UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

DETEKOVANIE KOMUNÍT V SOCIÁLNYCH SIEŤACH

BAKALÁRSKA PRÁCA

2017 Patricia SVITKOVÁ

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

DETEKOVANIE KOMUNÍT V SOCIÁLNYCH SIEŤACH

BAKALÁRSKA PRÁCA

Študijný program:	Ekonomická a finančná matematika
Študijný odbor:	1114 Aplikovaná matematika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Vedúci práce:	doc. RNDr. Beáta Stehlíková, PhD.

Bratislava 2017 **Patricia SVITKOVÁ**

33364154

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Patricia Svitková
Študijný program: ekonomická a finančná matematika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: aplikovaná matematika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Detekcia komunít v sociálnych sieťach
Community detection in social networks

Cieľ: Sociálna sieť pozostáva z vrcholov (predstavujú ľudí), ktoré môžu byť spojené hranami (to vyjadruje existenciu nejakej väzby). Jedným z klasických príkladov na komunity v sociálnych sieťach je "Zacharyho karate klub" [2], ktorý sa rozpadol, a na základe analýzy siete vzťahov sa dalo predpokadať, kto sa pridá ku ktorej skupine. Existujú rôzne algoritmy na hľadanie komunít v sieťach, popísané napríklad v [1]. Knižnica igraph [3] pre softvér R okrem iného obsahuje viaceré takéto algoritmy.

Bakalárska práca bude obsahovať:
(a) Vysvetlenie algoritmov, aby ich použitie nebolo "čiernou skrinkou" a ukážka ich použitia pre malé siete s uvedením jednotlivých krokov výpočtu.
(b) Praktickú aplikáciu pre zvolené siete, pričom aspoň 2-3 príklady budú založené na zbieraní vlastných dát

Literatúra:
[1] S. Fortunato: Community detection in graphs. Physics Reports 486 (2010) 75-174.
[2] W. W. Zachary, An information flow model for conflict and fission in small groups, Journal of Anthropological Research 33, 452-473 (1977).
[3] Functions to deal with the result of network community detection. http://igraph.org/r/doc/communities.html

Vedúci: doc. RNDr. Beáta Stehlíková, PhD.
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Daniel Ševčovič, CSc.

Dátum zadania: 25.10.2016

Dátum schválenia: 19.11.2016

doc. RNDr. Margaréta Halická, CSc.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie Moje poďakovanie v prvom rade patrí doc. RNDr. Beáte Stehlíkovej, PhD., mojej školiteľke, za jej ochotu, venovaný čas a podnetné pripomienky. Taktiež ďakujem svojim spolužiakom za pomoc pri zbieraní dát a svojej rodine a priateľom za ich trpezlivosť a podporu.

Abstrakt

SVITKOVÁ, Patricia: Detekovanie komunít v sociálnych sieťach [Bakalárska práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky; školiteľ: doc. RNDr. Beáta Stehlíková, PhD., Bratislava, 2017, 66 s.

V sociálnych sieťach často dochádza k fenoménu zhlukovania a členenia sa do menších skupín - komunít. Cieľom našej práce je predstaviť viacero spôsobov nazerania na daný jav a vysvetliť niektoré vybrané algoritmy, ktorými je možné odhadnúť komunity v skúmanej sieti. Bližšie sa pozrieme na Ford-Fulkersonov algoritmus skonštruovaný pôvodne pre iné účely a na jeho možnú implementáciu na danú problematiku. Ďalej predstavíme základnú myšlienku a podrobný postup v Girvan-Newmanovom algoritme. Taktiež sa budeme zaoberať algoritmom navrhnutom Clausetom, Newmanom a Moorom patriacom do triedy *greedy*. Na záver algoritmy aplikujeme na reálne sociálne siete a získané výsledky príhodným spôsobom porovnáme.

Kľúčové slová: zhlukovanie, algoritmy, kapacita rezu, *betweenness*, modularita

Abstract

SVITKOVÁ, Patricia: Community detection in social networks [Bachelor Thesis], Comenius University in Bratislava, Faculty of Mathematics, Physics and Informatics, Department of Applied Mathematics and Statistics; Supervisor: doc. RNDr. Beáta Stehlíková, PhD, Bratislava, 2016, 66 p.

In social networks often occurs a phenomena of clustering and partitioning into smaller groups. The aim of this bachelor thesis is to introduce various perspectives on looking on this effect and explain the chosen algorithms, which are able to reveal communities in the examined network. We look closer at Ford-Fulkerson algorithm, which was originally constructed for another purpose, and explore its possible implementation in this area. Then we explain the main idea and detailed procedure of Girvan-Newman algorithm. We also study the greedy algorithm by Clauset, Newman and Moore. In conclusion, we apply these algorithms to real social networks and compare obtained results.

Keywords: clustering, algorithms, cut capacity, betweenness, modularity

Obsah

Úvod

Siete zachytávajú systémy vzťahov medzi objektami. Sieťami je možné reprezentovať veľa systémov v našom svete, od priateľstiev medzi ľuďmi, cez správanie zvierat, až po systém neurónov v biológií, či rozličné systémy v informatike, politike alebo ekonómií. My sa budeme zaoberať sociálnymi sieťami, ktoré zachytávajú rôzne typy medziľudských vzťahov. Ľudia spolu tvoria sociálne siete či už na pracoviskách, v rodinách, v školách, v mestách alebo národoch. V súčasnej dobe sa stávajú rozšírenými aj tzv. online sociálne siete, ako napríklad Facebook alebo Twitter.

