуковани подграф је такође комплетан, тако да се кластеровање може формулисати као проблем партиционисања клике (енгл. *clique partitioning problem*).

Посматрајмо бинарну релацију на скупу чворова

$$(\forall v_i, v_j \in V)\ v_i \sim v_j \Leftrightarrow \text{чворови } v_i \text{ и } v_j \text{ припадају истом кластеру.}$$

Овако дефинисана релација $\sim$ је релација еквиваленције и може се интерпретирати бинарним променљивама:

$$x_{ij} = \begin{cases} 1, & \text{ако чворови } v_i \text{ и } v_j \text{ припадају истом кластеру,} \\ 0, & \text{иначе.} \end{cases}$$

Због рефлексивности имамо да је $\forall v_i \in V$, $x_{ii} = 1$, а самим тим и да је сума елемената на главној дијагонали за било коју партицију $\mathcal{P}$ константна и износи

$$\Omega = \sum_{v_i \in V} \frac{k_{v_i}^2}{4m^2}.$$

Због симетричности релације важи да је $\forall v_i, v_j \in V$, $x_{ij} = x_{ji}$, тако да се могу разматрати само промењљиве за $i < j$. Коначно, увођењем смене

$$w_{ij} = \frac{1}{m}\left(a_{ij} - \frac{k_{v_i}\, k_{v_j}}{2m}\right)$$

проблем максимизације модуларности се може формулисати као проблем бинарног целобројног програмирања:

$$\max \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij}\, x_{ij} - \Omega \tag{3.1}$$

при условима:

$$x_{ij} + x_{jk} - x_{ik} \leq 1, \qquad i, j, k = 1 \ldots n, \quad i < j < k, \tag{3.2}$$

$$x_{ij} - x_{jk} + x_{ik} \leq 1, \qquad i, j, k = 1 \ldots n, \quad i < j < k, \tag{3.3}$$

$$-x_{ij} + x_{jk} + x_{ik} \leq 1, \qquad i, j, k = 1 \ldots n, \quad i < j < k, \tag{3.4}$$

$$x_{ij} \in \{0, 1\}, \qquad i, j = 1 \ldots n, \quad i < j. \tag{3.5}$$

Ограничења (3.2) – (3.4) изражавају транзитивност уведене релације. Ако су чворови v_i и v_j у истом кластеру и чворови v_j и v_k у истом кластеру, тада и чворови v_i и v_k морају бити у истом кластеру. Наведена формулација садржи $\frac{n(n-1)}{2} = O(n^2)$ бинарних промењљивих (ограничење 3.5) и $\frac{3n(n-1)(n-2)}{2} = O(n^3)$ ограничења, а у литератури се могу пронаћи и компактније формулације [110, 111] са мањим бројем ограничења ($O(n^2)$), као и друге еквивалентне формулације [82, 112] које не захтевају увођење комплетног графа K_n, већ користе оригинални граф G.

3.2 Метода променљивих околина

Метода променљивих околина (енгл. *Variable Neighborhood Search* – VNS) је метахеуристика вођена једним решењем. Идеју VNS методе изложио је Младеновић 1995. године на симпозијуму „Optimization Days" у Монтреалу [113]. Две године касније, у раду [28], Младеновић и Хансен су детаљно описали VNS методу и применили за решавање проблема трговачког путника. До данас, VNS метода и њене варијанте су успешно примењене за решавање великог броја различитих проблема комбинаторне и глобалне оптимизације [114, 115, 116, 117, 118, 119].

Аутори VNS методе у раду [28] су кренули од чињеница да локални минимум у једној околини не мора бити локални минимум у другој околини, као и да је глобални минимум уједно и локални минимум у свим околинама. Осим тога, примећено је да су у реалним проблемима локални минимуми често релативно блиски, односно да локални минимум често пружа неке информације о глобалном минимуму. На пример, ако су решења неког реалног проблема кодирана низом атрибута, тада се може очекивати да два локална минимума имају значајан број атрибута са истим вредностима. Због наведених чињеница и емпиријског разматрања аутори предлажу систематску промену околина приликом претраживања простора решења. Различите околине $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_{k_{max}}$ могу бити дефинисане коришћењем различитих метрика или променом растојања у оквиру једне метрике тако да најчешће важи $|\mathcal{N}_1(x)| < |\mathcal{N}_2(x)| < \cdots < |\mathcal{N}_{k_{max}}|$. Параметар који одређује актуелну околину означава се са k и узима вредност од 1 до k_{max}.

Процес претраге се води коришћењем једног допустивог решења које се назива текуће решење x. Почетно решење (прво текуће решење) може бити унапред дато или се генерише на случајан начин. У свакој итерацији из околине $N_k(x)$ бира се једно случајно решење у фази размрдавања (енгл. *shaking phase*) које постаје полазно решење за локално претраживање. Након проналаска локалног минимума врши се упоређивање са текућим решењем. Ако је решење које одговара локалном минимуму боље од текућег решења, оно постаје ново текуће решење. У супротном, врши се промена околине из које се поново случајно бира решење за почетак локалне претраге. Када се поправи текуће решење, цео поступак се наставља коришћењем прве дефинисане околине. Псеудокод основне верзије VNS методе приказан је алгоритмом 3.2.

Алгоритам 3.2: Основна метода променљивих околина

улазни подаци: низ околина $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_{k_{max}}$

излазни подаци: решење x

```
1  x ← СлучајноРешење()
2  k ← 1
3  while услов заустављања do
4      x' ← Размрдавање(x, N_k)
5      x'' ← ЛокалнаПретрага(x')
6      if f(x'') < f(x) then
7          x ← x''
8          k ← 1
9      else
10         if k = k_max then
11             k ← 1
12         else
13             k ← k + 1
```

3.2.1 Варијанте методе променљивих околина

Уколико се из основне верзије VNS методе изостави фаза размрдавања, а локална претрага модификује тако да може да врши претраживање у свакој од околина $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_{k_{max}}$, добија се метода променљивог спуста (енгл. *Variable Neighborhood Descent* – VND). У овом случају, након генерисања почетног решења x, односно првог текућег решења, локална претрага се започиње од околине $\mathcal{N}_1(x)$. Уколико се пронађе боље решење x', оно постаје ново текуће и поново се врши претрага прве околине новог текућег решења. Уколико нема бољег решења у првој околини текућег решења, започиње се претрага друге околине $\mathcal{N}_2(x)$. Када се у било којој околини $\mathcal{N}_k(x)$ пронађе боље решење од текућег, цео поступак поново креће од прве околине. Дакле, крајњи резултат VND методе представља решење које је локални минимум у односу на све околине $\mathcal{N}_k$ $(k = 1, \ldots, k_{max})$. Повећавањем броја околина већа је шанса за проналазак глобалног минимума, али се повећава и време извршавања. Псеудокод VND методе приказан је алгоритмом 3.3.

Алгоритам 3.3: Метода променљивог спуста

улазни подаци: низ околина $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_{k_{max}}$

излазни подаци: решење x

```
1 x ← СлучајноРешење()
2 k ← 1
3 while k ≤ k_max do
4 |   x' ← ЛокалнаПретрага(x, N_k)
5 |   if f(x') < f(x) then
6 |   |   x ← x'
7 |   |   k ← 1
8 |   else
9 |   |   k ← k + 1
```

Са друге стране, уколико се из основне методе променљивих околина изостави локална претрага, која некад може да буде временски захтевна, добија се редукована метода променљивих околина (енгл. *Reduced Variable Neighborhood Search* – RVNS). Код ове варијанте VNS методе, претрага околина се врши бирањем случајног решења из околине текућег. За разлику од случајне претраге простора решења, RVNS метода је систематичнија и контролисана избором околина $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_{k_{max}}$. Због брзине извршавања редукована метода променљивих околина налази примену код инстанци веома великих димензија као и за добијање почетног решења за неку другу методу. Псеудокод RVNS методе приказан је алгоритмом 3.4.

У литератури постоји још неколико варијанти методе променљивих околина. На пример, често се локална претрага у основној VNS методи мења VND методом или неком другом хеуристиком и тако се добија општа метода променљивих околина [108] (енгл. *General Variable Neighborhood Search* – GVNS). Променом услова за прелазак у ново текуће решење тако да се не прихвата само боље решење, већ и решење лошијег квалитета, које је довољно далеко од текућег решења, добија се адаптивна метода променљивих околина [108] (енгл. *Skewed Variable Neighborhood Search* – SVNS). У односу на основну VNS методу неопходно је још дефинисати функцију за мерење удаљености између два решења $\mu(x_1, x_2)$, као и праг довољне удаљености λ након чега се услов преласка у ново текуће решење, $f(x'') \leq f(x)$, из алгоритма 3.2 мења условом $f(x'') - \lambda\,\mu(x, x'') < f(x)$.

Алгоритам 3.4: Редукована метода променљивих околина

улазни подаци: низ околина $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_{k_{max}}$

излазни подаци: решење x

```
1  x ← СлучајноРешење()
2  k ← 1
3  while услов заустављања do
4      x' ← Размрдавање(x, N_k)
5      if f(x') < f(x) then
6          x ← x'
7          k ← 1
8      else
9          if k = k_max then
10             k ← 1
11         else
12             k ← k + 1
```

Код методе променљивих околина са декомпозицијом [120] (енгл. *Variable Neighborhood Decomposition Search* – VNDS) у свакој итерацији врши се декомпозиција проблема, односно простора допустивих решења, који се затим разматра при локалном претраживању. VNDS налази примену код тешких проблема оптимизације у којима је честа локална претрага у комплетном простору решења неефикасна. Ипак, уколико постоје услови пожељно је повремено извршити локалну претрагу на комплетном простору јер се након поправке одређених компоненти решења мења и њихов утицај на решење у целини. Овај ефекат, назван гранични ефекат (енгл. *boundary effect*), примећен је у истраживањима [121, 122, 123].

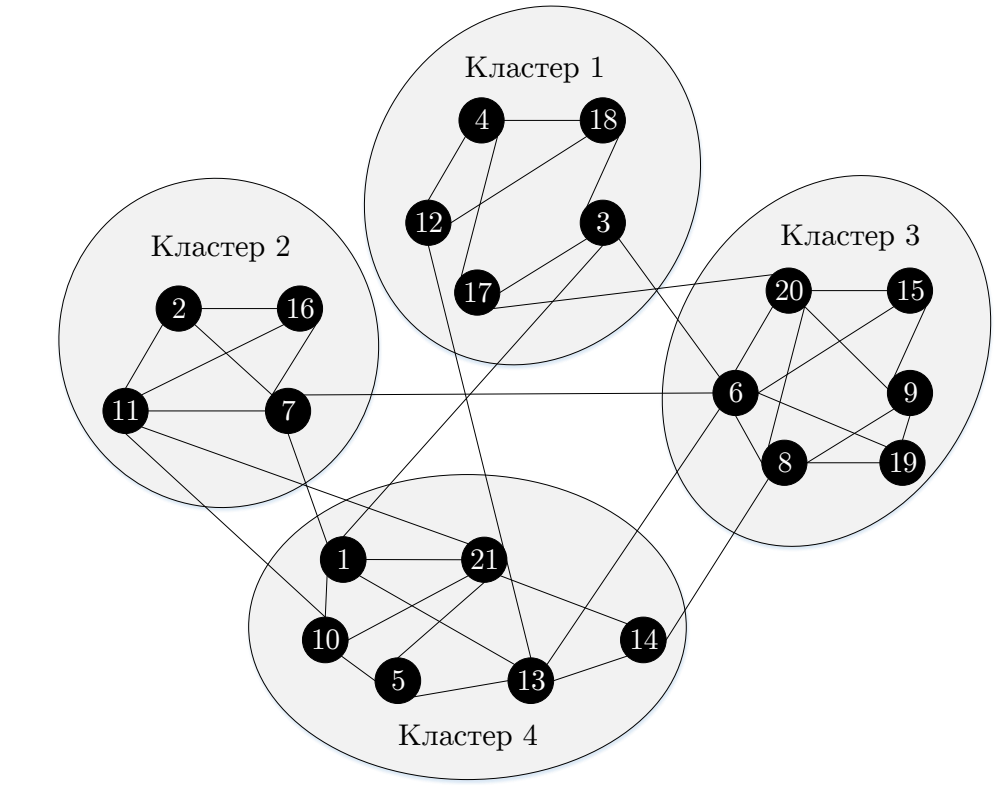
3.3 Метода променљивих околина за максимизацију модуларности

У наставку је описана нова адаптивна метода променљивих околина (енгл. *Ascent-Descent Variable Neighborhood Decomposition Search* – ADVNDS) развијена за максимизацију модуларности, односно кластеровање мреже. ADVNDS метода је заснована на методи променљивих околина. У циљу ефикасне примене за кластеровање на комплексним мрежама великих димензија ADVNDS

метода има додатни механизам за декомпозицију проблема на мање потпроблеме и унапређени механизам за превазилажење локалних максимума модуларности коришћењем критеријума за повремено прихватање лошијег решења од тренутно разматраног. Све компоненте предложене ADVNDS методе описане су кроз подсекције у наставку.

3.3.1 Кодирање решења

Нека је G граф са n чворова. Партиција графа G на дисјунктне кластере, односно једно решење, представља се низом x дужине n, тако да целобројна вредност на позицији i представља ознаку кластера коме припада чвор v_i. Пример кодирања једне партиције приказан је на слици 3.1.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
x	4	2	1	1	4	3	2	3	3	4	2	1	4	4	3	2	1	1	3	3	4

Слика 3.1: Кодирање решења

3.3.2 Околине решења

У фази размрдавања користе се четири типа околине:

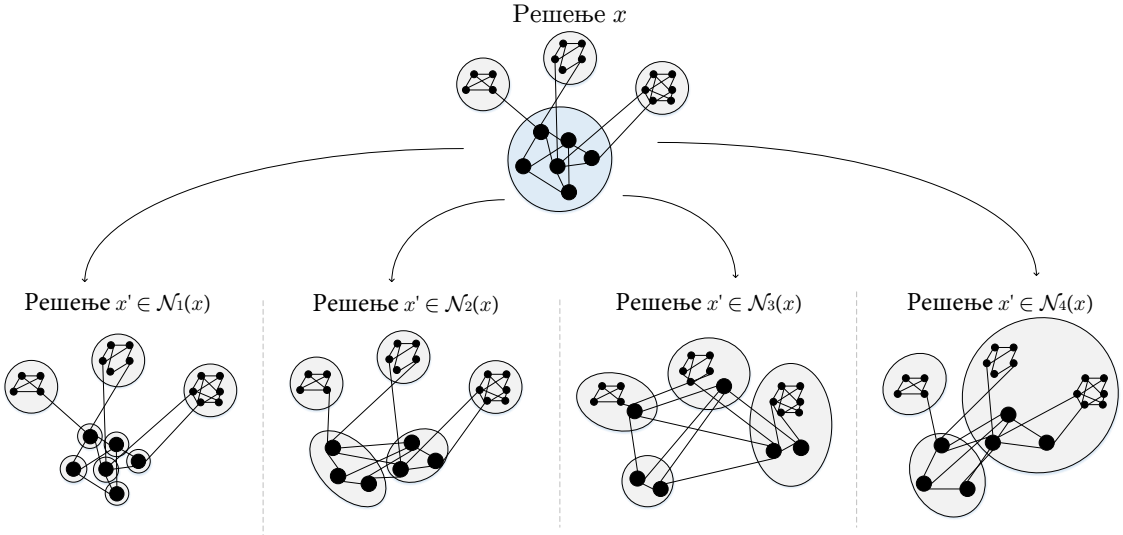
1. $\mathcal{N}_1(x)$ – садржи сва решења која се добијају трансформацијом решења x тако што чворови једног кластера постају једночлани кластери.

2. $\mathcal{N}_2(x)$ – садржи сва решења која се добијају поделом једног кластера из решења x у два кластера C_1 и C_2 таква да $||C_1| - |C_2|| \leq 1$.

3. $\mathcal{N}_3(x)$ – садржи сва решења која се добијају трансформацијом решења x тако да се чворови једног кластера придружују суседним чворовима или остају у истом кластеру.

4. $\mathcal{N}_4(x)$ – садржи сва решења која се добијају спајањем два или више кластера у решењу $x' \in \mathcal{N}_2(x)$.