Je veľa aspektov, ktoré možno v sieťach analyzovať. Jedným z nich je možné zhlukovanie skúmaných objektov do menších skupín (komunít), v rámci ktorých majú objekty silnejšie väzby. V sociálnych sieťach môže ísť napríklad združovanie ľudí s podobnými politickými, či náboženskými názormi alebo podobnými záujmami. Za začiatok analýzy sociálnych sietí s komunitnou štruktúrou sa pokladajú tridsiate roky minulého storočia a odvtedy sa v tejto disciplíne výrazne pokročilo. Jednou z možných využití skúmania uvedeného javu môžeme nájsť napríklad v marketingu, kde pri správnom zatriedení klienta s určitými záujmami mu je možné odporučiť vhodný produkt.

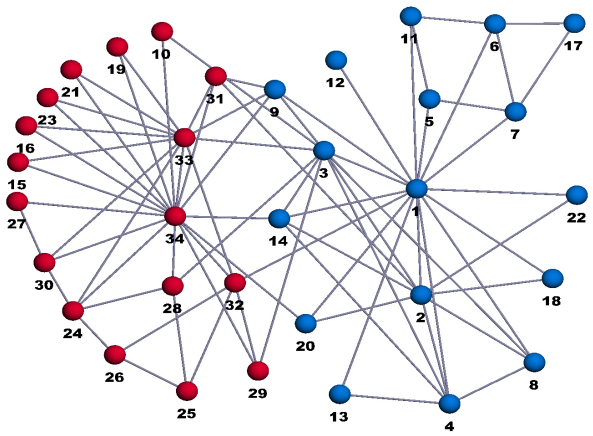
Cieľom našej práce bolo bližšie vysvetliť niektoré vybrané algoritmy, využívané v danej problematike. Predstavíme základné myšlienky, na ktorých boli stavané. Algoritmy následne podrobne vysvetlíme tak, aby boli čitateľsky prístupné. Pre lepšie pochopenie nechýba ani grafické znázornenie. Na záver aplikujeme nadobudnuté vedomosti do praxe a algoritmy otestujeme na sociálnych sieťach z reálnych dát.

V prvej kapitole oboznamujeme so základnou terminológiou z teórie grafov a definujeme ústredné pojmy, ktoré budú potrebné pri pochopení jednotlivých algoritmov a pri neskoršom porovnávaní získaných výsledkov. V druhej, tretej a štvrtej kapitole sa venujeme jednotlivým vybraným algoritmom, vysvetľujeme ich hlavnú motiváciu a podrobný proces výpočtov na malých príkladoch. Nechýba ani informovanie o príkazoch z knižníc softvéru R, ktorými aplikujeme algoritmy na zadanú sieť. Pre lepšiu všeobecnú predstavu v piatej kapitole v skratke predstavíme ďalšie možné metódy riešiace danú problematiku. V poslednej kapitole aplikujeme vybrané algoritmy na reálne sociálne siete a výsledné rozdelenia porovnáme medzi sebou i s reálnym rozdelením siete podľa jej charakteru a vyvodíme z toho konkrétne závery.

1 Definovanie pojmov

Na grafickú reprezentáciu sietí sa typicky využívajú grafy pozostávajúce z vrcholov, ktoré sú poprepájané hranami. Vrcholy predstavujú skúmané objekty a hrany reprezentujú vzťahy medzi nimi. Základy teórie grafov postavil významný matematik Leonhard Euler v roku 1736, keď sa venoval problematike siedmych mostov v pruskom meste Königsberg a dokázal neexistenciu možnosti prejdenia cez všetky mosty práve raz.

Grafmi je možné modelovať rôzne siete, od biochemických opisujúcich interakcie medzi proteínmi po siete internetových stránok navzájom poprepájaných hyperlinkami. Častými príkladmi sociálnych sietí býva mailová korešponedncia medzi pracovníkmi vo firme, sieť citácií medzi skupinami vedcov z viacerých vedných odborov, či jednoducho sieť vzťahov vrámci nejakej inštitúcie, klubu. Takým príkladom je aj dobre známa sieť vzťahov v "Zacharyho karate klube" z článku [18]. Ilustrujeme ju na obrázku 1 čerpanom zo stránky [3].



Obr. 1: Zacharyho karate klub

Teraz sa oboznámime so základnými pojmami a označeniami z oblasti teórie grafov. Graf G budeme označovať $G(V, E)$, kde pod V rozumieme množinu vrcholov grafu a E označuje množinu hrán medzi jeho vrcholmi. Pod zápisom $V(G)$, resp. $E(G)$ budeme chápať vektor vrcholov, resp. množinu hrán prislúchajúcu grafu G. Hranu e budeme označovať aj (u, v), kde u je vrchol, z ktorého hrana vychádza a vrchol v, do ktorého smeruje.