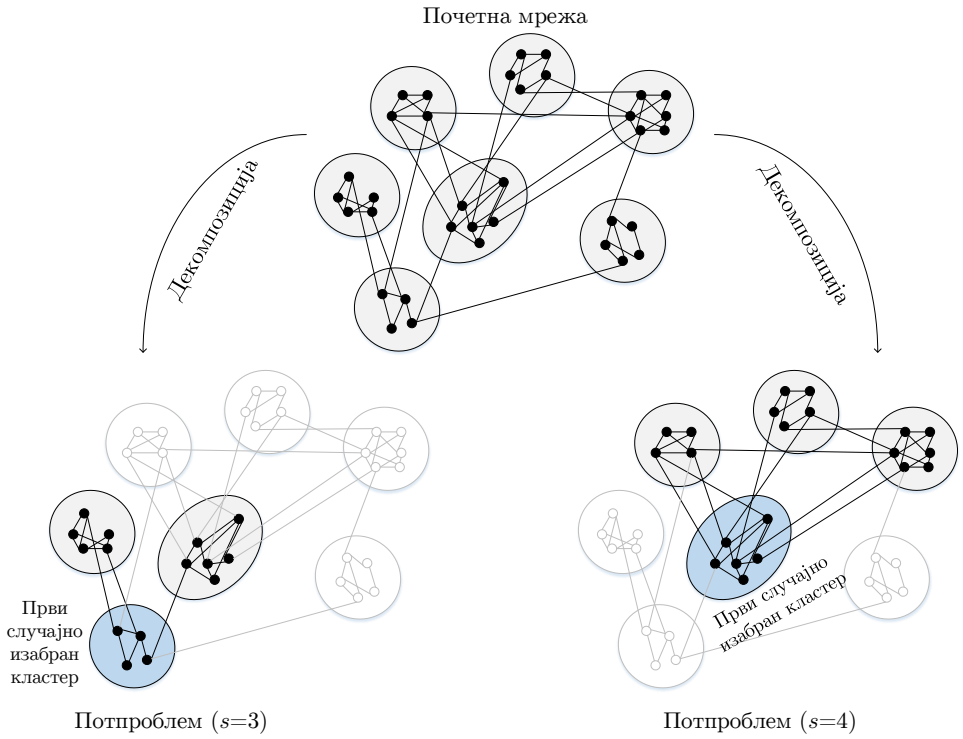
Илустрације решења из различитих околина приказане су на слици 3.2. Околине $\mathcal{N}_1, \mathcal{N}_2$ и $\mathcal{N}_3$ показале су се као веома добар избор у [83], а такође омогућавају једноставно ажурирање функције циља MQ израчунавањем нових вредности модуларности само за кластере у којима су се десиле промене. Околина $\mathcal{N}_4$ је инспирисана запажањем у раду [86], да се спајањем кластера могу превазићи локални максимуми модуларности. Трансформација којом је одређена ова околина представља композицију трансформације коришћене за околину $\mathcal{N}_2$ и трансформације спајања два или више кластера.



Слика 3.2: Примери решења која се налазе у дефинисаним околинама

3.3.3 Декомпозиција проблема

Приликом кластеровања комплексних мрежа већих димензија простор претраге може бити изузетно велики, а проналажење квалитетне партиције веома споро. Из тог разлога полазни проблем ће бити декомпонован на проблеме (графове) мањих димензија из којих се могу добити квалитетне компоненте за решење полазног проблема. Декомпозицији се мора приступити пажљиво како би имала ефекта, односно како би решење потпроблема заиста могло да побољша одређене компоненте полазног решења. На пример, решење добијено на подграфу конструисаном случајним избором чворова и грана неће моћи да побољша компоненте решења полазног проблема јер изостављени чворови и гране могу да имају значајан утицај на добијене кластере. Како би изостављени чворови и гране имали што мањи утицај на добијене кластере у подграфу, приликом декомпозиције и избора чворова користе се информације из најбоље добијене партиције. Декомпозицијом полазног проблема на проблем величине s конструише се индуковани подграф полазног графа коришћењем чворова из s кластера у датој партицији x. Пример реализације процедуре декомпозиције приказан је на слици 3.3.



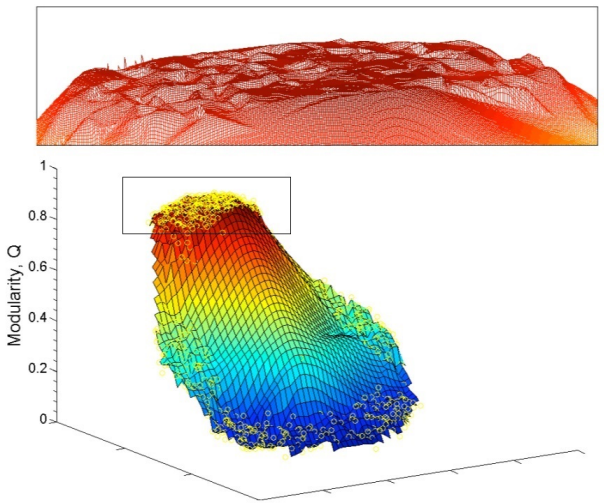
Слика 3.3: Пример декомпозиције за s=3 и s=4

Дакле, прво се бира један кластер из дате партиције на случајан начин, а након тога још $s-1$ суседних кластера, такође случајно. Затим, коришћењем скупа свих чворова из изабраних кластера конструише се индуковани подграф графа G, односно једна инстанца потпроблема.

3.3.4 Механизам за превазилажење локалних максимума

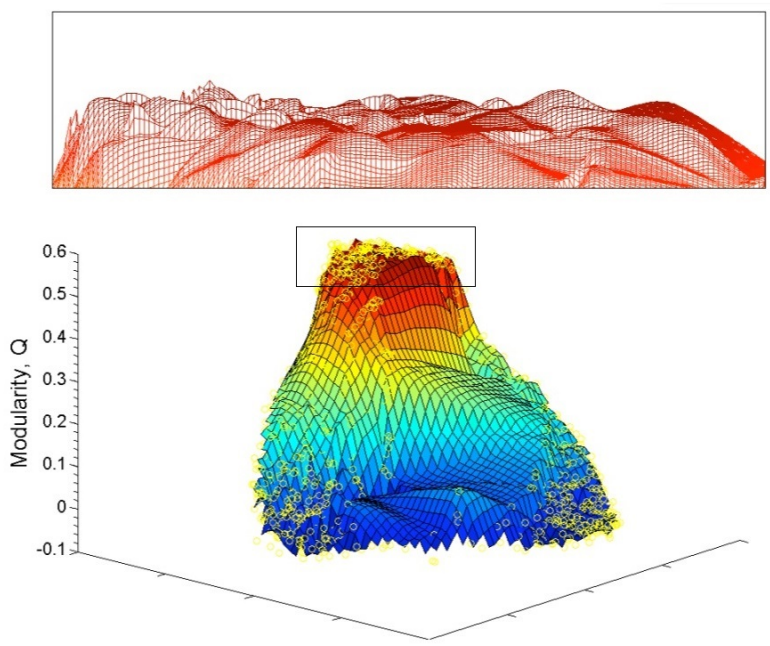
У фази размрдавања ADVNDS методе користе се претходно описане околине. Како околине нису угњеждене, оне се не смењују у датом поретку, већ се за сваку итерацију бира једна околина на случајан начин. Генерално, фаза размрдавања у основној верзији методе променљивих околина представља главни механизам за превазилажење локалних оптимума. У случају коришћења угњеждених околина (при чему је последња околина у низу довољно велика) механизам размрдавања омогућава ефикасно превазилажење локалних оптимума и диверсификацију претраге. У супротном, ефикасност размрдавања се смањује, процес претраге дуже траје, а може се завршити и у неком лошијем локалном оптимуму.

Са друге стране, анализом и визуализацијом модуларности [124] може се приметити да често постоји један плато, односно узвишени регион са јако великим бројем локалних максимума који су релативно блиски, а међу којима се налази и глобални максимум (слике 3.4 и 3.5).



Слика 3.4: Визулизација модуларности за граф $G(24,5)$ из секције 2.3.3 [124]

На слици 3.4 приказана је визуализација модуларности за граф $G(24, 5)$ из секције 2.3.3 на основу 997 партиција. На слици 3.5 приказана је визуализација за метаболичку мрежу „*Treponema pallidu*" на основу 1199 партиција. Изнад оба графика приказан је детаљније само узвишени регион. Поступак мапирања партиција на R^2 приказан је у раду [124].



Слика 3.5: Визулизација модуларности за метаболичку мрежу „*Treponema pallidu*" [124]

На основу приказаних визуализација јасно је да након достизања неког локалног максимума на платоу није неопходно разматрати сувише велику околину текућег решења, већ релативно блиска решења што управо омогућавају дефинисане околине. Како би било могуће ефикасно превазилажење локалних максимума, уведен је нови механизам односно AD критеријум (енгл. *ascent-descent criteria*) прихватања решења који дозвољава прелазак у лошије решење уколико претрага не напредује у неком периоду.

3.3.5 ADVNDS метода

Спајањем претходно описаних компоненти са VNS методом креирана је нова ADVNDS метода за максимизацију модуларности. Псеудокод ADVNDS методе приказан је алгоритмом 3.5.

Алгоритам 3.5: ADVNDS метода за кластеровање на мрежи

улазни подаци: Граф G, Параметри (a_0, r, s_{max}, t_{max})
излазни подаци: најбоље пронађено решење x^*

```
1  x ← ПочетноРешење()
2  x ← ЛокалнаПретрага(x, G, LPAm+)
3  x* ← x
4  s ← 1; a ← a_0
5  while t < t_max do
6      S ← Декомпозиција(x, s)
7      k ← СлучајанИзбор( {1, 2, 3, 4} )
8      x' ← Размрдавање(x, k, S)
9      x' ← ЛокалнаПретрага(x', S, LPAm+)
10     d ← MQ(x') − MQ(x)
11     if d ≥ 0 or e^{d/a} > СлучајанИзбор( [0, 1] ) then
12         x ← ЛокалнаПретрага(x', G, LPAm)
13         if MQ(x) > MQ(x*) then
14             x* ← x
15         s ← 1; a ← a_0
16     else
17         s ← s + 1; a ← a r
18         if s > min( БројКластера(x), s_max) then
19             s ← 1
20     t ← УтрошеноПроцесорскоВреме()
```

Почетно решење x у кораку 1 је генерисано на случајан начин. Затим се поправља применом локалне претраге у кораку 2 и поставља као најбоље решење x^* у кораку 3. Локална претрага као улазне параметре добија решење које покушава да поправи, простор претраге, односно граф који се разматра и алгоритам за пропагацију ознака (LPAm или LPAm+). Иницијализација параметара методе врши се у кораку 4: s – величина потпроблема који се креира у фази декомпозиције и a – параметар експоненцијалне расподеле који се користи приликом прихватања следећег решења. Главни део алгоритма представља итеративно понављање корака 5–19, док се не потроши дозвољено време t_{max}. У кораку 6 конструише се потпроблем S са s случајно изабраних

суседних кластера из партиције која одговара текућем решењу x. У кораку 7 врши се избор типа околине k која ће бити коришћена у фази размрдавања. Затим, у кораку 8 на текуће решење x примењује се фаза размрдавања користећи изабран тип околине у простору претраге који је одређен потпроблемом S, а добијено решење се чува као помоћно решење x'. Након фазе размрдавања, у кораку 9 локална претрага се примењује на помоћно решење x' алгоритмом LPAm+, али само у потпростору који је одређен потпроблемом S.

У кораку 10 израчунава се разлика између вредности модуларности помоћног и текућег решења. Услов у кораку 11 је од велике важности за методу јер се у њему доноси одлука да ли ће се применити локална претрага у целом простору решења. Стандардни услов $d \geq 0$ проширен је AD критеријумом који је имплементиран тако да прихвата лошије решење са вероватноћом $e^{d/a}$. AD услов се разматра само уколико је $d < 0$, а вероватноћа прихватања једног лошијег решења је већа што је то решење по квалитету ближе текућем решењу, односно што је d ближе нули. Са друге стране, вероватноћа се такође увећава са порастом параметра a кроз број неуспешних итерација. Уколико је решење боље од текућег или пролази AD критеријум, врши се локална претрага на целом простору коришћењем LPAm алгоритма у кораку 12. Овим кораком се отклања и евентуални гранични ефекат. Уколико је добијено решење након локалне претраге на целом простору боље од тренутно најбољег решења x^*, врши се његово ажурирање, а затим се свакако параметар декомпозиције s и вероватноће прихватања лошијег решења враћају на почетне вредности. У супротном, ако решење није квалитетније од текућег решења и не пролази AD критеријум, итерација се сматра неуспешном, а параметар вероватноће a се увећава множењем са датим фактором r (корак 17). Такође, увећава се и параметар декомпозиције s а уколико је достигао максималну вредност (s_{max} или број кластера у текућој партицији) поставља се на један (кораци 17-19).

На крају итерације у кораку 20 ажурира се утрошено процесорско време t. Након утрошеног времена, најбоље пронађено решење x^* представља резултат извршавања ADVNDS алгоритма.

3.4 Експериментални резултати

Предложена ADVNDS метода имплементирана је у програмском језику C++. Изворни код је компајлиран помоћу `g++` компајлера на Убунту оперативном систему. Сви експерименти су спроведени на рачунару са Intel Core i5-2400 - 3.10GHz процесором и 4 GB радне меморије.

За тестирање методе коришћене су DIMACS инстанце [1] сакупљене 2012. године у оквиру позива „*Graph Partitioning and Graph Clustering*". Скуп обухвата генерисане инстанце, као и инстанце комплексних мрежа из реалних примена. За потребе тестирања са различитим параметрима инстанце су подељене у две групе где прва група садржи инстанце са највише 500 чворова, а друга са више од 1000 чворова. Слична подела инстанци је коришћена у раду [83].

Максимално време извршавања алгоритма које се регулише параметром t_{max} је постављено на 180 секунди (3 минута) за инстанце из прве категорије и 3600 секунди (1 сат) за инстанце из друге категорије. Прелиминарна тестирања вредности осталих параметара показала су да алгоритам није превише осетљив на саме вредности тако да су за спровођење изабране вредности $a_{initial} = 0.001$, $r = 1.0001$ и $s_{max} = 15$. За сваку инстанцу алгоритам се извршава 10 пута што је у складу са устаљеном праксом из литературе [83, 96, 125, 126] за тестирања недетерминистичких алгоритама који решавају проблем кластеровања на комплексним мрежама.

У табели 3.1 приказани су резултати добијени у 10 независних извршавања ADVNDS алгоритма. Прва колона садржи назив инстанце, док друга и трећа колона описују њену величину: број чворова (n) и број грана (m). У четвртој колони приказана је просечна вредност MQ_{avg} функције из десет извршавања предложеног ADVNDS алгоритма, док је у петој колони дато и просечно време утрошено за добијање најбољег решења. Шеста колона представља просечан број прихватања лошијег решења w_{avg}. Седма и осма колона представљају максималну вредност модуларности Q_{best} из десет извршавања, односно број кластера у одговарајућој партицији $|\mathcal{P}_{best}|$. Последња колона приказује процентуалну вредност стандардне девијације модуларности у десет извршавања. У табели су истакнуте оптималне вредности модуларности и одговарајући број кластера.

[1] Све инстанце су јавно доступне и могу се могу преузети на адреси `http://www.cc.gatech.edu/dimacs10/downloads.shtml`.

Табела 3.1: Просечне и најбоље вредности модуларности добијене у 10 извршавања ADVNDS алгоритма

Инстанца	n	m	MQ_{avg}	t_{avg}	w_{avg}	MQ_{best}	$\lvert\mathcal{P}_{best}\rvert$	$\%\sigma(MQ)$
karate	34	78	0.41978961	0.25	0	**0.41978961**	**4**	0.000
chesapeake	39	170	0.26579585	0.33	0	**0.26579585**	**3**	0.000
dolphin	62	159	0.52851944	0.00	0	**0.52851944**	**5**	0.000
lesmis	77	254	0.56668798	0.00	0	**0.56668798**	**6**	0.000
polbooks	105	441	0.52723659	0.00	0	**0.52723659**	**5**	0.000
adjnoun	112	425	0.31336747	0.25	0	**0.31336747**	**7**	0.000
football	115	613	0.60456956	1.08	0	**0.60456956**	**10**	0.000
jazz	198	2742	0.44514385	3.05	0	**0.44514385**	**4**	0.000
celegansneural	297	2148	0.50378240	0.15	0	**0.50378240**	**6**	0.000
celegans_metabolic	453	2025	0.45324822	1.66	0	0.45324822	9	0.000
email	1133	5451	0.58282897	29.73	12	0.58282897	10	0.000
polblogs	1490	16715	0.42710511	0.12	0	0.42710511	278	0.000
netscience	1589	2742	0.95989999	0.12	0	0.95989999	407	0.000
power	4941	6594	0.94086519	1643.94	283	0.94097423	42	0.005
hep-th	8361	15751	0.85792731	1786.35	1495	0.85808847	1366	0.011
PGPgiantcompo	10680	24316	0.88639112	1800.00	119	0.88664690	103	0.016
cond-mat	16706	121251	0.85387268	1842.00	266	0.85414744	1244	0.020
astro-ph	16726	47594	0.74502039	2088.75	93	0.74524637	1051	0.013
as-22july06	22963	48436	0.67789771	1795.12	8	0.67838109	38	0.025
cond-mat-2003	31163	120029	0.77843785	3428.28	75	0.77924749	1673	0.055
cond-mat-2005	40421	175691	0.74607478	3412.24	11	0.74718099	1903	0.054
krong500slogn16	65536	2456071	0.06554011	1761.05	7	0.06566119	10222	0.019

Предложени ADVNDS алгоритам проналази оптимална решења за све инстанце из прве категорије за која су оптимална решења позната у литератури [83]. За инстанцу „celegans_metabolic" оптимално решење још увек није познато. Осим тога, на 9 од 13 инстанци из друге категорије ADVNDS алгоритам је поправио најбоља позната решења [2] из литературе [127].