Dôležitým aspektom môže byť v sociálnych sieťach aj fakt, že každý vzťah, resp.

väzba, spájajúca dvoch ľudí je iná. Tým pádom aj tok informácií predávaných medzi ľudmi je nerovnomerný. U ľudí, ktorý sa stretávajú častejšie a zdieľajú podobné názory a teda spolu možno viac komunikujú, sa jednotlivé informácie predajú s vyššou pravdepodobnosťou ako medzi ľudmi, ktorí spolu interagujú menej. Je veľa rôznych možných prístupov, akým spôsobom možno "silu"daného vzťahu odmerať. Napríklad v prípade spomenutého karate klubu to bolo sčítaním spoločných sociálnych interakcií daných dvoch členov mimo klubu [18]. V reči grafov pridelíme každej hrane kladnú hodnotu $c(u, v) > 0$. Nazývame ju *kapacitou hrany*. V takom prípade ide o *(hranovo) ohodnotený* graf. Občas sa zvyknú kapacity reprezentovať *viacnásobnými hranami* . Prakticky to znamená, že dané dva vrcholy spája toľko hrán, koľko bola hodnota kapacity príslušnej hrany. Pri sieťach s vysokým počtom vrcholov sa však pre jednoduchosť niekedy zvykne predpokladať len jednotná sila väzby a teda kapacity hrán budeme zanedbávať resp. nastavíme každej hodnotu 1. V taktomto prípade sa jedná o *neohodnotený* graf.

Taktiež môžeme grafy rozdeliť na *orientované* a *neorientované*. Orientované majú, ako už z názvu vyplýva, jasne danú orientáciu každej hrany. V neorientovanom grafe predpokladáme symetriu vzťahov - teda ak existuje hrana vychádzajúca z vrcholu a smerujúca do vrcholu b, tak existuje aj hrana vedúca z b do a s rovnakou kapacitou. Často sa na reprezentáciu sociálnych sietí využívajú práve neorientované grafy. Slovenská terminológia a označenia sú prevzaté z [11].

Vyššie sme už spomenuli pojem tok informácií medzi ľudmi. V teórií grafov je pojem tok bežne zaužívaný a bude hrať jednu z ústredných úloh vo Ford-Fulkersonovom algoritme. Tok, ktorý ideme študovať bude mať vždy svoj *zdroj (source)*, z ktorého bude vychádzať a *ústie (sink)*, v ktorom zaniká. Zväčša sa zdroj označuje s a ústie t. Uvádzame nasledovnú definíciu, prebratú z [9]:

Definícia 1.1. *Tok v sieti (G, s, t, c) znázorňuje priradenie nezápornej hodnoty $f(u, v)$ každej hrane $(u, v) \in E(G)$, pričom platí:*

- *pre každú hranu $(u, v) \in E(G)$:*

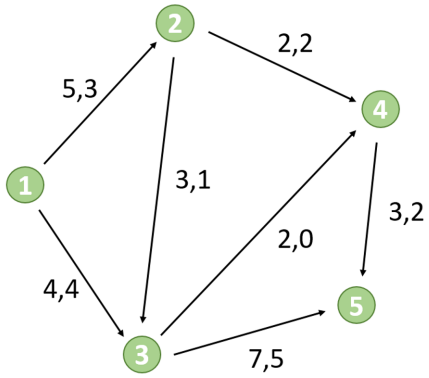
$$f(u, v) \leq c(u, v), \tag{1}$$

- *pre každý vrchol $v \in V(G)$ okrem vrcholov s a t platí:*

$$\sum_{u \in V} f(u, v) = \sum_{w \in V} f(v, w). \tag{2}$$

Veľkost toku označujeme ako $W(f) := \sum_{v} f(s, v)$, kde s je zdrojom. Ak sa veľkosť toku v hrane (u, v) rovná svojej kapacite, hranu nazývame nasýtenou.

Pre lepšiu predstavu si uveďme príklad. Na obrázku 2 demonštrujeme jeden z možných tokov, ktorý daným grafom môže prúdiť. Vrchol č.1 je jeho zdrojom a vrchol označený číslom 5 je jeho ústím. Dvojice čísel udávajú veľkosť kapacity a toku v danej hrane.



Obr. 2: Príklad toku v grafe

Pod pojmom *cesta* budeme rozumieť takú postupnosť hrán, vychádzajúcej z vrchola s a končiacej vo vrchole t, pričom sa dodržuje orientácia hrán.

Ďalším významným pojmom je $s - t$ rez v sieti. Uvedieme jeho definíciu, ktorú preberáme z [9]. Taktiež vysvetľuje spôsob výpočtu jeho kapacity a toku v ňom.

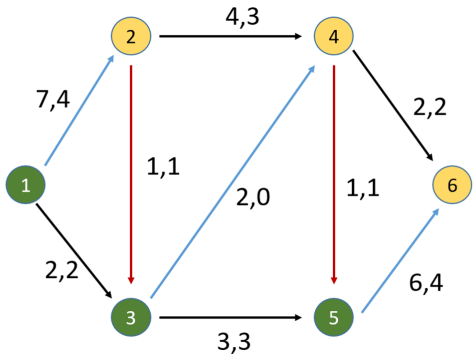
Definícia 1.2. *Rez v sieti $G(V, E)$ je rozdelenie množiny vrcholov $V(G)$ na dve množiny S a T také, že zdroj $s \in S$ a ústie $t \in T$. Kapacita rezu sa rovná súčtu kapacít hrán, ktoré smerujú z množiny S do množiny T:*

$$c(S, T) = \sum_{u \in S, v \in T} c(u, v). \tag{3}$$

Veľkosť toku v $s-t$ reze určíme ako súčet tokov v hranách smerujúcich z množiny S do množiny T mínus súčet tokov v hranách, smerujúcich opačne:

$$f(S) := \sum_{a \in S, b \in T} f(a,b) - \sum_{b \in T, a \in A} f(b,a). \tag{4}$$

Na obrázku 3 je znázornený $s-t$ rez, kde vrcholy $\{1,3,5\} \in S$ a vrcholy $\{2,4,6\} \in T$. Kapacitu rezu vieme určiť podľa vyššie uvedeného vzorca ako $c(S,T) = 7+2+6 = 15$ a veľkosť toku prechádzajúci rezom ako $f(S) = 4+0+4-1-1 = 6$.



Obr. 3: Ilustrácia $s-t$ rezu v grafe

Pojem rez sa však dá chápať oveľa komplexnejšie. Pri problematike detekcií komunít bude našou snahou skúmanú sieť rozdeliť podľa predpokladaných komunít (klastrov). Prirodzene môže ísť aj o viaceré, nielen dva. Takto vzniká *rozdelenie* siete, teda začlenenie jednotlivých vrcholov do klastrov. Veľkosť rezu, ktorým sme takéto rozdelenie vytvorili sa dá chápať všeobecne ako súčet kapacít hrán, ktoré sme rozdelením "prerezali".

Na porovnanie kvality jednotlivých nájdených rozdelení siete budeme často používať tzv. *funkciu kvality*, ktorá priradí každému rozdeleniu siete nejakú hodnotu. Pod kvalitou rozdelenia si treba predstaviť, nakoľko nájdené zoskupenia vrcholov odpovedajú očakávaným vlastnostiam komunít. Treba si ujasniť, že pojem komunita nie je explicitne matematicky zadefinovaná, je chápaná skôr intuitívne, viac v podsekcii 2.1. Z tohto dôvodu existuje aj viacero spôsobov, ako definovať funkciu kvality. My uvedieme nasledovné dve z článku [4] a tretiu, vysvetlenú v článku [7, str. 15].

Definícia 1.3. *Pod pojmom coverage (ponechávame anglický názov) rozumieme takú funkciu kvality, že jej hodnota pre rozdelenie P sa rovná*

$$Cov(P) = \frac{\sum_{i,j} A_{ij}\delta(C_i, C_j)}{\sum_{i,j} A_{ij}}, \tag{5}$$

kde funkcia $\delta(C_i, C_j)$ vracia hodnotu 1, ak vrchol i a j pochádzajú z rovnakého klastru, inak vracia hodnotu 0. A je maticou susednosti skúmaného grafu, t.j. maticou $n \times n$, ktorej člen $a_{ij} = 1$, ak existuje hrana medzi i-tym a j-tym vrcholom, inak sú členy nulové. Voľne povedané coverage predstavuje podiel vnútroklastrových väzieb ku všetkým.

Táto definícia funkcie kvality má veľkú nevýhodu, pretože robí nasledovnú vec. Po zlúčení ktorýchkoľvek dvoch, aspoň jednou hranou prepojených klastrov nejakého rozdelenia, tomuto novému rozdeleniu po zlúčení bude automaticky priradená vyššia hodnota, nehľadiac na to, či dané zlúčenie "pomohlo"pri detekovaní komunít. Všimnime si, že ak by sme priradili všetky vrcholy do jedného klastra, toto rozdelenie by dosiahlo mieru *coverage* rovnú 1, čo je maximum. To však neznamená, že ide o optimálne rozdelenie.

Definícia 1.4. *Pod pojmom conductance (vodivosť) klastra C_k sa vo všeobecnosti rozumie*

$$\phi(C_k) = \frac{\sum_{i \in C_k, j \notin C_k} A_{ij}}{min(A(C_k), A(\overline{C_k}))}, \tag{6}$$

kde A je maticou susednosti, $\overline{C_k} = V(G) - C_k$ a $A(C_k)$ vypočítame ako $\sum_{i \in C_k, j \in V(G)} A_{i,j}$. Čím nižšiu hodnotu dostávame, tým menej je daný klaster prepojený so zvyškom grafu. Takýto klaster teda výzerá mať dobrý potenciál na silné komunitné črty. My budeme pod pojmom conductance rozdelenia P rozumieť takú funkciu kvality, ktorej predpis je

$$\phi(P) = 1 - \frac{1}{k}\sum_{k} \phi(C_k), \tag{7}$$

čiže priemer vodivosti klastrov odčítaný od 1. Týmto definovaním dostávame hodnotu v rozmedzí $0-1$, kde však platí, že tentoraz vyššia hodnota bude signalizovať kvalitnejšie rozdelenie do komunít, pretože vodivosť medzi klastrami dosahuje v priemere nižšie hodnoty.

Predstavíme aj ďalší spôsob, ako definovať funkciu kvality, ktorá bude neskôr pre niektoré algoritmy kľúčová. Jeho hlavnou myšlienkou je, že od náhodného grafu sa vo všeobecnosti neočakáva, že bude mať črty zoskupovania vrcholov do komunít. Budeme špeciálnym spôsobom porovnávať hustotu[1] hrán podgrafov skúmanej siete (od ktorých prirodzene, ak sa jedná o komunity očakávame veľkú hustotu) s hustotou rovnakých podgrafov avšak v náhodnom grafe.

V náhodnom grafe si treba pod jeho hranami predstaviť všetky možné hrany (aj také, ktoré vedú z jedného vrcholu späť do toho istého), ktorých kapacity sú rovné pravdepodobnostiam vzniku daných hrán. Práve tieto pravdepodobnosti budú ďalej využité.