Детаљније поређење са алгоритмима развијеним у оквиру DIMACS позива дато је у табели 3.2. У поређење је, осим претходне верзије VNDS алгоритма [83], укључен и CGGCi алгоритам (енгл. *Core Groups Graph Clustering Scheme* – CGGC) инспирисан учењем ансамбла (енгл. *ensemble learning*) [128]. Главна идеја је да се након добијања различитих решења просечног квалитета брзим алгоритмима пронађе максимално преклапање у њима и користи у даљој претрази за квалитетним решењем. Ова два алгоритма дала су најбоље

[2] Резултати са DIMACS такмичења могу се преузети на адреси: http://www.cc.gatech.edu/dimacs10/results/Modularity_Quality/

резултате у оквиру DIMACS позива „Graph Partitioning and Graph Clustering" [127].

Прва колона у табели 3.2 садржи назив инстанце. Друга, четврта и шеста колона садрже највеће вредности модуларности MQ_{best}, добијене редом са CGGCi, VNDS и ADVNDS алгоритмима на свакој инстанци. Трећа, пета и седма колона представљају позицију на ранг листи која разматра 15 алгоритама са такмичења.

Табела 3.2: Поређење ADVNDS алгоритма са алгоритмима развијеним у оквиру DIMACS позива

Инстанца	**CGGCi**		**VNDS**		**ADVNDS**	
	MQ_{best}	**Ранг**	MQ_{best}	**Ранг**	MQ_{best}	**Ранг**
celegans_metabolic	0.45214778	#2	0.45324822	#1	0.45324822	#1
as-22july06	0.67826683	#2	0.67757548	#3	0.67838109	#1
astro-ph	0.74384869	#3	0.74462084	#2	0.74524637	#1
cond-mat-2003	0.77868235	#2	0.77671700	#3	0.77924749	#1
cond-mat-2005	0.74625408	#2	0.74506494	#3	0.74718099	#1
cond-mat	0.85309726	#3	0.85340200	#2	0.85414744	#1
e-mail	0.58187658	#2	0.58282897	#1	0.58282897	#1
hep-th	0.85655364	#3	0.85769200	#2	0.85808847	#1
netscience	0.95989999	#1	0.95989999	#1	0.95989999	#1
PGPgiantcompo	0.88656423	#2	0.88608182	#3	0.88664690	#1
polblogs	0.42709696	#2	0.42710511	#1	0.42710511	#1
power	0.94033178	#3	0.94085057	#2	0.94097423	#1
krong500slogn16	0.06372656	#4	0.06505580	#2	0.06566119	#1

Као што се може видети из табеле 3.2 ADVNDS алгоритам је рангиран као први на свих 13 инстанци. На девет инстанци ADVNDS даје значајно квалитетније решење, док на четири инстанце даје решење истог квалитета као VNDS алгоритам. Само на једној инстанци (netscience) сва три алгоритма дају решење истог квалитета. У поређењу VNDS и CGGCi алгоритма, VNDS даје квалитетнија решења од CGGCi на 8 од 13 инстанци.

3.4.1 Ефекти модификација

У односу на раније предложен VNDS алгоритам [83], ADVNDS алгоритам уводи прихватање лошијег решења и нову структуру околина у циљу ефикаснијег превазилажења локалних максимума. Може се поставити питање која је од ових модификација значајнија. Из тог разлога спроведена је додатна експериментална анализа у којој су упоређени ефекти сваке од модификација кроз четири алгоритма, и то VNDS без модификација са три околине (VNDS-3), VNDS са новом (четвртом) околином (VNDS-4) и одговарајуће верзије са AD правилом за прихватање лошијег решења (ADVNDS-3, ADVNDS-4). Резултати поређења приказани су у табели 3.3.

За сваку верзију алгоритма и сваку инстанцу у табели 3.3 приказано је процентуално одступање модуларности добијених решења $\%\sigma(MQ)$ од модуларности најбољег познатог решења MQ_{best}. Поред тога, за верзије алгоритама са новом околином приказан је ефективан број коришћења нове (четврте) околине. На пример, 6/383 означава да је у 383 побољшања текућег решења након фазе размрдавања четврта околина била заслужна у 6 случајева.

Табела 3.3: Поређење различитих верзија VNDS и ADVNDS алгоритма

Инстанца	**3 околине**		**4 околине**				MQ_{best}
	VNDS-3	**ADVNDS-3**	**VNDS-4**		**ADVNDS-4**		
	$\%\sigma(MQ)$		$\%\sigma(MQ)$	#4	$\%\sigma(MQ)$	#4	
as-22july06	0.17%	0.17%	0.19%	4/144	0.19%	4/144	0.6784
astro-ph	0.24%	0.24%	0.13%	6/383	0.11%	6/383	0.7452
cond-mat-2003	0.47%	0.47%	0.49%	13/512	0.49%	13/512	0.7792
cond-mat-2005	0,55%	0.55%	0.45%	27/530	0.45%	27/530	0.7472
cond-mat	0.09%	0.05%	0.11%	6/279	0.07%	5/304	0.8541
hep-th	0.06%	0.02%	0.08%	2/120	0.03%	22/270	0.8581
PGPgiantcompo	0,10%	0.07%	0.07%	3/107	0.06%	2/151	0.8866
power	0.03%	0.01%	0.01%	0/92	0.01%	8/255	0.9410
Просек:	*0.21%*	*0.20%*	*0.19%*	*2.32%*	*0.18%*	*3.28%*	

Из табеле 3.3 може се приметити да обе модификације имају позитиван утицај на квалитет крајњег решења. У погледу четврте околине примећује се да је VNDS-4 алгоритам у просеку мало бољи од VNDS-3 алгоритма (због мањег одступања 0.19% у односу на 0.21%). Такође, ADVNDS-4 је бољи од ADVNDS-3 алгоритма (одступање 0.18% наспрам 0.20%). Ефекти AD правила видљиви су из поређења VNDS-3 са ADVNDS-3 и VNDS-4 са ADVNDS-4 алгоритмом. У оба поређења у просеку је боља верзија са AD правилом за прихватање лошијег решења. Из табеле 3.3 се такође може приметити да четврта околина има већи утицај када се користи са AD правилом (у просеку 3.28% успешности), него без овог правила (у просеку 2.32% успешности).

3.4.2 Поређења са осталим хеуристикама из литературе

Како је проблем кластеровања веома актуелан, након 2012. године предложено је неколико различитих метода кластеровања заснованих на оптимизацији модуларности, међу којима су: LPA-CNP (енгл. *Label Propagation with weighted Coherent Neighborhood Propinquity*) предложен у раду [129], MMCD (енгл. *Multi-objective Memetic Community Detection*) предложен у [126] и COMBO предложен у [130]. Осим ових метода, у поређење су укључене и општије методе које омогућавају добијање расплинутих кластера[3]: FC (енгл. *Fuzzy Clustering*) предложен у раду [131], GAFCD (енгл. *Genetic Algorithm for Fuzzy Community Detection*) предложен у [132] и MDP (енгл. *Membership-Degree Propagation*) предложен у [125]. Ове методе се могу користити за добијање дисјунктних кластера једноставним редефинисањем финалне партиције тако што се сваки чвор смести у кластер коме припада са највећом вероватноћом (видети [125]).

У табели 3.4 представљано је поређење на осам инстанци за које су доступни резултати већине наведених хеуристика. Прва колона садржи назив инстанце. Кроз остале колоне за сваку хеуристику приказана је највећа вредност модуларности добијене партиције са дисјунктним кластерима. Истакнуте вредности означавају максималну вредност модуларности за одговарајућу инстанцу. Резултати који се не могу пронаћи у литератури означени су са „-”.

[3]За кластере кажемо да су расплинути ако нису дисјунктни већ се преклапају и у том случају пожељно је да сваки чвор има вероватноћу припадања сваком од кластера.

Табела 3.4: Поређење ADVNDS алгоритма са последње предложеним алгоритмима у литератури

Инстанца	**Алгоритам**						
	MDP	COMBO	LPA-CNP	FC	GAFCD	MMCD	ADVNDS
karate	**0.4198**	**0.4198**	0.3718	0.3573	**0.4198**	**0.4198**	**0.4198**
dolphin	0.5265	0.5268	0.4833	0.4602	**0.5285**	**0.5285**	**0.5285**
lesmis	0.5619	0.5619	0.3916	0.4308	0.5619	0.5600	**0.5667**
polbook	0.5269	**0.5272**	0.4600	0.4787	**0.5272**	**0.5272**	**0.5272**
footbal	**0.6046**	**0.6046**	0.6007	0.5793	**0.6046**	**0.6046**	**0.6046**
e-mail	0.5766	0.5815	0.0000	0.3657	0.5729	0.5749	**0.5828**
power	0.9389	0.9384	0.8633	0.7450	0.8453	0.9394	**0.9410**
PGPgiantcompo	0.8832	0.8795	0.7400	-	-	0.8270	**0.8866**

Табела 3.4 показује да ADVNDS алгоритам даје најквалитетније партиције у погледу модуларности на свим инстанцама што још једном потврђује његову ефикасност за решавање проблема кластеровања максимизацијом модуларности.

Глава 4

E-функција за кластеровање на мрежи

У овом поглављу предложена је нова функција за мерење квалитета партиције чијом се оптимизацијом могу идентификовати кластери у комплексној мрежи. Мотивацију за овај правац истраживања представља анализа дата у секцији 2.3.3 која приказује ограничења и недостатке функције модуларности. Истраживање функција за мерење квалитета партиције је веома актуелно (видети секцију 2.3.4), јер ове функције омогућавају формулисање кластеровања као проблема комбинаторне оптимизације. Четврто поглавље је организовано кроз три секције. У секцији 4.1 дефинисана је E-функција, анализирана су њена својства и приказан потенцијал за идентификовање кластера у мрежи. У секцији 4.2 предложена је генеричка метода променљивих околина за кластеровање оптимизацијом дате функције. У секцији 4.3 су представљени и упоређени експериментални резултати добијени оптимизацијом E-функције и модуларности помоћу предложене методе и то на генерисаним и реалним инстанцама из литературе за које је исправна подела на кластере позната. Резултати представљени у овој глави су такође приказани у радовима [133, 134].

4.1 E-функција

Основна идеја E-функције је одређивање квалитета кластера C посматрајући његову унутрашњу структуру и однос према осталим кластерима. Унутрашња структура кластера је добро описана његовом густином. На пример, потпун подграф који представља идеалан кластер има густину један што је

уједно и максимална вредност. Међутим, мера унутрашњег квалитета кластера би требало да буде осетљива на структуру мреже. Тако кластеру густине d_C у ретској мреже треба придружити већу вредност унутрашњег квалитета, него кластеру густине d_C у густој мрежи. Из тог разлога, добра мера унутрашње структуре је разлика вредности густине кластера d_C и густине мреже d_G. На тај начин сваки подграф са два или више чворова који има већу густину од целе мреже представља потенцијални кластер. Овако дефинисана унутрашња мера квалитета не разликује потпун подграф са неколико чворова и потпун подграф са десетинама чворова у истој мрежи. Штавише, сваки пар чворова који је повезан граном представља идеалан кластер у контексту овако дефинисане мере. Овај проблем се може превазићи укључивањем информације о броју чворова подграфа, односно да број чворова множи разлику између наведених густина, тј. $n_C\,(d_C - d_G)$. Како би и мале промене у густини кластера или броју чворова у кластеру биле јасно уочљиве, можемо применити експоненцијалну функцију. Коначно, израз за вредност унутрашњег квалитета кластера C, у ознаци $EQ^+(C)$, је

$$EQ^+(C) = \begin{cases} e^{n_C\,(d_C - d_G)}, & n_C \neq 1, \\ 0, & n_C = 1. \end{cases}$$

Однос кластера C према другим кластерима у мрежи може се окарактерисати бројем спољних грана l_c, тј. бројем грана од чворова из кластера ка чворовима у другим кластерима. Прихватљив број спољних грана у великој мери зависи од контекста кластеровања и циља који се жели постићи. Из тог разлога, поред броја спољних грана, може се увести параметар r који регулише степен утицаја спољних грана на укупни квалитет кластера. Такође, важно је узети у обзир информацију о броју чворова у кластеру како би се уочила разлика између два кластера са истим бројем спољних грана, а различитим бројем чворова. Природно је да кластер са више чворова има и више спољних грана које га повезују са остатком мреже. Из тог разлога, вредност која мери однос кластера C према другим кластерима, у ознаци $EQ^-(C)$, зависи од броја спољних грана по чвору $\frac{2l_c}{n_c}$ и израчунава се на следећи начин

$$EQ^-(C) = e^{r\frac{2l_c}{n_c}}.$$

Разлика $EQ^+(C) - EQ^-(C)$ карактерише укупан Е-квалитет кластера C, док је укупан Е-квалитет партиције збир квалитета свих кластера у партицији,

тј.

$$EQ(\mathcal{P}) = \sum_{C\in\mathcal{P}} \left[EQ^+(C) - EQ^-(C)\right].$$

Под претпоставком да кластери у партицији $\mathcal{P}$ нису једночлани, Е-квалитет партиције је

$$EQ(\mathcal{P}) = \sum_{C\in\mathcal{P}} \left[e^{n_C\left(\frac{2m_C}{n_C(n_C-1)} - \frac{2m}{n(n-1)}\right)} - e^{r\frac{2l_C}{n_C}}\right].$$

4.1.1 Особине партиција добијених максимизацијом Е-функције

У овој подсекцији анализирана је Е-функција, односно њене могућности за идентификовање кластера у процесу максимизације. За доказивање одређених особина биће коришћена следећа лема 4.1.

Лема 4.1. Нека је $x \geq 3$ и $0 < y \leq \frac{1}{2}$. Тада важи неједнакост

$$e^{x(1-y)} - e^{\frac{4}{x}} > \frac{1}{2}\left(e^{x(1-2y)} - e^{\frac{2}{x}}\right).$$

Доказ. Нека је $f(x,y)$ функција дефинисана на домену $D_f = [3,+\infty)\times\left(0,\frac{1}{2}\right]$ као

$$f(x,y) = e^{x(1-y)} - \frac{1}{2}e^{x(1-2y)} - e^{\frac{4}{x}} + \frac{1}{2}e^{\frac{2}{x}}.$$

Парцијални извод функције $f(x,y)$ по промењливој x је

$$f'_x(x,y) = (1-y)\,e^{x(1-y)} - \left(\frac{1}{2}-y\right)e^{x(1-2y)} + \frac{4}{x^2}e^{\frac{4}{x}} - \frac{1}{x^2}e^{\frac{2}{x}},$$

а парцијални извод по промењливој y је

$$f'_y(x,y) = -xe^{x(1-2y)}(e^{xy}-1).$$

Како је

$$(1-y)\,e^{x(1-y)} > \left(\frac{1}{2}-y\right)e^{x(1-y)} \geq \left(\frac{1}{2}-y\right)e^{x(1-2y)}, \quad \text{за свако } (x,y)\in D_f,$$

и

$$\frac{4}{x^2}e^{\frac{4}{x}} > \frac{1}{x^2}e^{\frac{4}{x}} > \frac{1}{x^2}e^{\frac{2}{x}}, \quad \text{за свако } (x,y)\in D_f,$$

закључује се да је $f'_x(x,y) > 0$, одакле следи да за фиксирано y, f је строго растућа функција за $x \in [3, +\infty)$. Дакле, $f(x,y) \geq f(3,y)$ за свако $(x,y) \in D_f$. Са друге стране, јасно је да је $f'_y(x,y) < 0$, па стога за фиксирано x, f је строго опадајућа функција за $y \in \left(0, \frac{1}{2}\right]$. Дакле, важи да је $f(3,y) \geq f\left(3, \frac{1}{2}\right)$ за свако $y \in \left(0, \frac{1}{2}\right]$ одакле је

$$f(x,y) \geq f\left(3, \frac{1}{2}\right) = \frac{1}{2}\left(e^{\frac{2}{3}} - 2e^{\frac{4}{3}} + 2e^{\frac{3}{2}} - 1\right) \approx 1.16 > 0, \quad \text{за свако } (x,y) \in D_f.$$

Коначно, како је $f(x,y) > 0$ на домену D_f, закључује се да је

$$e^{x(1-y)} - e^{\frac{4}{x}} > \frac{1}{2}\left(e^{x(1-2y)} - e^{\frac{2}{x}}\right), \quad \text{за } x \geq 3 \quad \text{и} \quad 0 < y \leq \frac{1}{2},$$

што је требало доказати. ■

Теорема 4.1. Партиција комплетног графа са n чворова ($n \geq 4$) добијена максимизацијом Е-функције садржи један кластер.