Definícia 1.5. *Funkcia kvality, ktorá je založená na spomínanom porovnaní hustôt hrán v analyzovanom a náhodnom grafe sa nazýva modularita a je definovaná nasledovným predpisom:*

$$Q = \frac{1}{2m} \sum_{i,j} (A_{ij} - P_{ij}) \delta(C_i, C_j), \tag{8}$$

kde m je počet hrán v grafe, A je maticou susednosti a hodnota P_{ij} zodpovedá pravdepodobnosti, s ktorou vznikne hrana medzi i a j v náhodnom grafe.

Je viacero spôsobov, ako zvoliť nulový model (spôsob tvorby náhodného grafu). Najčastejším býva *konfiguračný model* (angl. *configuration model*). Na to, aby v takom náhodnom grafe vznikla hrana, musia sa stretnúť medzi danými dvoma vrcholmi tzv. *polovičné hrany* (anlg. *stubs*). V pôvodnom grafe ich máme $2m$. Premenná k_i, resp. k_j prezentuje *stupeň* vrchola i, resp. j, t.j. počet hrán spájajúcich daný vrchol s ostatnými. Pravdepodobnosť vzniku hrany spojenej s vrcholom i sa rovná $p_i = \frac{k_i}{2m}$. Pravdepodobnosť vzniku hrany medzi vrcholami i a j sa rovná $2mp_ip_j = \frac{k_ik_j}{2m}$. Vzorec na výpočet modularity tak môžeme prepísať na:

$$Q = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{k_ik_j}{2m} \right) \delta(C_i, C_j). \tag{9}$$

[1] *Hustotu* grafu môžeme vo všeobecnosti definovať pomerom počtu hrán v danom grafe ku počtu hrán, ktoré by mohli v danom grafe existovať - tzn. v prípade, že všetky vrcholy sú navzájom pospájané. Všimnime si, že takto definovaná hustota je relevantná len v prípade neohodnotených grafov.

Daný výpočet sa dá využiť iba ak sa jedná o neohodnotený graf. Newman vo svojom neskoršom článku [12, str. 7] uvádza, že jednoduchým zovšeobecnením ho vieme prispôsobiť aj pre ohodnotené grafy:

$$Q = \frac{1}{2w} \sum_{i,j} \left(w_{ij} - \frac{w_i w_j}{2w} \right) \delta(C_i, C_j), \tag{10}$$

kde w_{ij} predstavuje kapacitu hrany medzi vrcholom i a j, w_i vypočítame ako súčet kapacít hrán, ktoré vychádzajú z vrchola i a premenná w sa rovná súčtu kapacít všetkých hrán. Analogicky sa dá upraviť aj vzorec pre výpočet hodnoty *coverage* a *conductance*.

Na meranie zhody dvoch rôznych rozdelení jednej siete budeme využívať nasledovný vzorec, uvedený v [7, str. 78]:

Definícia 1.6. *Randov index, merajúci podobnosť dvoch rozdelení X a Y je definovaný ako*

$$R(X, Y) = \frac{a_{11} + a_{00}}{a_{11} + a_{00} + a_{10} + a_{01}} , \tag{11}$$

kde a_{11} predstavuje počet dvojíc vrcholov, ktoré boli v oboch rozdeleniach priradené do rovnakého klastru. Premenná a_{00} prezentuje počet dvojíc vrcholov, ktoré v oboch rozdeleniach boli zaradené do rozdielnych klastrov. Premenná a_{10} vyjadruje počet dvojíc, ktoré v rozdelení X boli v spoločnom a v rozdelení Y rozdielnom klastri a premenná a_{01} analogicky opačne.

2 Ford-Fulkersonov algoritmus

2.1 Motivácia

Našim prvým algoritmom, ktorým sa budeme zaoberať, sa nazýva Ford-Fulkersonov (skrátene FF). Aplikácií daného algoritmu na sociálnu sieť a následnému skúmaniu fenoménu komunít sa venoval W. Zachary vo svojom článku [18]. Pozoroval vzťahy medzi členmi istého karate klubu po dobu troch rokov. Každý vzťah dvoch jedincov ohodnotil hodnotou, ktorá odhadovala silu vzájomného puta - teda každej hrane v grafe priradil jej kapacitu. Urobil to na základe počtu spoločných sociálnych interakcií, do ktorých sa členovia zapájali mimo aktivít samotného klubu. V klube došlo ku konfliktu medzi prezidentom klubu a jedným inštruktorom karate. Klub sa nakoniec aj oficiálne rozdelil na dva. Zachary pomocou algoritmu odhadol príslušnosť jednotlivých členov k jednej (skupina podporujúca prezidenta klubu) alebo druhej (skupina uznávajúca názor inštruktora) komunite. Tú následne porovnal s oficiálnym rozdelením do dvoch klubov.

FF algoritmus bol pôvodne zkonštruovaný na hľadanie maximálneho toku zo zdroja do ústia. Typickým príkladov využitia je nájdenie maximálneho množstva vody, ktorý môže pretiecť určitým systémom potrubí za nejakú časovú jednotku. Avšak hodnota maximálneho toku v sieti sa rovná kapacite minimálnemu rezu (toto tvrdenie poznáme pod názvom *Max Flow - Min Cut* a neskôr uvedieme aj jeho dôkaz). Tento rez v sieti oddeľuje zdroj od ústia. Ako výstup dostávame dve rozdelené množiny vrcholov - dve komunity. V prípade karate klubu bolo vhodné označit riaditeľa klubu ako zdroj a daného inštruktora ako ústie, resp. naopak, pretože cieľom bolo začleniť ich do opačných komunít.