Доказ. Нека је $G = (V, E)$ комплетан граф са n ($n \geq 4$) чворова и $m = \frac{n(n-1)}{2}$ грана. Нека је $\mathcal{P}_1^n$ партиција формирана од једног кластера C_1^1 који садржи свих n чворова графа G. Тада је

$$EQ(\mathcal{P}_1^n) = e^{n\left(\frac{2\frac{n(n-1)}{2}}{n(n-1)} - \frac{2\frac{n(n-1)}{2}}{n(n-1)}\right)} - e^{\frac{2\cdot 0}{n}} = 1 - 1 = 0.$$

Нека је $\mathcal{P}_2^{n_1,n_2}$ партиција коју чине два кластера C_1^2 и C_2^2 који садрже n_1 и n_2 чворова, респективно. Број грана између кластера C_1^2 и C_2^2 је $n_1\, n_2$, док квалитет партиције $\mathcal{P}_2^{n_1,n_2}$ износи

$$\begin{aligned} EQ(\mathcal{P}_2^{n_1,n_2}) &= e^{n_1\left(\frac{2\frac{n_1(n_1-1)}{2}}{n_1(n_1-1)} - \frac{2\frac{n(n-1)}{2}}{n(n-1)}\right)} - e^{\frac{2n_1n_2}{n_1}} + e^{n_2\left(\frac{2\frac{n_2(n_2-1)}{2}}{n_2(n_2-1)} - \frac{2\frac{n(n-1)}{2}}{n(n-1)}\right)} - e^{\frac{2n_1n_2}{n_2}} \\ &= 1 - e^{2n_1} + 1 - e^{2n_2} \\ &= 2 - (e^{2n_1} + e^{2n_2}). \end{aligned} \tag{4.1}$$

Како је $\min\{e^{2n_1} + e^{2n_2} \mid n_1 + n_2 = 4\,, n_1 \geq 2, n_2 \geq 2\} = 2e^4$ и достиже се у тачки $(n_1, n_2) = (2,2)$, имамо да је

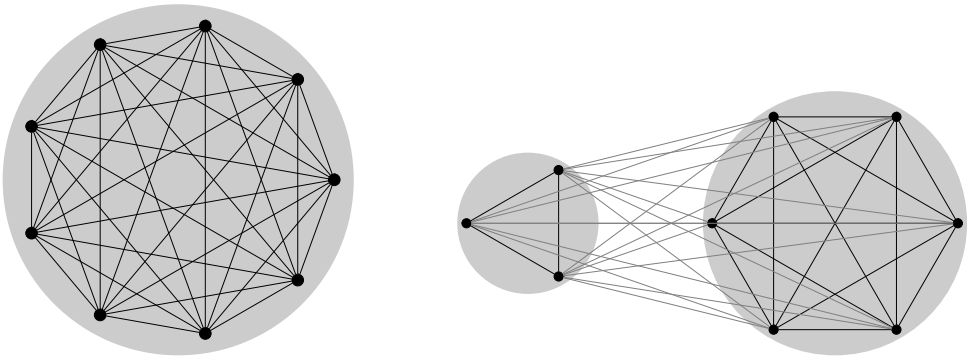
$$EQ(\mathcal{P}_2^{n_1,n_2}) = 2 - (e^{2n_1} + e^{2n_2}) < 2 - 2e^4 < EQ(\mathcal{P}_1^n),$$

одакле следи да максимизација EQ функције неће поделити комплетан граф G на два кластера. Две партиције комплетног графа са 9 чворова приказане су на слици 4.1.

Како је сваки кластер (индуковани подграф) комплетног графа и сам комплетан, свака подела графа G у k кластера може се добити полазећи од графа G као једног кластера и итеративном поделом на два нова кластера. Из тог разлога, јасно је да ће кроз сваку итерацију квалитет партиције по EQ функцији бити све мањи, тј. имамо да је

$$EQ(\mathcal{P}_k^{n_1,\ldots,n_k}) < EQ(\mathcal{P}_1^n), \quad \text{за свако} \quad 2 \leq k \leq \left\lfloor \frac{n}{2} \right\rfloor,$$

где је $\mathcal{P}_k^{n_1,\ldots,n_k}$ партиција графа која садржи k кластера $C_1^k, \ldots, C_k^k$, са $n_1, \ldots n_k$ чворова, респективно, $n_i \geq 2$, $(i = 1, \ldots k)$ и $\sum_{i=1}^{k} n_i = n$. Дакле, партиција добијена максимизацијом EQ функције у комплетном графу увек садржи један кластер. ■



$\mathcal{P}_1^9$ $\mathcal{P}_2^{3,6}$

Слика 4.1: Партиције $\mathcal{P}_1^9$ и $\mathcal{P}_2^{3,6}$ комплетног графа K_9

Теорема 4.2. Нека је G граф који се састоји од $l \geq 3$ клика $K_1, K_2, \ldots K_l$ од којих свака садржи s чворова ($s \geq 3$), а при томе свака клика K_i је повезана једном граном са кликом $K_{(i+1) \bmod l}$, за $i = 1, \ldots l$. Партиција добијена максимизацијом Е-функције у графу G не садржи кластер који се састоји од две клике спојене једном граном.

Доказ. Укупан број чворова у графу G је $n = s\,l$. Како су сваке две узастопне клике са s чворова повезане једном граном, укупан број грана у графу G је

$\frac{1}{2}s\,l\,(s-1)+l$. Нека је $\mathcal{P}_1$ партиција графа G, која се састоји од l кластера, где сваки кластер одговара једној клики. Без губитка општости претпоставимо да је l паран број и нека је $\mathcal{P}_2$ партиција графа G која се састоји од $\frac{l}{2}$ кластера, где сваки кластер одговара унији две узастопне клике. Партиције $\mathcal{P}_1$ и $\mathcal{P}_2$ графа који се састоји од 12 спојених клика K_6 приказане су на слици 4.2. Доказаћемо да је, према вредностима EQ функције, квалитет партиције $\mathcal{P}_1$ већи од квалитета партиције $\mathcal{P}_2$ у графу G.

На почетку анализираћемо густину графа G и густину кластера у партицијама $\mathcal{P}_1$ и $\mathcal{P}_2$. Густина графа G је $d_G = \frac{s^2-s+2}{s^2l-s}$. Доказаћемо да је

$$0 < d_G \leq \frac{1}{3}.$$

Нека је $l \geq 3$ фиксирано, а затим посматрајмо густину d графа G као функцију промењљиве s, односно $d : [3, +\infty) \longrightarrow (0, 1]$. Први извод функције d је $d'(s) = \frac{s^2(l-1)-4ls+2}{s^2(sl-1)^2}$. Како је $l - 1 > 0$, јасно је да је $d'(s) < 0$ за $s \in (s_1, s_2)$ и $d'(s) > 0$ за $s \in (-\infty, s_1) \cup (s_2, +\infty)$, где су s_1 и s_2 реална решења једначине $(l-1)s^2 - 4ls + 2 = 0$. Детаљнијом анализом саме квадратне функције добија се да је за $l \geq 3$ решење $s_1 \in \left(0, 3 - 2\sqrt{2}\right]$, а $s_2 \in \left(4, 3 + 2\sqrt{2}\right]$. Дакле, $d(s)$ је монотоно растућа функција на $(3, s_2)$, па на том интервалу важи

$$d(s) \leq d(3) = \frac{8}{9l-3} \leq \frac{1}{l}, \quad \text{за свако } l \geq 3.$$

Са друге стране, функција $d(s)$ је монотоно опадајућа на интервалу $(s_2, +\infty)$ па важи

$$d(s) < \lim_{s\to\infty} d(s) = \lim_{s\to\infty} \frac{s^2-s+2}{s^2l-s} = \frac{1}{l}.$$

Из претходна два разматрања следи да за $s \geq 3$ важи

$$0 < d(s) \leq \frac{1}{l}, \quad \text{за свако } \ l \geq 3,$$

одакле је

$$0 < d_G \leq \frac{1}{3}.$$

Како су сви кластери C_i^1 $(i = 1, \ldots, l)$ у партицији $\mathcal{P}_1$ комплетни подграфови важи

$$d_{C_i^1} = 1, \quad \text{за свако } i = 1, \ldots, l.$$

Са друге стране, густина кластера C_j^2 $(j = 1, \ldots, l/2)$ у партицији $\mathcal{P}_2$ износи $d_{C_j^2} = \frac{s^2-s+1}{2s^2-s}$, а једноставном анализом се утврђује да

$$0 < d_{C_j^2} < \frac{1}{2}, \quad \text{за свако } j = 1, \ldots, l/2.$$

Посматрајмо квалитет партиција $\mathcal{P}_1$ и $\mathcal{P}_2$

$$EQ(\mathcal{P}_1) = \sum_{i=1}^{l} \left[e^{s\left(d_{C_i^1} - d_G\right)} - e^{\frac{4}{s}} \right] = l \left(e^{s(1-d_G)} - e^{\frac{4}{s}} \right),$$

$$EQ(\mathcal{P}_2) = \sum_{j=1}^{l/2} \left[e^{2s\left(d_{C_j^2} - d_G\right)} - e^{\frac{2}{s}} \right] \leq \frac{l}{2} \left(e^{2s(\frac{1}{2} - d_G)} - e^{\frac{2}{s}} \right).$$

Применом леме 4.1 за $x = s$ и $y = d_G$, следи да је

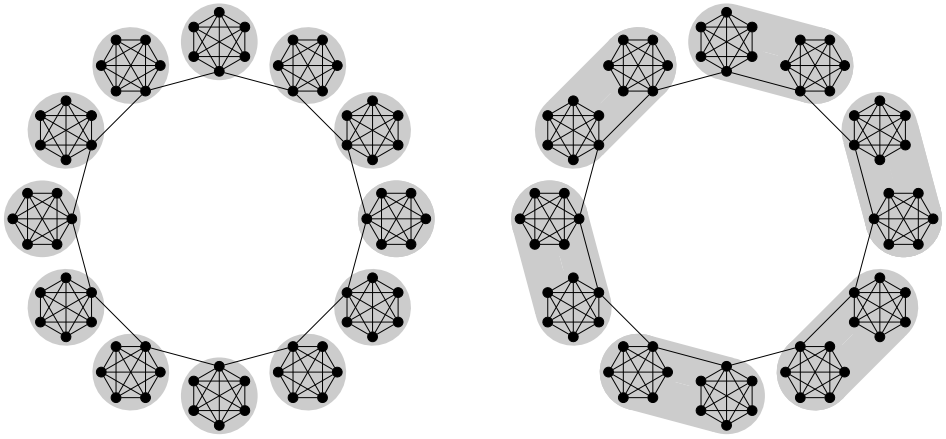
$$\left(e^{s(1-d_G)} - e^{\frac{4}{s}} \right) > \frac{1}{2} \left(e^{2s(\frac{1}{2} - d_G)} - e^{\frac{2}{s}} \right),$$

односно да је

$$EQ(\mathcal{P}_1) = l \left(e^{s(1-d_G)} - e^{\frac{4}{s}} \right) \geq \frac{l}{2} \left(e^{2s(\frac{1}{2} - d_G)} - e^{\frac{2}{s}} \right) \geq EQ(\mathcal{P}_2),$$

што је требало доказати. ■

Сличним разматрањем могуће је показати да је у контексту EQ функције партиција $\mathcal{P}_1$ квалитетнија од партиције $\mathcal{P}_k$, где је $\mathcal{P}_k$ партиција графа G која садржи $\frac{l}{k}$ кластера, добијених спајањем k узастопних клика.



Слика 4.2: Партиције $\mathcal{P}_1$ и $\mathcal{P}_2$ графа који се састоји од 12 спојених клика K_6

Теорема 4.3. Нека је G граф који се састоји од две клике K_s^1, K_s^2 са по s чворова ($s \geq 4$), две клике K_q^3, K_q^4 са по q чворова ($3 \leq q < s$) и додатне четири гране које повезују клике K_s^1 и K_s^2, K_s^2 и K_q^3, K_s^2 и K_q^4, K_q^3 и K_q^4. Пар-

тиција добијена максимизацијом Е-функције у графу G не садржи кластер који се састоји од спојених клика K_q^3 и K_q^4.

Доказ. Укупан број чворова у графу G је $n = 2q + 2s$, док је укупан број грана $m = q(q-1) + s(s-1) + 4$. Нека је $\mathcal{P}_1$ партиција графа G која се састоји од четири кластера који одговарају по једној клики. Нека је $\mathcal{P}_2$ партиција G која се састоји од три кластера, где прва два одговарају кликама са s чворова, а трећи одговара унији две клике са q чворова. Партиције $\mathcal{P}_1$ и $\mathcal{P}_2$ приказане су на примеру графа G за $s = 20$ и $q = 5$ на слици 4.3.

Показаћемо да је на графу G партиција $\mathcal{P}_1$ квалитетнија од партиције $\mathcal{P}_2$, у односу на EQ функцију.

Посматрајмо прво густину графа G, а затим и густине кластера у партицијама $\mathcal{P}_1$ и $\mathcal{P}_2$. Густина графа G је $d_G = \frac{q^2-q+s^2-s+4}{2q^2+4qs-q+2s^2-s}$. Јасно је да је $d_G > 0$, и претпоставимо да је $d_G < \frac{1}{2}$. У том случају важи

$$\frac{q^2-q+s^2-s+4}{2q^2+4qs-q+2s^2-s} < \frac{1}{2} \quad \Leftrightarrow \quad \frac{qs+\frac{q}{4}+\frac{s}{4}-2}{(q+s-\frac{1}{2})(q+s)} > 0$$

што је испуњено за свако $q \geq 4$ и $3 \leq s < q$.

Дакле,

$$0 < d_G \leq \frac{1}{2}.$$

Како су сви кластери у партицији $\mathcal{P}_1$ и два кластера у партицији $\mathcal{P}_2$ комплетни подграфови, важи да је $d_{C_i^1} = 1$, за $i = 1, 2, 3, 4$ и $d_{C_j^2} = 1$, за $j = 1, 2$. Густина трећег кластера у партицији $\mathcal{P}_2$ је $d_{C_3^2} = \frac{s^2-s+1}{2s^2-s}$, и једноставно се доказује да је $0 < d_{C_3^2} < \frac{1}{2}$.

Вредности EQ функције за партиције $\mathcal{P}_1$ и $\mathcal{P}_2$ износе:

$$EQ(\mathcal{P}_1) = 2e^{q(1-d_G)} - e^{\frac{2}{q}} - e^{\frac{6}{q}} + 2e^{s(1-d_G)} - 2e^{\frac{4}{s}},$$

$$EQ(\mathcal{P}_2) = 2e^{q(1-d_G)} - e^{\frac{2}{q}} - e^{\frac{6}{q}} + e^{2s\left(d_{C_3^2}-d_G\right)} - e^{\frac{2}{s}}.$$

Посматрајмо разлику $EQ(\mathcal{P}_1) - EQ(\mathcal{P}_2)$. Како је $0 < d_{C_3^2} < \frac{1}{2}$, важи да је

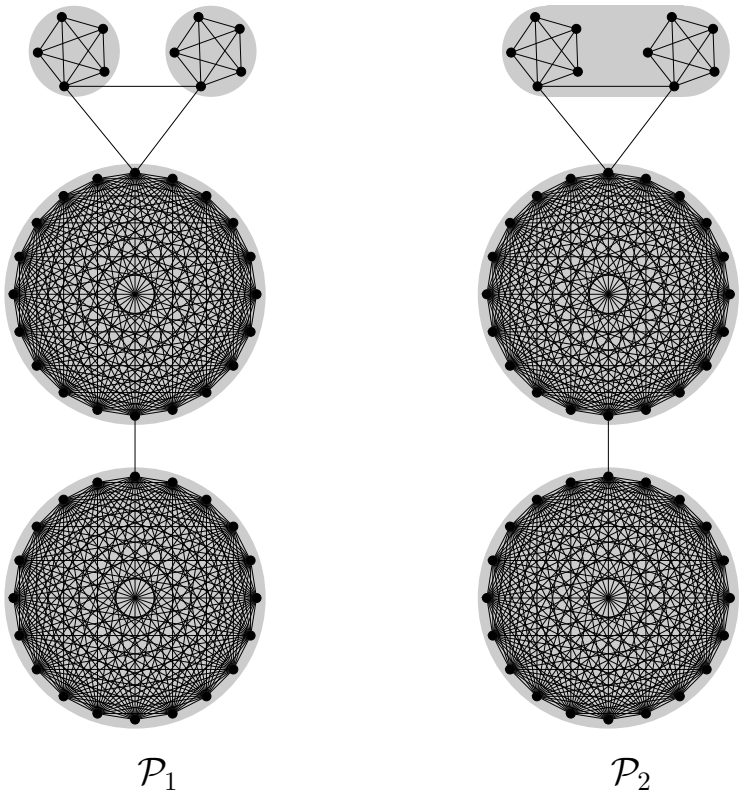
$$EQ(\mathcal{P}_1) - EQ(\mathcal{P}_2) \geq 2\left(e^{s(1-d_G)} - e^{\frac{4}{s}} - \frac{1}{2}\left(e^{2s(1-d_G)} - e^{\frac{2}{s}}\right)\right).$$

Применом леме 4.1 за $x = s$ и $y = d_G$ следи да је

$$EQ(\mathcal{P}_1) > EQ(\mathcal{P}_2),$$

што је требало доказати.

■



Слика 4.3: Партиције $\mathcal{P}_1$ и $\mathcal{P}_2$ графа G ($s = 20$, $q = 5$)

4.2 Генеричка метода претраге промењљивим околинама

У циљу тестирања предложене Е-функције и поређења са модуларношћу развијена је генеричка метода промењљивих околина која врши кластеровање мреже максимизацијом дате функције F која мери квалитет партиције. Партиција мреже на дисјунктне кластере, односно једно решење представљено је низом x дужине n тако да целобројна вредност на позицији i представља ознаку кластера коме припада чвор v_i. У фази размрдавања користе се четири типа околина:

1. $\mathcal{N}_1(x, k)$ – садржи сва решења која се добијају трансформацијом решења x тако што чворови из k кластера постају једночлани кластери.