Ako sme už spomínali v predošlej kapitole, pojem komunita nie je exaktne definovaný. Je však prirodzené očakávať, že k optimálnemu rezu, z hľadiska oddelenia dvoch komunít v sieti dochádza v najslabších prepojeniach medzi ľudmi. V rámci komunity očakávame totižto silno poprepájanú sieť vzťahov a naopak slabé prepojenia s objektami mimo komunity. A teda pri oddeľovaní jednej komunity od druhej bude našou snahou "pretrhnúť"čo najslabšie a čo najmenej vzťahov - čo matematicky zodpovedá snahe nájsť najmenšiu kapacitu rezu.

Existuje aj ďalší aspekt, prečo by sme mali týmto spôsobom postupovať. Častokrát je zo strany komunít nežiadúce, aby sa druhá skupina dozvedela informácie kolujúce v ich komunite. Môže ísť napríklad o nejaké tajomstvo, ohováranie alebo politické stratégie ako to bolo v prípade karate klubu - informácia, kedy sa bude konať zasadanie, na ktorom sa rozhodovalo o dôležitých záležitostiach klubu. Tým pádom sa komunity prirodzene snažia minimalizovať šancu, že sa informácia dostane "von". Avšak ak sa informáciu dozvedela druhá komunita, je jasné, že im ju musel zdeliť niekto, kto mal s druhou komunitou nejakú väzbu.

V prípade karate klubu algoritmus preukázal slušnú úspešnosť, až 33 z 34 členov zadelil správne. Zdá sa, že by FF algoritmus mohol dobre fungovať pri hľadaní rozdelenia siete do dvoch komunít, najmä keď sa máme k dispozícii odhadnuté sily väzby medzi jednotlivými objektami siete, čiže sú určené kapacity hrán. Ako vstupom však treba určiť dva objekty, o ktorých je žiadané, aby boli začlenené do opačných komunít.

2.2 Vysvetlenie princípu

V prvom rade sa postupne cez pomocné lemy 2.1 a 2.2 dopracujeme k dôkazu tvrdenia *Max Flow - Min Cut* a potom bližšie popíšeme, akým spôsobom FF algoritmus funguje. Dôkazy jednotlivých lém a tvrdenia sú prebraté z [9].

Lema 2.1. *Veľkosti toku f(S) prechádzajúci ľubovoľným s-t rezom v sieti (G,s,t,c) sa rovná veľkosti celkového toku v sieti:*

$$f(S) = W(f). \tag{12}$$

Dôkaz:

Uvažujme nasledovný výraz

$$A := \sum_{v \in V} f(s,v) - \sum_{a \in S, a \neq s} \left(\sum_{v \in V} f(v,a) - \sum_{w \in V} f(a,w) \right) - \sum_{a \in S, b \in T} f(a,b) + \sum_{b \in T, a \in S} f(b,a). \tag{13}$$

Výraz A je v skutočnosti rovný 0. Po dlhšom uvážení sa dá prísť na to, že každá hodnota $f(u,v)$, pričom aspoň jeden z vrcholov u, v patrí množine S , sa vo výraze vyskytuje práve dvakrát - raz s kladným znamienkom a raz so záporným. Ďalej si

uvedomme, že

$$\sum_{a\in S, a\neq s}\left(\sum_{v\in V} f(v,a) - \sum_{w\in V} f(a,w)\right) = 0, \tag{14}$$

pretože sa môžeme odvolať na vlastnosť toku z rovnice (2) z jeho definície. Z toho už vyplýva nasledovná rovnosť:

$$W(f) := \sum_{v\in V} f(s,v) = \sum_{a\in S, b\in T} f(a,b) - \sum_{b\in T, a\in S} f(b,a) = f(S). \tag{15}$$

□

Všimnime si zaujímavý fakt. Uvažujme hrany, nachádzajúce sa v reze, t.j. také (u,v), že $u \in S$ a $v \in T$. Keby sme ich odstránili zo siete, množiny S a T by už viac neboli prepojené. Treba si však uvedomiť, že zdroj toku patrí do množiny S a ústie do T. To znamená, že keď chceme nejaký tok dostať do ústia, celý tento tok musí prechádzať hranami v reze. Z toho vyplýva, že maximálny tok bude zhora ohraničený súčtom kapacít hrán v reze. O tom pojednáva aj nasledovná lema.

Lema 2.2. *Pre každú sieť (G,s,t,c), každý tok f v tejto sieti a každý $s-t$ rez platí:*

$$\sum_{v\in V} f(s,v) \leq \sum_{a\in S, b\in T} c(a,b). \tag{16}$$

Dôkaz: Platnosť vieme jednoducho ukázať pomocou lemy 2.1 a rovností (1) a (4):

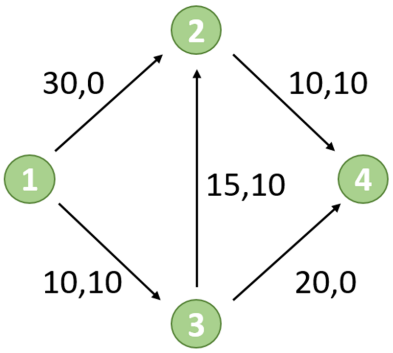
$$W(f) = \sum_{v\in V} f(s,v) = f(S) \leq \sum_{a\in S, b\in T} f(a,b) \leq \sum_{a\in S, b\in T} c(a,b) = c(S,T). \tag{17}$$

□

Lema 2.2 nám však implikuje, že ak nájdeme tok s veľkosťou rovnou kapacite nejakého s-t rezu, daný tok je maximálny možný.