2. $\mathcal{N}_2(x, k)$ – садржи сва решења која се добијају поделом k кластера из решења x тако да од сваког кластера настају два нова кластера.

3. $\mathcal{N}_3(x, k)$ – садржи сва решења која се добијају спајањем два суседна кластера у један и то k пута.

4. $\mathcal{N}_4(x, k)$ – садржи сва решења која се добијају померањем k чворова из једног кластера у један суседни кластер.

Псеудокод генеричке VNS методе за кластеровање мреже приказан је алгоритмом 4.6.

Алгоритам 4.6: Генеричка метода променљивих околина

улазни подаци: граф G, функција F, параметари k_{max}, t_{max}
излазни подаци: најбоље пронађено решење x

```
1  x ← СлучајноРешење()
2  x ← LPA(x)
3  k ← 1
4  while t < t_max do
5      t ← СлучајанИзбор(1, 2, 3, 4)
6      x' ← Размрдавање(x, k, t)
7      x' ← LPA(x')
8      if F(x') > F(x) then
9          x ← x'
10     if k ≥ min(k_max, БројКластера(x)) then
11         k ← 1
12     k ← k + 1
13     t ← УтрошеноПроцесорскоВреме()
```

Метода као улазне податке очекује граф G и произвољну функцију за мерење квалитета партиције $F : \mathcal{P} \longrightarrow R$, чијом се оптимизацијом идентификују кластери. Метода има два параметра, k_{max} којим се одређује максимална величина околине приликом претраге и t_{max} којим се одређује максимално време извршавања. Почетно решење x генерише се у првом кораку обиласком чворова у случајном поретку и додељивањем ознака кластера неозначеним чворовима и њиховим неозначеним суседима. Затим се у другом кораку почетно решење x поправља LPA алгоритмом. У трећем кораку врши се иницијализација бројача који одређује величину околине. Главни део методе представља итеративно понављање наредби од линије 5 до линије 14 у циљу претраживања простора допустивих решења. У првом кораку сваке итерације (линија 5)

на случајан начин се бира тип околине t која ће бити коришћена у тој итерацији. Затим се врши размрдавање текућег решења x у околини величине k изабраног типа t (линија 6). На добијено решење x' примењује се LPA алгоритам (линија 7). Добијено решење x' се упоређује са текућим решењем x помоћу функције F (линија 8). Ако решење x' има већу вредност функције квалитета F, оно постаје текуће решење x за следећу итерацију. На крају сваке итерације проверава се да ли је бројач околина k достигао максималну дефинисану вредност k_{max} и ако јесте враћа се на почетну вредност. Након што је утрошено дозвољено време извршавања t_{max}, прекида се итеративно претраживање а текуће решење представља уједно и најбоље пронађено решење.

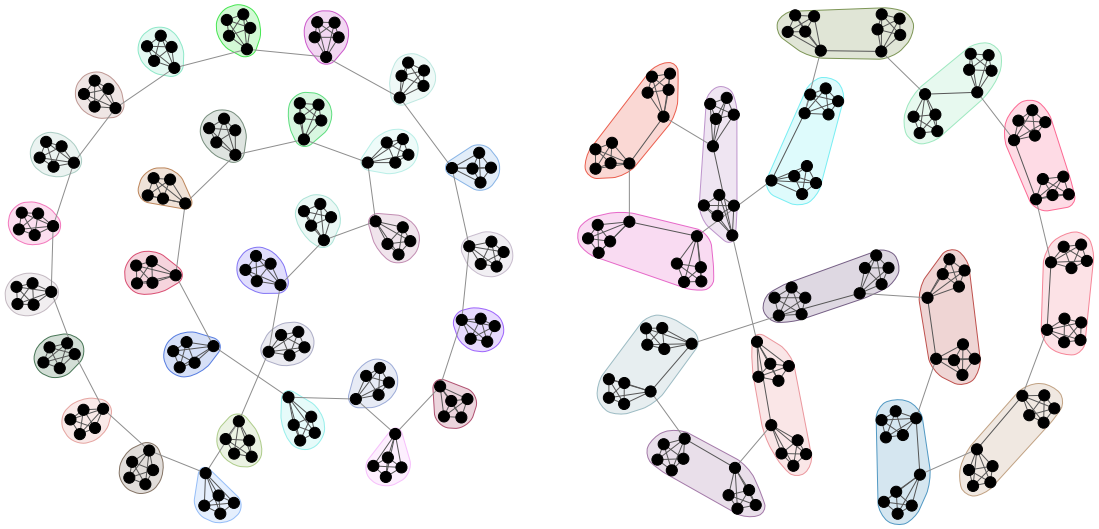
4.3 Експериментални резултати

У овој секцији приказани су резултати тестирања предложене Е-функције (EQ) за кластеровање, као и поређења са функцијом модуларности MQ. Скуп тест инстанци садржи генерисане мреже али и комплексне мреже из различитих домена примене. Генерисане инстанце омогућавају проверу валидности и поузданост Е-функције при контролисаним условима, док са друге стране реалне инстанце омогућавају да се сагледају њене могућности у практичним условима. За тестирање и поређење ове две функције коришћена је предложена генеричка метода променљивих околина. Алгоритам је имплементиран у програмском језику C++. Сва тестирања су извршена на рачунару са Intel Core i5-2400 - 3.10GHz процесором и 4GB радне меморије под Убунту оперативним системом. За максималну величину околине коришћена је вредност $k_{max} = 50$, док је максимално време извршавања 300 секунди $t_{max} = 300$. За вредност параметра r, који регулише степен утицаја спољних грана у EQ функцији, коришћена је вредност један за генерисане инстанце, док је за реалне инстанце коришћена вредност коефицијента корелације између максималне величине клике и степена чворова [135].

4.3.1 Генерисане инстанце

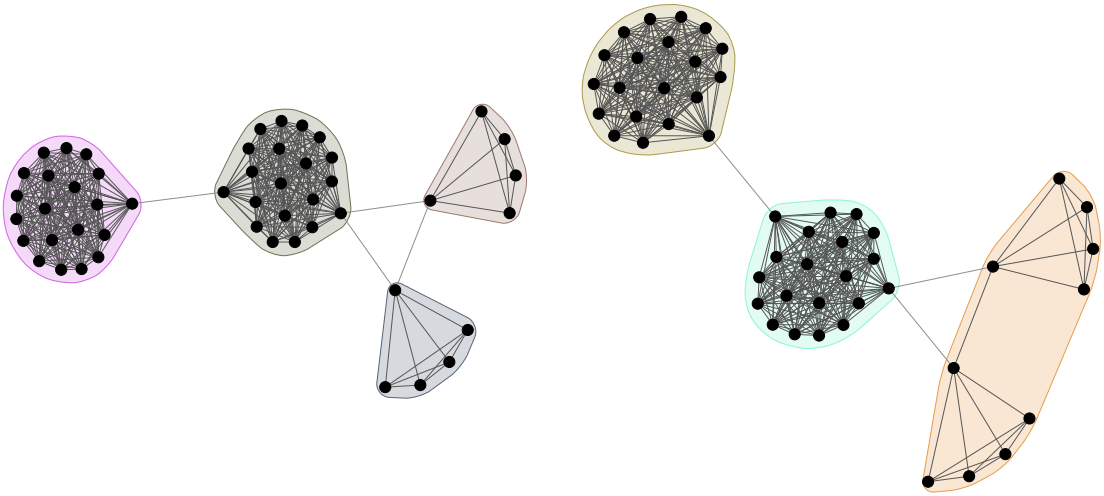
Скуп генерисаних инстанци који се често користи у литератури приликом истраживања функција за мерење квалитета партиције састоји се од четири мреже са јасно дефинисаним кластерима.

1. Мрежа са цикличним кликама. Прва инстанца (`ring of cliques`) из литературе [98, 105] је мрежа која се састоји од 30 клика са по 5 чворова повезаних у цикличан низ. Исправан резултат кластеровања је партиција у којој сваки кластер одговара једној клики. Метода претраге променљивим околинама са Е-функцијом (VNS-EQ) јасно препознаје 30 кластера у мрежи, док са MQ функцијом (VNS-MQ) уочава свега 16 кластера (слика 4.4). Резултат VNS-MQ методе је очекиван када се у обзир узму закључци из секције 2.3.3.



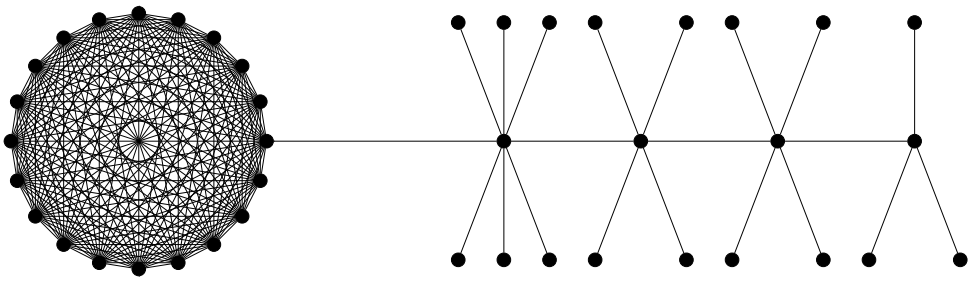
Слика 4.4: Партиције добијене генеричком методом променљивих околина на првој генерисаној инстанци

2. Мрежа са два пара идентичних клика. Друга инстанца (`pairwise cliques`) [98, 105, 136] је мрежа која се састоји од две велике клике са по 20 чворова, међусобно повезане једном граном, и две мале клике са по 5 чворова које су повезане једном граном, како међусобно тако и са великом кликом. Партиција добијена VNS-MQ методом садржи три кластера, где први и други одговарају великим кликама, док трећи кластер одговара унији две мале клике (слика 4.5). И ова инстанца потврђује да се при максимизацији функције модуларности јавља тенденција спајања више јасно дефинисаних малих кластера у један, без обзира на њихову структуру. Партиција добијена VNS-EQ методом садржи четири кластера који одговарају свакој од клика, што потврђује да се максимизацијом Е-функције могу уочити кластери различитих величина.



Слика 4.5: Партиције добијене генеричком методом променљивих околина на другој генерисаној инстанци

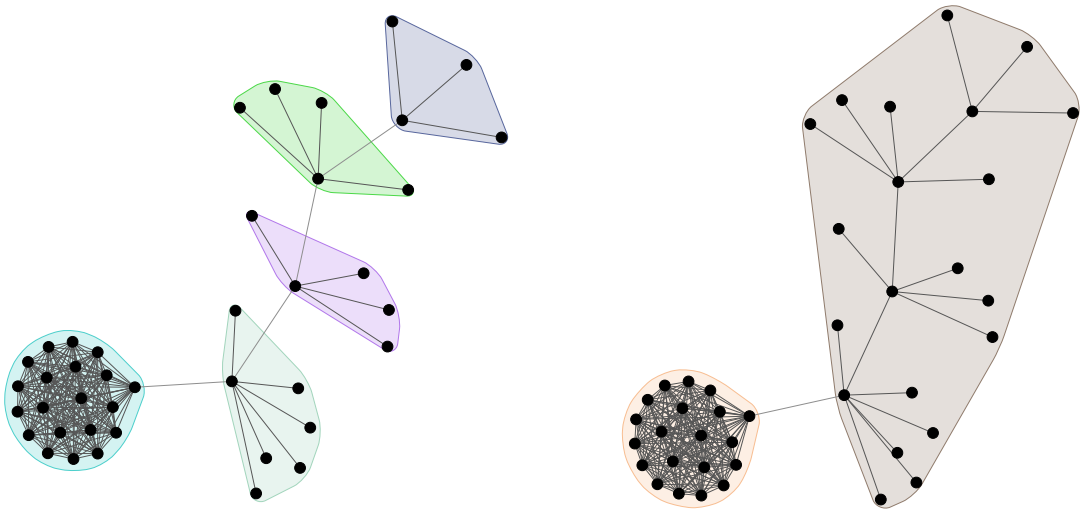
3. Клика праћена низом звезда. Трећа генерисана инстанца (`chain of stars`), предложена у [99], је мрежа сачињена од једне клике са 20 чворова на коју се надовезују четири графа са структуром звезде и различитим бројем чворова (слика 4.6).



Слика 4.6: Мрежа сачињена од клике праћене низом звезда

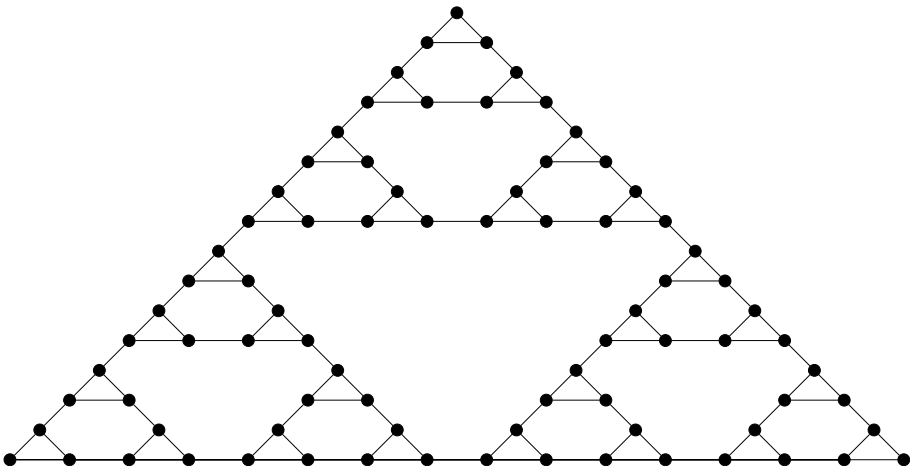
Сама мрежа има једноставну топологију, али представља једну од најтежих генерисаних инстанци јер садржи кластере различитих величина, а истовремено и различитих густина. Кластеровањем помоћу VNS-EQ методе могу се идентификовати пет кластера (слика 4.7). Са друге стране, VNS-MQ метода идентификује само 2 кластера, први који одговара клики, а други унији графова са топологијом звезде. На овој инстанци у раду [136] тестирана је модуларност са параметром резолуције MQ_r за регулисање разлике између

стварног и очекиваног односа грана у оригиналној дефиницији модуларности. Добијени резултати показују да не постоји вредност параметра r за који се максимизацијом модуларности може идентификовати пет кластера у овој мрежи.



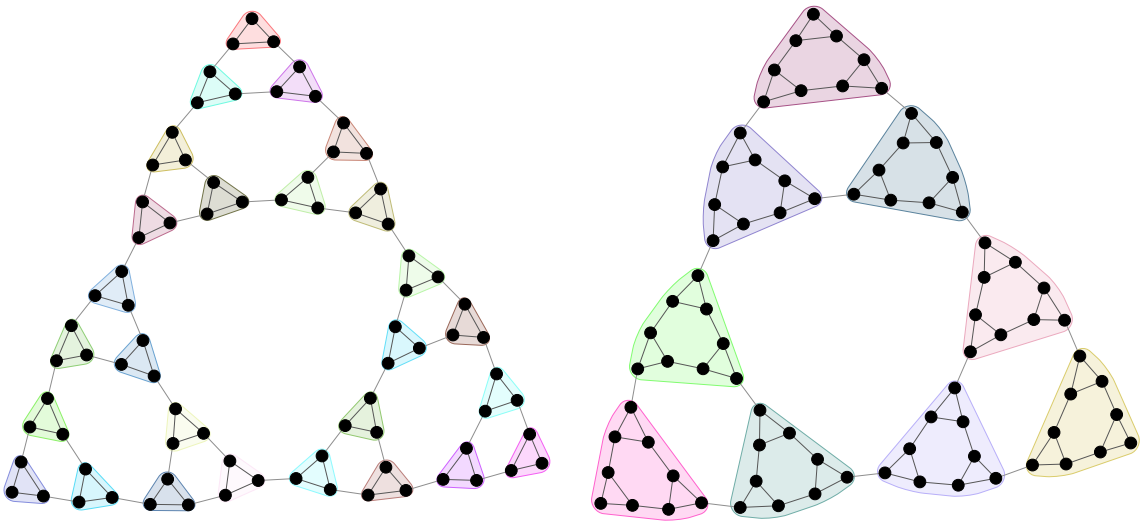
Слика 4.7: Партиције добијене генеричком методом променљивих околина на трећој генерисаној инстанци

4. Ханојски граф. Четврти пример (`hanoi graph`) [104, 77] представља Ханојски граф са хијерархијском организацијом чворова (слика 4.8). Ханојски граф H_n одговара дозвољеним потезима у проблему Ханојских кула. Граф H_n има 3^n чворова, $3(3^n - 1)/2$ грана и 3^{n-1} клика са по 3 чвора (троугла).



Слика 4.8: Ханојски граф H_4

За разлику од претходних мрежа у којима је подела на кластере очигледна, у овој мрежи постоји хијерархијска организација чворова. На првом нивоу организације то је унија три међусобно повезана чвора (троугао), на другом нивоу организације то је унија три међусобно повезана троугла итд. до n-тог нивоа на коме је цео граф. Из тог разлога на овој инстанци се може тестирати далековидност методе, тј. у овом случају, функције за мерење квалитета кластеровања. Кластеровањем на H_4 графу помоћу VNS-MQ методе могу се идентификовати девет кластера, тј. други ниво организације чворова (слика 4.9). Применом VNS-EQ методе добија се детаљнија подела са 27 кластера, тј. први ниво организације чворова. Дакле, Е-функција омогућава идентификовање организације на основном нивоу, за разлику од функције модуларности.