Stále však ostáva otázkou, ako maximálny tok hľadať? Podľa čoho iného ešte vieme zistiť, či sme ho už našli? Čo ak na začiatku zvolíme nejakým spôsobom "zlú"cestu?

Predstavme si jednoduchý príklad siete na obrázku 4, kde vrchol 1 predstavuje zdroj a vrchol 4 ústie. Na začiatku sme zvolili cestu z vrchola $1 \to 3 \to 2 \to 4$. Veľkosť toku prechádzajúci touto cestou sme určili tak, aby daný tok vedel prejsť každou hranou cesty, teda aby kapacita všetkých hrán bola väčšia alebo rovná množstvu toku. Zároveň



Obr. 4: Tok v grafe, ktorý na prvý pohľad nevieme navýšiť

sme sa snažili túto veľkosť maximalizovať. Zvolili sme teda veľkosť toku 10. Všimnime si, že ostatné dve možné cesty z 1 do 4 - cesty 1→2→4 a 1→3→4 obsahujú nasýtenú hranu, čo znamená, že pokiaľ nejakým spôsobom nezmeníme momentálne prúdiaci tok v sieti, nevieme ho viac navýšiť. Pravdou však je, že keby sme na začiatku zvolili radšej cesty 1→2→4, 1→3→4 o napr. veľkosti 7, už by sme dosiahli väčší tok rovný 14.

Pri riešení takýchto situácií bude potrebný takzvaný *reziduálny graf* . Tento graf možno vo všeobecnosti chápať ako graf „možností“, čo sa dá vykonať v pôvodnom grafe s doterajším tokom. Jednou z požiadaviek naňho je, aby nám umožnil spätne vrátiť tok hranou, ktorá sa ukazuje, že nebola optimálne zvolená a poslať ho inou cestou. Zároveň budeme vyžadovať, aby ukázal, o koľko jednotiek toku sa dá tok v danej hrane navýšiť, teda akú má ešte dostupnú, nevyužitú kapacitu. Spomenuté požiadavky vieme zhrnúť do matematického zápisu, ktorý nám určuje tzv. *rezervy* jednotlivých hrán (u, v) v danej sieti predpisom:

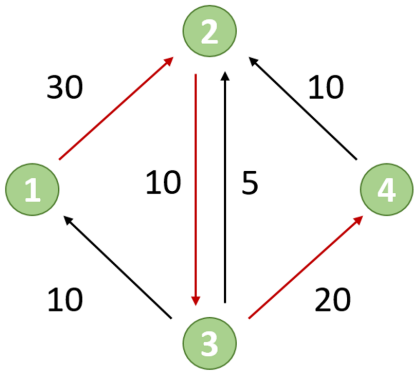
$$r(u, v) = c(u, v) - f(u, v) + f(v, u). \tag{18}$$

Je dôležité dodať, že reziduálny graf vytvára a určuje rezervy aj hranám opačnej orientácie, ako boli v pôvodnej sieti. Práve tieto hrany budú umožňovať aktuálny tok vracať istým spôsobom späť. Hrany s nulovou rezervou sa do grafu nezaznačujú.

Pod pojmom *nenasýtená polocesta* v reziduálnom grafe budeme rozumieť takú cestu, spájajúcu vrchol s s vrcholom t, ktorej rezerva každej hrany polocesty je kladná. *Rezervou polocesty* nazývame najmenšiu rezervu hrany spomedzi všetkých hrán polocesty.

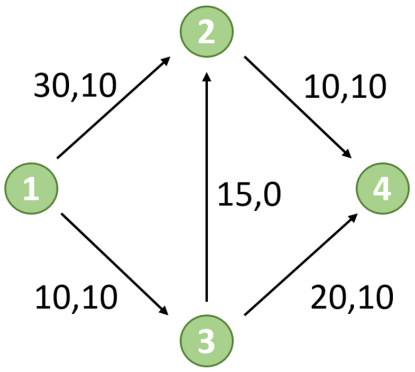
Vráťme sa k predošlej situácií z obrázka 4, kde na prvý pohľad nebolo jasné, ako po-

kračovať. Zostrojme príslušný reziduálny graf s vypočítanými rezervami podľa rovnice (18) - viď obrázok 5. Červenou farbou je vyznačená nenasýtená polocesta P. Rezervou tejto polocesty bude hodnota 10.



Obr. 5: Reziduálny graf

Akú úlohu má nenasýtená polocesta pri navýšení aktuálneho toku? Ona určuje, v ktorých hranách budeme tok meniť. O akú hodnotu ho zmeníme, to určuje rezerva tejto polocesty. Všimnime si, že cez hrany $(1, 2)$ a $(3, 4)$ vieme poslať 10 jednotiek toku bez problémov. K nejasnostiam prichádza pri hrane $(2, 3)$ v reziduálnom grafe. K nej totiž neexistuje príslušná hrana v pôvodnom. Existuje iba opačne orientovaná. Nenasýtená polocesta P poukazuje na to, že by bolo optimálne poslať 10 jednotiek toku z vrcholu 2 do vrcholu 3. To je ale rovné situácií, že by sme aktuálny tok v opačnom smere o 10 jednotiek znížili. Urobme tak a dostávame nový pozmenený tok, viď obrázok 6.