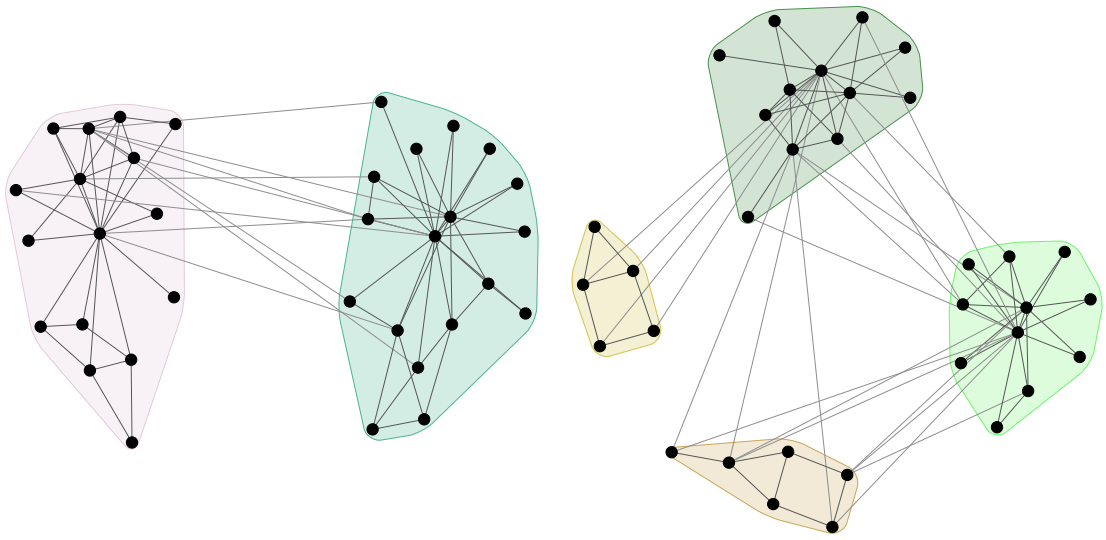
VNS-EQ VNS-MQ

Слика 4.9: Партиције добијене генеричком методом променљивих околина на четвртој генерисаној инстанци

4.3.2 Реалне инстанце

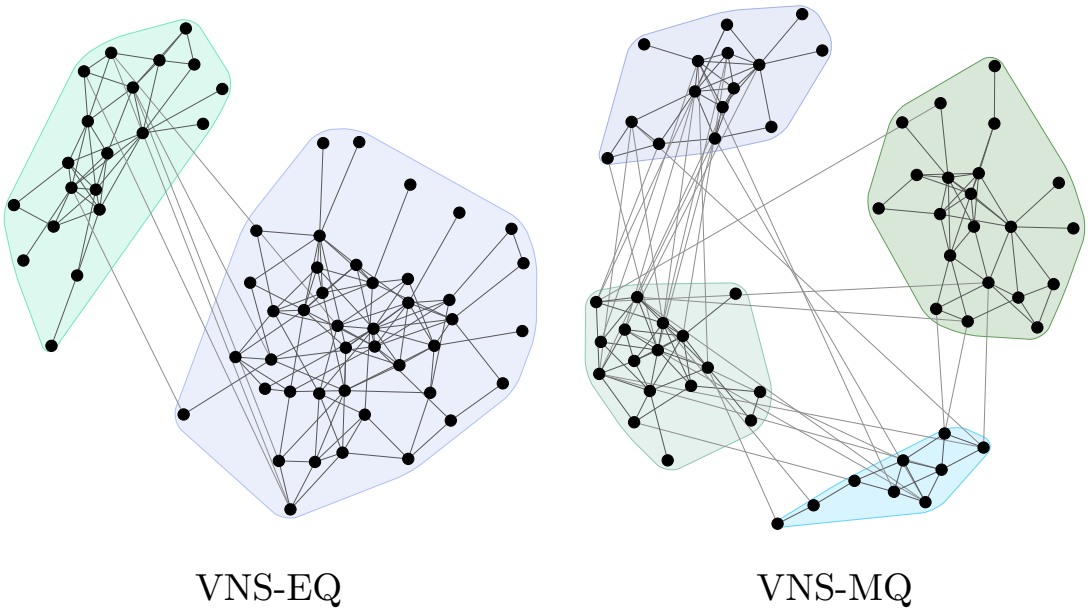
Мреже које се јављају у реалним применама имају другачија својства у поређењу са генерисаним инстанцама. Међутим, у веома малом броју случајева је унапред позната исправна подела на кластере.

1. Захаријев карате клуб. Мрежа је настала је као резултат Захаријевог надгледања интеракција између чланова универзитетског карате клуба [66] и описана је у секцији 2.2. Партиција добијена кластеровањем VNS-EQ методом, у потпуности одговара подели коју је описао Захарије. Са друге стране, партиција добијена VNS-MQ методом садржи четири релативно мала кластера која су приказана на слици 4.10. Спајањем четири идентификована кластера није могуће добити Захаријеву поделу у две групе.



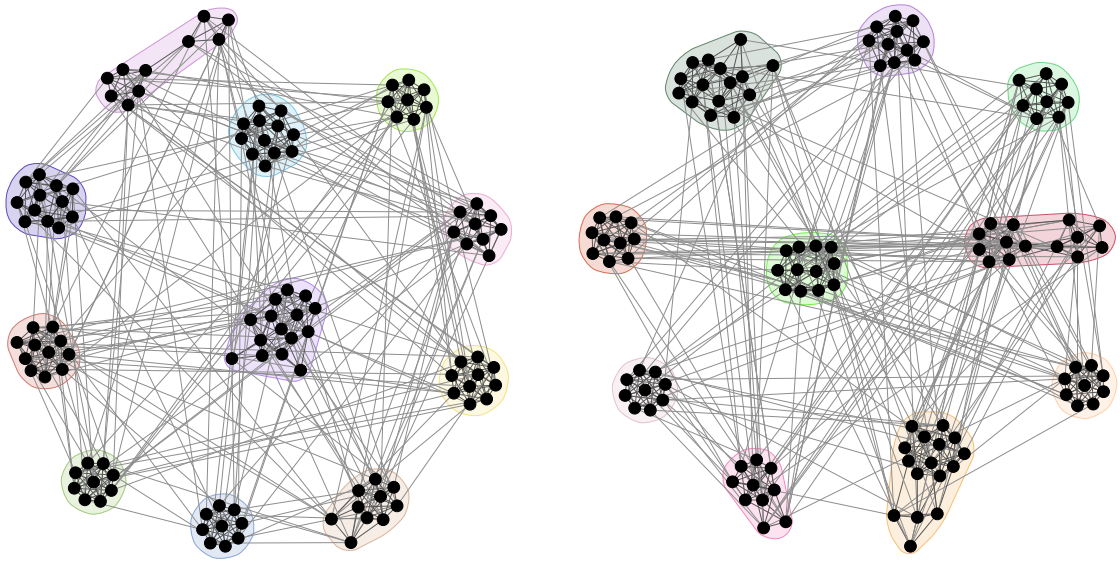
Слика 4.10: Партиције добијене генеричком методом променљивих околина за Захаријев карате клуб

2. Мрежа делфина. Подаци за креирање мреже су прикупљени кроз седмогодишње посматрање 62 делфина и одређених интеракција у фјордовима на Новом Зеланду [137]. Разматрањем добијене мреже може се јасно уочити подела у две групе на основу пола, па се из тог разлога користи у многобројним тестирањима метода кластеровања. VNS-EQ метода и на овом примеру идентификује партицију са два кластера која одговара очекиваној подели (слика 4.11). Кластеровањем помоћу VNS-MQ методе резултујућа партиција садржи четири кластера.



Слика 4.11: Партиције добијене генеричком методом променљивих околина за мрежу делфина

3. Амерички универзитетски фудбал. Ова мрежа описује одигране фудбалске утакмице између тимова I-A универзитетске дивизије током регуларног дела јесење сезоне у 2000. години [138]. Сви тимови, осим 8 независних, сврстани су у једну од 11 конференција. Чворови у мрежи (укупно 115) представљају тимове, док гране (укупно 613) представљају одигране утакмице између тимова. Највећи број утакмица је одигран између тимова из исте конференције. Поставља се питање да ли се кластеровањем ове мреже може реконструисати број конференција и њихов састав. VNS-MQ метода идентификује 10 кластера у мрежи, па реконструкција на основу добијене партиције није у потпуности могућа (слика 4.12). Са друге стране, VNS-EQ метода идентификује 11 кластера. Готово сви тимови су исправно груписани и налазе се у оквиру кластера са тимовима из своје конференције. Осам независних тимова који у стварности не припадају ниједној конференцији, придружени су групама у оквиру којих су имали највећи број утакмица, и лако се могу уочити у оквиру кластера.



VNS-EQ VNS-MQ

Слика 4.12: Партиције добијене генеричком методом променљивих околина за фудбалску мрежу

4.3.3 Компаративна анализа

Компаративна анализа модуларности и Е-функције представљена је у табелама 4.1 и 4.2.

У табели 4.1 упоређени су резултати добијени на инстанцама за које је позната коректна партиција. Прва врста означава називе инстанце, док су у другој и трећој врсти приказани број чворова n, односно број грана m. Прва колона представља назив методе. Осим VNS-EQ и VNS-MQ методе, у табели се налази и Луванова метода (LUV-MQ) [84] која користи MQ функцију. Поређењем VNS-MQ и LUV методе биће верификована исправност саме генеричке методе променљивих околина за максимизацију модуларности. За сваку инстанцу и добијено решење поменутим методама приказан је:

- $|\mathcal{P}|$ – број кластера;
- EQ – вредност Е-функције;
- MQ – вредност модуларности;
- $\mathcal{L}_{avg}$ – просечан број излазних грана по кластеру;
- $\mathcal{D}_{avg}$ – просечна густина кластера.

За сваку инстанцу истакнуте су најбоље вредности.

Табела 4.1: Поређење EQ и MQ функције на инстанцама за које се зна коректна подела у кластере

		chain of stars	pairwise cliques	hanoi graph	ring of cliques	karate	dolphins	football
	n	41	50	81	150	34	62	115
	m	211	404	120	330	78	159	613
	$\lvert\mathcal{P}\rvert$	**5**	**4**	27	**30**	**2**	**2**	**11**
	EQ_{best}	$\mathbf{2.70{\cdot}10^{6}}$	$\mathbf{1.27{\cdot}10^{6}}$	$\mathbf{2.93{\cdot}10^{2}}$	$\mathbf{3.72{\cdot}10^{3}}$	$\mathbf{4.69{\cdot}10^{0}}$	$\mathbf{1.91{\cdot}10^{1}}$	$\mathbf{1.87{\cdot}10^{4}}$
VNS-EQ	MQ	0.1634	0.5416	0.6379	0.8758	0.3715	0.3735	0.5999
	$\mathcal{L}_{avg}$	1.60	**2.00**	2.89	**2.00**	10.00	**6.00**	**34.18**
	$\mathcal{D}_{avg}$	0.52	**1.00**	**1.00**	**1.00**	0.25	0.18	**0.80**
	$\lvert\mathcal{P}\rvert$	2	3	9	16	4	4	10
	EQ	$2.70{\cdot}10^{6}$	$1.27{\cdot}10^{6}$	$1.12{\cdot}10^{2}$	$1.32{\cdot}10^{3}$	$-8.39{\cdot}10^{1}$	$-2.42{\cdot}10^{2}$	$1.69{\cdot}10^{4}$
VNS-MQ	MQ_{best}	**0.1707**	**0.5426**	**0.7889**	0.8871	0.4156	**0.5268**	**0.6046**
	$\mathcal{L}_{avg}$	**1.00**	**2.00**	**2.67**	**2.00**	**9.50**	16.00	35.80
	$\mathcal{D}_{avg}$	**0.55**	0.82	0.33	0.53	**0.49**	0.29	0.76
	$\lvert\mathcal{P}\rvert$	2	3	9	15	4	5	10
	EQ	$2.70{\cdot}10^{6}$	$1.27{\cdot}10^{6}$	$1.12{\cdot}10^{2}$	$1.15{\cdot}10^{3}$	$-8.00{\cdot}10^{1}$	$-5.57{\cdot}10^{2}$	$1.69{\cdot}10^{4}$
LUV-MQ	MQ_{best}	**0.1707**	**0.5426**	**0.7889**	**0.8879**	**0.4188**	0.5185	**0.6046**
	$\mathcal{L}_{avg}$	**1.00**	**2.00**	**2.67**	**2.00**	10.50	15.60	35.80
	$\mathcal{D}_{avg}$	**0.55**	0.82	0.33	0.47	0.45	**0.35**	0.76

Резултати показују да методе VNS-MQ и LUV-MQ производе партиције сличног квалитета у контексту модуларности. За четири инстанце максимална вредност модуларности се поклапа, на две инстанце LUV-MQ метода даје већу вредност модуларности, а на једној VNS-MQ метода. Број кластера у добијеним партицијама се разликује само на једној инстанци. Као што је већ показано, VNS-EQ метода на свим инстанцама даје коректну партицију, а самим тим и коректан број кластера $|\mathcal{P}|$. Међутим, узимајући само $\mathcal{L}_{avg}$ за критеријум рангирања VNS-EQ метода је последње рангирана (3 поена) иза LUV-MQ методе са (4 поена) и VNS-MQ методе (5 поена). Дакле, просечан број излазних грана по кластеру $\mathcal{L}_{avg}$ није добар показатељ коректности и квалитета партиције. Просечна густина кластера $\mathcal{D}_{avg}$ представља сигурнији критеријум рангирања с обзиром на то да коректне партиције добијене са

VNS-EQ методом препознаје са 3 поена и првим местом, док друго место са по 2 поена деле VNS-MQ и LUV-MQ методе.

У табели 4.2, која има исти формат као претходна, упоређени су резултати на додатном скупу инстанци већих димензија:

- `celegans` – метаболичка мрежа ваљкастог црва;
- `netscience` – мрежа коауторства истраживача који се баве науком о мрежама;
- `power` – мрежа која осликава структуру електричне мреже западних чланица САД-а;
- `hep-th` – мрежа коауторства истраживача теорије високе енергије добијена на основу објављених радова у е-принт архиви (1. 1. 1995. – 31. 12. 1999.);
- `PGP` – мрежа корисника Pretty Good Privacy софтвера (протокол за шифровање и дешифровање података);
- `astro-ph` – мрежа коауторства истраживача у области астрофизике добијена на основу објављених радова у е-принт архиви (1. 1. 1995. – 31. 12. 1999.);
- `condmat` – мрежа коауторства истраживача на пољу физике кондензоване материје добијена на основу објављених радова у е-принт архиви (1. 1. 1995. – 31. 12. 1999.);
- `as22july06` – мрежа која осликава структуру интернет мреже на нивоу аутономних система добијена реконструкцијом из таблица BGP протокола;
- `condmat2005` – мрежа коауторства истраживача на пољу физике кондензоване материје добијена на основу објављених радова у е-принт архиви (1. 1. 1995. – 31. 3. 2005.).

Табела 4.2: Додатно поређење метода кластеровања на инстанцама већих димензија

		celegans	netscience	power	hep-th	PGP	astro-ph	condmat	as22july06	condmat2005
	n	453	1589	4941	8361	10680	16706	16726	22963	40421
	m	2025	2742	6594	15751	24316	121251	47594	48436	175691
VNS-EQ	$\|\mathcal{P}\|$	13	456	486	1852	969	1516	2393	420	3462
	EQ_{best}	$\mathbf{9.04{\cdot}10^2}$	$\mathbf{2.17{\cdot}10^8}$	$\mathbf{4.58{\cdot}10^3}$	$\mathbf{2.45{\cdot}10^{10}}$	$\mathbf{6.26{\cdot}10^7}$	$\mathbf{4.51{\cdot}10^{19}}$	$\mathbf{1.98{\cdot}10^7}$	$\mathbf{2.50{\cdot}10^3}$	$\mathbf{3.20{\cdot}10^{13}}$
	MQ	0.3825	0.9109	0.8051	0.7712	0.8083	0.5643	0.7379	0.5098	0.4745
	$\mathcal{L}_{avg}$	**122.31**	0.90	**5.19**	3.47	8.62	**21.37**	10.30	**64.50**	**14.19**
	$\mathcal{D}_{avg}$	**0.43**	**0.62**	**0.34**	**0.42**	**0.44**	**0.45**	**0.56**	0.24	**0.56**
VNS-MQ	$\|\mathcal{P}\|$	9	290	299	1715	921	1675	2376	361	4144
	EQ	$5.53{\cdot}10^2$	$4.12{\cdot}10^7$	$2.87{\cdot}10^3$	$2.44{\cdot}10^{10}$	$1.87{\cdot}10^6$	$1.41{\cdot}10^{19}$	$2.33{\cdot}10^6$	$2.33{\cdot}10^3$	$3.20{\cdot}10^{13}$
	MQ_{best}	**0.4435**	0.9548	0.8496	0.7867	0.8185	0.6803	0.7420	0.5709	0.6330
	$\mathcal{L}_{avg}$	159.56	0.40	6.31	3.66	**8.12**	38.08	10.20	67.09	29.89
	$\mathcal{D}_{avg}$	0.34	0.52	0.27	0.41	0.44	0.44	0.56	**0.26**	0.54
LUV-MQ	$\|\mathcal{P}\|$	9	406	40	1380	99	1080	1253	41	1876
	EQ	$2.73{\cdot}10^2$	$2.17{\cdot}10^8$	$4.84{\cdot}10^2$	$2.44{\cdot}10^{10}$	$1.33{\cdot}10^6$	$5.11{\cdot}10^{16}$	$1.68{\cdot}10^5$	$4.35{\cdot}10^2$	$1.68{\cdot}10^8$
	MQ	0.4407	**0.9597**	**0.9363**	**0.8486**	**0.8834**	**0.7268**	**0.8462**	**0.6614**	**0.7224**
	$\mathcal{L}_{avg}$	189.78	**0.14**	11.30	**2.78**	38.16	48.32	**9.57**	524.63	41.63
	$\mathcal{D}_{avg}$	0.19	0.60	0.03	0.38	0.12	0.33	0.53	0.10	0.47

Партиције добијене максимизацијом EQ функције најчешће имају већи број кластера у односу на партиције добијене максимизацијом модуларности, али то није увек случај нпр. за инстанце `astro-ph` и `condmat2005`. Квалитет партиција добијених VNS-EQ методом по оба критеријума ($\mathcal{D}_{avg}$ и $\mathcal{L}_{avg}$) надмашује квалитет партиција добијених другим методама. Посматрајући просечан број излазних грана по кластеру, методе су рангиране у следећем поретку: VNS-EQ, LUV-MQ и VNS-MQ са резултатом $5 : 2 : 1$, респективно. Посматрајући просечну густину кластера, VNS-EQ метода је значајно боља од осталих јер производи квалитетнију партицију у осам од девет случајева. Осим тога, партиције добијене VNS-EQ методом имају модуларност MQ која је веома блиска максималној модуларности MQ_{best}, добијеној са VNS-MQ или LUV-MQ методом, што још једном потврђује да Е-функција има велики потенцијал за кластеровање на комплексним мрежама.

Глава 5

Закључак

Развијање метода кластеровања на комплексним мрежама је од великог значаја за разумевање динамике и еволуције комплексних мрежа које се појављују у различитим доменима. Идентификовање кластера може послужити за бољу визуализацију, као и за утврђивање особина појединачних чворова, односно њихових улога у мрежи. На пример, поједини чворови у кластеру могу имати улогу у повезивању кластера са остатком мреже, док други чворови могу имати улогу контролисања и стабилизације кластера.

Најчешће коришћен приступ за спровођење кластеровања на мрежи представља максимизација модуларности. У дисертацији је предложена ADVNDS метода која је заснована на методи променљивих околина за максимизацију модуларноси. У циљу ефикасне примене на комплексним мрежама великих димензија развијен је механизам за декомпозицију проблема на потпроблеме и побољшан механизам за превазилажење локалних максимума модуларности коришћењем критеријума за повремено прихватање лошијег решења од тренутно разматраног. За тестирање методе коришћене су DIMACS инстанце. Добијени резултати су упоређени са најбољим резултатима презентованим у литератури за разматрани проблем, који су добијени двема методама развијеним у оквиру DIMACS позива 2012. године. Осим тога, добијени резултати су упоређени и са резултатима шест метода развијених након 2012. године које су се издвојиле у литератури. Анализа резултата је показала да предложена ADVNDS метода надмашује постојеће методе и поправља најбоља позната решења на 9 од 13 тешких DIMACS инстанци.

Како кластеровање максимизацијом модуларности није погодно за откривање малих кластера у мрежама великих димензија чак и када су очигледни, у

оквиру дисертације је предложена нова мера која је названа E-функција. Кроз три тврђења показано је да нова мера превазилази недостатке који карактеришу модуларност и има потенцијал за идентификовање кластера у мрежи. За потребе детаљног тестирања и поређења предложене E-функције и модуларности развијена је генеричка метода променљивих околина. Рачунски експерименти спроведени су на генерисаним и реалним инстанцама из литературе за које је исправна подела на кластере позната. Добијени резултати потврђују теоријска разматрања и показују да се оптимизацијом E-функције на свим инстанцама могу идентификовати очекивани кластери.

Истраживања спроведена у оквиру ове дисертације дају важан допринос у решавању проблема кластеровања, као и областима комбинаторне оптимизације и машинског учења. Најважнији резултати који представљају научни допринос ове дисертације су:

- Развој нове методе (ADVNDS) која је заснована на методи променљивих околина за максимизацију модуларности.
- Нови приступ за превазилажење локалних максимума модуларности који се једноставно може применити приликом решавања других проблема математичке оптимизације.
- Примена предложене методе кластеровања и представљање нових побољшаних резултата у односу на познате резултате из литературе.
- Конструкција нове функције (E-функција) за мерење квалитета партиционисања мреже која превазилази недостатке уочене детаљном анализом модуларности.
- Развој генеричке методе променљивих околина за оптимизацију произвољне реалне функције којом се мери квалитет партиције.
- Примена генеричке методе променљивих околина за кластеровање комплексних мрежа максимизацијом модуларности и E-функције уз визуализацију и поређење резултата.

Резултати ове дисертације отварају нове могућности за истраживање проблема расплинутог кластеровања и кластеровања на вишеслојним мрежама модификацијом ADVNDS методе. Осим за максимизацију модуларности, предложени AD критеријум прихватања лошијег решења се може применити при

решавању других проблема са сличним карактеристикама. Правци даљег истраживања могу бити уопштавање E-функције за кластеровање тежинских мрежа или развијање специјализованих метода оптимизације које су се показале као ефикасне за максимизацију модуларности. Такође, погодно је размотрити кластеровање вишекритеријумском оптимизацијом модуларности, E-функције и других предложених функција из литературе.

Литература

[1] S. M. Manson, „Simplifying complexity: a review of complexity theory", *Geoforum*, vol. 32, no. 3, pp. 405–414, 2001.

[2] Y. Bar-Yam, „General features of complex systems", *Encyclopedia of Life Support Systems*, 2002.

[3] L. Euler, „Solutio problematis ad geometriam situs pertinentis", *Commentarii Academiae Scientiarum Petropolitanae*, pp. 128–140, 1741.

[4] D. König, *Theorie der endlichen und unendlichen Graphen: Kombinatorische Topologie der Streckenkomplexe*, vol. 16. Akademische Verlagsgesellschaft mbh, 1936.

[5] N. Biggs, E. K. Lloyd, and R. J. Wilson, *Graph Theory, 1736–1936*. Oxford University Press, 1986.

[6] D. Cvetković and M. Milić, *Teorija grafova i njene primene*. Beogradski izdavačko-grafički zavod, 1971.

[7] V. I. Voloshin, *Introduction to graph and hypergraph theory*. Nova Science Publication, 2009.

[8] J. A. Anderson, *Discrete mathematics with combinatorics*. Prentice Hall, 2001.

[9] M. Živković, *Algoritmi*. Matematički fakultet, Beograd, 2000.

[10] S. A. Cook, „The complexity of theorem-proving procedures", *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pp. 151–158, 1971.

[11] R. M. Karp, „Reducibility among combinatorial problems", *Complexity of Computer Computations*, pp. 85–103, 1972.

[12] P. Janičić, *Matematička logika u računarstvu*. Matematički fakultet, Beograd, 2008.

[13] P. E. Gill, W. Murray, M. A. Saunders, J. A. Tomlin, and M. H. Wright, „George B. Dantzig and systems optimization", *Discrete Optimization*, vol. 5, no. 2, pp. 151–158, 2008.

[14] L. G. Khachiyan, „A polynomial algorithm in linear programming", *Doklady Akademii Nauk*, vol. 244, no. 5, pp. 1093–1096, 1979.

[15] N. Karmarkar, „A new polynomial-time algorithm for linear programming", *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, pp. 302–311, 1984.

[16] Z. Stanimirović, *Nelinearno programiranje*. Matematički fakultet, Beograd, 2014.

[17] O. L. Mangasarian, *Nonlinear programming*. SIAM, 1994.

[18] M. Stanojević, *Metoda grananja i ograničavanja*. Fakultet organizacionih nauka, Beograd, 2018.

[19] S. Krčevinac, M. Čangalović, V. Kovačević-Vujčić, M. Martić, and M. Vujošević, *Operaciona istraživanja 1*. Fakultet organizacionih nauka, Beograd, 2006.

[20] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, „Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning", *Discrete Optimization*, vol. 19, pp. 79–102, 2016.

[21] M. W. P. Savelsbergh, „Branch and price: Integer programming with column generation", *Encyclopedia of Optimization*, pp. 328–332, 2009.

[22] E. W. Dijkstra *et al.*, „A note on two problems in connexion with graphs", *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[23] R. Bellman, „On a routing problem", *Quarterly of Applied Mathematics*, vol. 16, no. 1, pp. 87–90, 1958.

[24] F. Glover, „Future paths for integer programming and links to artificial intelligence", *Computers & Operations Research*, vol. 13, no. 5, pp. 533–549, 1986.

[25] P. J. Van Laarhoven and E. H. Aarts, „Simulated annealing", *Simulated Annealing: Theory and Applications*, pp. 7–15, 1987.

[26] F. Glover, „Tabu search — part I", *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190–206, 1989.

[27] F. Glover, „Tabu search — part II", *ORSA Journal on Computing*, vol. 2, no. 1, pp. 4–32, 1990.

[28] N. Mladenović and P. Hansen, „Variable neighborhood search", *Computers & Operations Research*, vol. 24, no. 11, pp. 1097–1100, 1997.

[29] T. A. Feo and M. G. Resende, „Greedy randomized adaptive search procedures", *Journal of Global Optimization*, vol. 6, no. 2, pp. 109–133, 1995.

[30] A. E. Eiben and J. E. Smith, *Introduction to evolutionary computing*. Springer, 2015.

[31] N. M. Al-Salami, „Evolutionary algorithm definition", *American Journal of Engineering and Applied Sciences*, vol. 2, no. 4, pp. 789–795, 2009.

[32] J. Kennedy, „Swarm intelligence", *Handbook of Nature-Inspired and Innovative Computing*, pp. 187–219, 2006.

[33] M. Dorigo, M. Birattari, and T. Stutzle, „Ant colony optimization", *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[34] R. Martí, M. Laguna, and F. Glover, „Principles of scatter search", *European Journal of Operational Research*, vol. 169, no. 2, pp. 359–372, 2006.

[35] F. Glover, M. Laguna, and R. Martí, „Scatter search", *Advances in Evolutionary Computing*, pp. 519–537, 2003.

[36] R. C. Eberhart and Y. Shi, *Computational intelligence: concepts to implementations*. Elsevier, 2011.

[37] Z. Stanimirović, M. Marić, N. Radojičić, and S. Božović, „Two efficient hybrid metaheuristic methods for solving the load balance problem", *Applied and Computational Mathematics*, vol. 13, no. 3, pp. 332–349, 2014.

[38] M. Marić, Z. Stanimirović, and S. Božović, „Hybrid metaheuristic method for determining locations for long-term health care facilities", *Annals of Operations Research*, vol. 227, no. 1, pp. 3–23, 2015.

[39] P. Moscato, „On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms", *Caltech Concurrent Computation Program*, i. 826, 1989.

[40] M. Marić, Z. Stanimirović, and P. Stanojević, „An efficient memetic algorithm for the uncapacitated single allocation hub location problem", *Soft Computing*, vol. 17, no. 3, pp. 445–466, 2013.

[41] M. Marić, Z. Stanimirović, A. Djenić, and P. Stanojević, „Memetic algorithm for solving the multilevel uncapacitated facility location problem", *Informatica*, vol. 25, no. 3, pp. 439–466, 2014.

[42] P. Stanojević, M. Marić, and Z. Stanimirović, „A hybridization of an evolutionary algorithm and a parallel branch and bound for solving the capacitated single allocation hub location problem", *Applied Soft Computing*, vol. 33, pp. 24–36, 2015.

[43] D. Schermer, M. Moeini, and O. Wendt, „A matheuristic for the vehicle routing problem with drones and its variants", *Transportation Research Part C: Emerging Technologies*, vol. 106, pp. 166–204, 2019.

[44] J. G. Villegas, C. Prins, C. Prodhon, A. L. Medaglia, and N. Velasco, „A matheuristic for the truck and trailer routing problem", *European Journal of Operational Research*, vol. 230, no. 2, pp. 231–244, 2013.

[45] T. Ting, X.-S. Yang, S. Cheng, and K. Huang, „Hybrid metaheuristic algorithms: past, present, and future", *Recent advances in swarm intelligence and evolutionary computation*, pp. 71–83, 2015.

[46] M. Abdel-Basset, L. Abdel-Fatah, and A. K. Sangaiah, „Metaheuristic algorithms: A comprehensive review", *Computational intelligence for multimedia big data on the cloud with engineering applications*, pp. 185–231, 2018.

[47] E. Paul and R. Alfréd, „On random graphs I", *Publicationes Mathematicae Debrecen*, vol. 6, pp. 290–297, 1959.

[48] P. Erdős and A. Rényi, „On the evolution of random graphs", *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, no. 1, pp. 17–60, 1960.

[49] M. Faloutsos, P. Faloutsos, and C. Faloutsos, „On power-law relationships of the internet topology", *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 4, pp. 251–262, 1999.

[50] A.-L. Barabási and R. Albert, „Emergence of scaling in random networks", *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[51] H. Jeong, S. P. Mason, A.-L. Barabási, and Z. N. Oltvai, „Lethality and centrality in protein networks", *Nature*, vol. 411, no. 6833, pp. 41–42, 2001.

[52] M. E. Newman, „The structure of scientific collaboration networks", *Proceedings of the National Academy of Sciences*, vol. 98, no. 2, pp. 404–409, 2001.

[53] R. Cohen and S. Havlin, „Scale-free networks are ultrasmall", *Physical Review Letters*, vol. 90, no. 5, i. 058701, 2003.

[54] A.-L. Barabási, *Network Science*. Cambridge University Press, 2016.

[55] D. J. Watts and S. H. Strogatz, „Collective dynamics of small-world networks", *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[56] M. Kaiser and C. C. Hilgetag, „Modelling the development of cortical systems networks", *Neurocomputing*, vol. 58, pp. 297–302, 2004.

[57] M. Catanzaro, M. Boguná, and R. Pastor-Satorras, „Generation of uncorrelated random scale-free networks", *Physical Review E*, vol. 71, no. 2, i. 027103, 2005.

[58] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*. Addison-Wesley, 1983.

[59] W.-F. Pan, B. Jiang, and B. Li, „Refactoring software packages via community detection in complex software networks", *International Journal of Automation and Computing*, vol. 10, no. 2, pp. 157–166, 2013.

[60] M. Zanin, P. Cano, J. M. Buldú, and O. Celma, „Complex networks in recommendation systems", *Proceedings of the 2nd WSEAS International Conference on Computer Engineering and Applications*, pp. 120–124, 2008.

[61] C. A. R. Pinheiro, „Community detection to identify fraud events in telecommunications networks", *SAS SUGI Proceedings: Customer Intelligence*, 2012.

[62] P. Xanthopoulos, A. Arulselvan, V. Boginski, and P. Pardalos, „A retrospective review of social networks", *Proceedings of the 2009 International Conference on Advances in Social Network Analysis and Mining*, pp. 300–305, 2009.

[63] N. S. Mitić, S. N. Malkov, J. J. Kovačević, G. M. Pavlović-Lažetić, and M. V. Beljanski, „Structural disorder of plasmid-encoded proteins in bacteria and archaea", *BMC bioinformatics*, vol. 19, no. 1, pp. 1–18, 2018.

[64] J. Chen, H. Zhang, Z.-H. Guan, and T. Li, „Epidemic spreading on networks with overlapping community structure", *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 4, pp. 1848–1854, 2012.

[65] G. Ren and X. Wang, „Epidemic spreading in time-varying community networks", *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 24, no. 2, i. 023116, 2014.

[66] W. W. Zachary, „An information flow model for conflict and fission in small groups", *Journal of Anthropological Research*, vol. 33, no. 4, pp. 452–473, 1977.

[67] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, „Defining and identifying communities in networks", *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 9, pp. 2658–2663, 2004.

[68] J. Eustace, X. Wang, and Y. Cui, „Community detection using local neighborhood in complex networks", *Physica A: Statistical Mechanics and its Applications*, vol. 436, pp. 665–677, 2015.

[69] J. Li, X. Wang, and J. Eustace, „Detecting overlapping communities by seed community in weighted complex networks", *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 23, pp. 6125–6134, 2013.

[70] U. N. Raghavan, R. Albert, and S. Kumara, „Near linear time algorithm to detect community structures in large-scale networks", *Physical Review E*, vol. 76, no. 3, i. 036106, 2007.

[71] X. Wang and J. Li, „Detecting communities by the core-vertex and intimate degree in complex networks", *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 10, pp. 2555–2563, 2013.

[72] M. E. Newman and M. Girvan, „Finding and evaluating community structure in networks", *Physical review E*, vol. 69, no. 2, i. 026113, 2004.

[73] B. Bollobás, „A probabilistic proof of an asymptotic formula for the number of labelled regular graphs", *European Journal of Combinatorics*, vol. 1, no. 4, pp. 311–316, 1980.

[74] F. Chung and L. Lu, „Connected components in random graphs with given expected degree sequences", *Annals of Combinatorics*, vol. 6, no. 2, pp. 125–145, 2002.

[75] Y. Zhao, E. Levina, J. Zhu, *et al.*, „Consistency of community detection in networks under degree-corrected stochastic block models", *The Annals of Statistics*, vol. 40, no. 4, pp. 2266–2292, 2012.

[76] M. Rosvall and C. T. Bergstrom, „Maps of random walks on complex networks reveal community structure", *Proceedings of the National Academy of Sciences*, vol. 105, no. 4, pp. 1118–1123, 2008.

[77] M. Rosvall and C. T. Bergstrom, „Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems", *PLOS ONE*, vol. 6, no. 4, i. 18209, 2011.

[78] B. Ball, B. Karrer, and M. E. Newman, „Efficient and principled method for detecting communities in networks", *Physical Review E*, vol. 84, no. 3, i. 036103, 2011.

[79] S. Fortunato, „Community detection in graphs", *Physics reports*, vol. 486, no. 3, pp. 75–174, 2010.

[80] S. Fortunato and D. Hric, „Community detection in networks: A user guide", *Physics reports*, vol. 659, pp. 1–44, 2016.

[81] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, „On modularity clustering", *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 2, pp. 172–188, 2008.

[82] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron, and L. Liberti, „Column generation algorithms for exact modularity maximization in networks", *Physical Review E*, vol. 82, no. 4, i. 046112, 2010.

[83] D. Aloise, G. Caporossi, P. Hansen, L. Liberti, S. Perron, and M. Ruiz, „Modularity maximization in networks by variable neighborhood search", *Graph Partitioning and Graph Clustering*, vol. 588, pp. 113–127, 2013.

[84] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, „Fast unfolding of communities in large networks", *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, i. 10008, 2008.

[85] M. J. Barber and J. W. Clark, „Detecting network communities by propagating labels under constraints", *Physical Review E*, vol. 80, no. 2, i. 026129, 2009.

[86] X. Liu and T. Murata, „Advanced modularity-specialized label propagation algorithm for detecting communities in networks", *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 7, pp. 1493–1500, 2010.

[87] M. Tasgin, A. Herdagdelen, and H. Bingol, „Community detection in complex networks using genetic algorithms", *Proceedings of the European Conference on Complex Systems*, Apr. 2006.

[88] R. Shang, J. Bai, L. Jiao, and C. Jin, „Community detection based on modularity and an improved genetic algorithm", *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 5, pp. 1215–1231, 2013.

[89] S. Bilal and M. Abdelouahab, „Evolutionary algorithm and modularity for detecting communities in networks", *Physica A: Statistical Mechanics and its Applications*, vol. 473, pp. 89–96, 2017.

[90] Z. Lü and W. Huang, „Iterated tabu search for identifying community structure in complex networks", *Physical Review E*, vol. 80, no. 2, i. 026130, 2009.

[91] O. Gach and J.-K. Hao, „Combined neighborhood tabu search for community detection in complex networks", *RAIRO - Operations Research*, vol. 50, no. 2, pp. 269–283, 2016.

[92] R. Guimera and L. A. N. Amaral, „Functional cartography of complex metabolic networks", *Nature*, vol. 433, no. 7028, pp. 895–900, 2005.

[93] J. Liu and T. Liu, „Detecting community structure in complex networks using simulated annealing with k-means algorithms", *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 11, pp. 2300–2309, 2010.

[94] C. Honghao, F. Zuren, and R. Zhigang, „Community detection using ant colony optimization", *IEEE Congress on Evolutionary Computation*, pp. 3072–3078, 2013.

[95] D. He, J. Liu, D. Liu, D. Jin, and Z. Jia, „Ant colony optimization for community detection in large-scale complex networks", *7th International Conference on Natural Computation*, vol. 2, pp. 1151–1155, 2011.

[96] M. C. Nascimento and L. Pitsoulis, „Community detection by modularity maximization using GRASP with path relinking", *Computers & Operations Research*, vol. 40, no. 12, pp. 3121–3131, 2013.

[97] L. M. Naeni, R. Berretta, and P. Moscato, „MA-Net: A reliable memetic algorithm for community detection by modularity optimization", *Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems*, pp. 311–323, 2015.

[98] S. Fortunato and M. Barthelemy, „Resolution limit in community detection", *Proceedings of the National Academy of Sciences*, vol. 104, no. 1, pp. 36–41, 2007.

[99] A. Arenas, A. Fernandez, and S. Gomez, „Analysis of the structure of complex networks at different resolution levels", *New Journal of Physics*, vol. 10, no. 5, i. 053039, 2008.

[100] P. Pons and M. Latapy, „Post-processing hierarchical community structures: Quality improvements and multi-scale view", *Theoretical Computer Science*, vol. 412, no. 8-10, pp. 892–900, 2011.

[101] A. Lancichinetti and S. Fortunato, „Limits of modularity maximization in community detection", *Physical Review E*, vol. 84, no. 6, i. 066122, 2011.

[102] Z. Li, S. Zhang, R.-S. Wang, X.-S. Zhang, and L. Chen, „Quantitative function for community detection", *Physical Review E*, vol. 77, no. 3, i. 036109, 2008.

[103] M. Chen, T. Nguyen, and B. K. Szymanski, „A new metric for quality of network community structure", *ASE Human Journal*, vol. 2, no. 4, pp. 226–240, 2013.

[104] A. Miyauchi and Y. Kawase, „Z-score-based modularity for community detection in networks", *PLOS ONE*, vol. 11, no. 1, i. 0147805, 2016.

[105] S. Fortunato, „Quality functions in community detection", *SPIE Fourth International Symposium on Fluctuations and Noise*, i. 660108, 2007.

[106] J. Creusefond, T. Largillier, and S. Peyronnet, „On the evaluation potential of quality functions in community detection for different contexts", *International Conference and School on Network Science*, pp. 111–125, 2016.

[107] P. Hansen, N. Mladenović, J. Brimberg, and J. A. M. Pérez, „Variable neighborhood search", *Handbook of metaheuristics*, pp. 57–97, 2019.

[108] P. Hansen, N. Mladenović, R. Todosijević, and S. Hanafi, „Variable neighborhood search: basics and variants", *Euro Journal on Computational Optimization*, vol. 5, no. 3, pp. 423–454, 2017.

[109] D. Džamić, D. Aloise, and N. Mladenović, „Ascent–descent variable neighborhood decomposition search for community detection by modularity maximization", *Annals of Operations Research*, vol. 272, no. 1-2, pp. 273–287, 2019.

[110] T. N. Dinh and M. T. Thai, „Toward optimal community detection: From trees to general weighted networks", *Internet Mathematics*, vol. 11, no. 3, pp. 181–200, 2015.

[111] A. Miyauchi and N. Sukegawa, „Redundant constraints in the standard formulation for the clique partitioning problem", *Optimization Letters*, vol. 9, no. 1, pp. 199–207, 2015.

[112] G. Xu, S. Tsoka, and L. G. Papageorgiou, „Finding community structures in complex networks using mixed integer optimisation", *The European Physical Journal B*, vol. 60, no. 2, pp. 231–239, 2007.

[113] N. Mladenovic, „A variable neighborhood algorithm-a new metaheuristic for combinatorial optimization", *Optimization Days*, 1995.

[114] A. Anokić, Z. Stanimirović, T. Davidović, and Đ. Stakić, „Variable neighborhood search based approaches to a vehicle scheduling problem in agriculture", *International Transactions in Operational Research*, vol. 27, no. 1, pp. 26–56, 2020.

[115] S. Mišković, Z. Stanimirović, and I. Grujičić, „An efficient variable neighborhood search for solving a robust dynamic facility location problem in emergency service network", *Electronic Notes in Discrete Mathematics*, vol. 47, pp. 261–268, 2015.

[116] N. Mladenović, M. Dražić, V. Kovačevic-Vujčić, and M. Čangalović, „General variable neighborhood search for the continuous optimization", *European Journal of Operational Research*, vol. 191, no. 3, pp. 753–770, 2008.

[117] A. Bačević, N. Vilimonović, I. Dabić, J. Petrović, D. Damnjanović, and D. Džamić, „Variable neighborhood search heuristic for nonconvex portfolio optimization", *The Engineering Economist*, vol. 64, no. 3, pp. 254–274, 2019.

[118] A. Djenić, M. Marić, Z. Stanimirović, and P. Stanojević, „A variable neighbourhood search method for solving the long-term care facility location problem", *IMA Journal of Management Mathematics*, vol. 28, no. 2, pp. 321–338, 2017.

[119] A. Djenić, N. Radojičić, M. Marić, and M. Mladenović, „Parallel VNS for bus terminal location problem", *Applied Soft Computing*, vol. 42, pp. 448–458, 2016.

[120] P. Hansen, N. Mladenović, and D. Perez-Britos, „Variable neighborhood decomposition search", *Journal of Heuristics*, vol. 7, no. 4, pp. 335–350, 2001.

[121] J. Lazić, S. Hanafi, N. Mladenović, and D. Urošević, „Variable neighbourhood decomposition search for 0–1 mixed integer programs", *Computers & Operations Research*, vol. 37, no. 6, pp. 1055–1067, 2010.

[122] P. Hansen, J. Brimberg, D. Urošević, and N. Mladenović, „Primal-dual variable neighborhood search for the simple plant-location problem", *INFORMS Journal on Computing*, vol. 19, no. 4, pp. 552–564, 2007.

[123] R. Plotnikov, A. Erzin, and N. Mladenovic, „VNDS for the min-power symmetric connectivity problem", *Optimization Letters*, vol. 13, no. 8, pp. 1897–1911, 2019.

[124] B. H. Good, Y.-A. De Montjoye, and A. Clauset, „Performance of modularity maximization in practical contexts", *Physical Review E*, vol. 81, no. 4, i. 046106, 2010.

[125] H. Zhang, X. Chen, J. Li, and B. Zhou, „Fuzzy community detection via modularity guided membership-degree propagation", *Pattern Recognition Letters*, vol. 70, pp. 66–72, 2016.

[126] P. Wu and L. Pan, „Multi-objective community detection based on memetic algorithm", *PLOS ONE*, vol. 10, no. 5, i. e0126845, 2015.

[127] D. A. Bader, H. Meyerhenke, P. Sanders, and D. Wagner, *Graph partitioning and graph clustering*, vol. 588. American Mathematical Society, 2013.

[128] M. Ovelgönne and A. Geyer-Schulz, „An ensemble learning strategy for graph clustering", *Graph Partitioning and Graph Clustering*, vol. 588, pp. 187–205, 2012.

[129] H. Lou, S. Li, and Y. Zhao, „Detecting community structure using label propagation with weighted coherent neighborhood propinquity", *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 14, pp. 3095–3105, 2013.

[130] S. Sobolevsky, R. Campari, A. Belyi, and C. Ratti, „General optimization technique for high-quality community detection in complex networks", *Physical Review E*, vol. 90, no. 1, i. 012811, 2014.

[131] P. G. Sun, „Community detection by fuzzy clustering", *Physica A: Statistical Mechanics and its Applications*, vol. 419, pp. 408–416, 2015.

[132] J. Su and T. C. Havens, „Fuzzy community detection in social networks using a genetic algortihm", *IEEE International Conference on Fuzzy Systems*, pp. 2039–2046, 2014.

[133] D. Džamić, J. Pei, M. Marić, N. Mladenović, and P. M. Pardalos, „Exponential quality function for community detection in complex networks", *International Transactions in Operational Research*, vol. 27, no. 1, pp. 245–266, 2020.

[134] D. Ž. Džamić, „Some properties of E-quality function for network clustering", *Yugoslav Journal of Operations Research*, vol. 31, no. 1, pp. 65–74, 2021.

[135] N. Meghanathan, „Distribution of maximal clique size of the vertices for theoretical small-world networks and real-world networks", *International journal of Computer Networks & Communications*, vol. 7, pp. 21—41, July 2015.

[136] A. Bettinelli, P. Hansen, and L. Liberti, „Algorithm for parametric community detection in networks", *Physical Review E*, vol. 86, no. 1, i. 016107, 2012.

[137] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, „The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations", *Behavioral Ecology and Sociobiology*, vol. 54, no. 4, pp. 396–405, 2003.

[138] M. Girvan and M. E. Newman, „Community structure in social and biological networks", *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

Биографија аутора

Душан Цамић рођен је 28. 7. 1990. године у Крушевцу. Основну школу „Николај Велимировић" и гиманзију „Свети Трифун" у Александровцу завршио је са одличним успехом. Математички факултет у Београду уписао је 2009. године. Дипломирао је у јуну 2013. године на смеру Математика (модул Рачунарство и информатика) са просечном оценом 9,36. Мастер академске студије завршио је 2014. године са просечном оценом 10,00 и одбрањеном мастер тезом под називом „Решавање проблема распоређивања послова у вишефазној производњи применом хибридних метахеуристичких алгоритама" под менторством др Мирослава Марића. По завршетку мастер студија уписао је докторске студије студијског програма Информатика, на Математичком факултету Универзитета у Београду. Положио је све испите на докторским студијама са просечном оценом 10,00.

Од 2013. до 2014. године радио је на Математичком факултету Универзитета у Београду, као сарадник у настави за ужу научну област Математичка анализа. У зимском семестру школске 2013/14. изводио је практикум/вежбе на предметима Анализа 1А и Математика 1Ц (Физички факултет). Од децембра 2013. године запослен је на Факултету организационих наука Универзитета у Београду, као сарадник у настави, а од децембра 2015. године као асистент, за ужу научну област Математичке методе у менаџменту и информатици. До сада је изводио вежбе на предметима Математика 1, Математика 2, Математика 3 и Дискретне математичке структуре. У анкетама за вредновање педагошког рада од стране студената редовно је оцењиван високим оценама. Коаутор је две збирке задатака из предмета Математика 2 и Математика 3 које се користе као основна литература на Факултету организационих наука. Од 2014. године члан је комисије за састављање задатака за пријемни испит из Математике, на Факултету организационих наука. На такмичењу „ФОН Хакатон" био је стручни ментор 2018. и 2020. године. Члан

је Друштва математичара Србије од 2016. године. Од 2018. године члан је удружења „Млади математичар“ које организује међународно такмичење „Мост математике”. Од 2014. године активно учествује у раду групе за образовни софтвер на Математичком факултету.

Од 2015. до 2021. године био је члан пројекта „Математички модели и методе оптимизације великих система”, пројекат бр. 174010, Министарства просвете, науке и технолошког развоја Републике Србије. Аутор је 18 публикација – научних радова публикованих у међународним часописима (од чега 4 на SCI листи), зборницима радова са међународних и националних научних скупова објављених у целини или изводу. Био је члан организационог одбора балканске конференције о операционим истраживањима – BALCOR 2018 и међународног симпозијума о операционим истраживањима – SYMOPIS 2019.

Прилог 1.

Изјава о ауторству

Потписани-а Душан Џамић

број индекса 2033/2014

Изјављујем

да је докторска дисертација под насловом

Нове методе кластеровања на комплексним мрежама

- резултат сопственог истраживачког рада,
- да предложена дисертација у целини ни у деловима није била предложена за добијање било које дипломе према студијским програмима других високошколских установа,
- да су резултати коректно наведени и
- да нисам кршио/ла ауторска права и користио интелектуалну својину других лица.

Потпис докторанда

У Београду, 17. 3. 2021.

Прилог 2.

Изјава о истоветности штампане и електронске верзије докторског рада

Име и презиме аутора Душан Џамић

Број индекса 2033/2014

Студијски програм Информатика

Наслов рада Нове методе кластеровања на комплексним мрежама

Ментор проф. др Мирослав Марић

Изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао/ла за објављивање на порталу **Дигиталног репозиторијума Универзитета у Београду.**

Дозвољавам да се објаве моји лични подаци везани за добијање академског звања доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним страницама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Београду.

Потпис докторанда

У Београду, 17. 3. 2021.

_____

Прилог 3.

Изјава о коришћењу

Овлашћујем Универзитетску библиотеку „Светозар Марковић“ да у Дигитални репозиторијум Универзитета у Београду унесе моју докторску дисертацију под насловом:

Нове методе кластеровања на комплексним мрежама

која је моје ауторско дело.

Дисертацију са свим прилозима предао/ла сам у електронском формату погодном за трајно архивирање.

Моју докторску дисертацију похрањену у Дигитални репозиторијум Универзитета у Београду могу да користе сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons) за коју сам се одлучио/ла.

1. Ауторство

2. Ауторство - некомерцијално

3. **Ауторство – некомерцијално – без прераде**

4. Ауторство – некомерцијално – делити под истим условима

5. Ауторство – без прераде

6. Ауторство – делити под истим условима

(Молимо да заокружите само једну од шест понуђених лиценци, кратак опис лиценци дат је на полеђини листа).

Потпис докторанда

У Београду, 17. 3. 2021.

_____