Obr. 6: Nový tok

Po opätovnom skonštruovaní reziduálneho grafu sa ľahko presvedčíme, že v ňom už neexistuje nenasýtená polocesta. Čo to však pre nás znamená? Naozaj sme už dostali maximálny tok, alebo sa dá navýšiť ešte iným spôsobom?

Týmto sa dostávame sa k záverečnému tvrdeniu, ktoré nám zodpovie na naše otázky. Uvádzame jeho znenie s miernou modifkáciou dôkazu, uvedenom v [9].

Tvrdenie 2.3. ***Max Flow - Min Cut (Ford-Fulkerson - 1956)***

Nasledovné tri podmienky pre tok f v sieti (G, s, t, c) sú ekvivaletné:

1. *existuje $s-t$ rez, ktorého kapacita $c(S,T) = W(f)$*
2. *tok f je maximálny*
3. *neexistuje nenasýtená polocesta v reziduálnom grafe*

Dôkaz:

1. $\Rightarrow$ 2.

Platnosť tejto implikácie sme už objasnili pri *leme 2.*

2. $\Rightarrow$ 3.

Keď existuje nenasýtená polocesta, upravujeme toky v hranách podľa príslušných pravidiel v podrobnom popise algoritmu nižšie. Všimnime si, že k toku v začiatočnej hrane tejto polocesty štartujúcej zo zdroja sa príslušné r_{min} (rezerva nenasýtenej cesty reziduálneho grafu) pripočíta. Je tomu tak preto, že daná hrana je súhlasne orientovaná ako hrana v pôvodnom grafe a teda zo zdroja vychádza. Ak by aj existovala hrana, ktorá v pôvodnom grafe do vrchola s vchádza, nikdy nebude využitá, lebo v našej situácií nemá zmysel posielať tok späť do zdroja. Z toho vyplýva, že ak existuje nenasýtená polocesta v reziduálnom grafe, veľkosť toku by sme vedeli navýšiť

$$W(f) = \sum_{v} f(s,v) + r_{min} \tag{19}$$

a z toho vyplýva, že pôvodný tok nebol maximálny.

3. $\Rightarrow$ 1.

Ostáva dokázať, že ak neexistuje nenásytená polocesta v reziduálnom grafe, tak veľkosť toku sa rovná kapacite nejakého $s-t$ rezu.

Nech f je taký tok, pre ktorý je bod *3.* splnený. Uvažujme takú množinu S, ktorá obsahuje vrcholy, do ktorých sa dá z vrcholu s dostať pomocou kladných rezerv hrán v reziduálnom grafe, Vrchol s tiež radíme do množiny S. Potom platí, že vrchol $s \in S$ kvôli samotnej definícií množiny a $t \notin S$, lebo inak by existovala nenasýtená polocesta. Množinu S môžeme chápať ako rez. Všimnime si, že všetky hrany (a, b), ktoré sa nachádzajú v $s - t$ reze ($a \in S$, $b \in T$) musia mať nulovú rezervu v reziduálnom grafe, inak by vrchol $b \in S$. To podľa vzťahu (18) znamená, že

$$c(a, b) - f(a, b) + f(b, a) = 0. \tag{20}$$

Vieme, že $f(a, b) \leq c(a, b)$ a $f(b, a) \geq 0$, takže jediným spôsobom ako dostať rovnosť s nulou je

$$f(a, b) = c(a, b), \tag{21}$$

$$f(b, a) = 0. \tag{22}$$

Už si stačí uvedomiť nasledovné vzťahy:

$$W(f) = f(S) = \sum_{a \in S, b \in T} f(a, b) - \sum_{b \in T, a \in S} f(b, a) = \sum_{a \in S, b \in T} c(a, b) = c(S, T). \tag{23}$$

□

Nasleduje zrhnutie FF algoritmu:

- vlož danú sieť $(G = (V, E), s, t, c)$
- nastav $\forall (u, v) : f(u, v) := 0$
- vytvor reziduálny graf, vypočítaj rezervy pre všetky hrany $(u, v) \in E(G)$ a aj pre hrany opačné (v, u)
- pokiaľ existuje nenasýtená polocesta P z s do t
  - vypočítaj r_{min}, čo sa rovná rezerve polocesty P
  - definuj f_{new} ako:
    - $f_{new}(u, v) = r_{min}$, ak $(u, v) \in P \wedge (u, v) \in E(G)$

    * $f_{new}(u,v) = -r_{min}$, ak $(v,u) \in P \wedge (v,u) \notin E(G)$
    * $f_{new}(u,v) = 0$, inak
  - vypočítaj nový tok $\forall(u,v) : f(u,v) = f(u,v) + f_{new}(u,v)$
  - vypočítaj nové rezervy hrán v reziduálnom grafe
- vráť tok f

Vďaka tomuto tvrdeniu vieme, že FF spoľahlivo nájde maximálny tok, pričom v príslušnom reziduánom grafe už neexistuje nenasýtená polocesta zo zdroja k ústiu. Po vypočítaní maximálneho toku v sieti ešte ostáva nájsť minimálny rez a rozdeliť vrcholy do príslušných dvoch množín - hľadaných komunít. Tak, ako sme v dôkaze tvrdenia 2.3 v tretej časti spomenuli, do množiny S priradíme všetky vrcholy, do ktorých sa dá dostať zo zdroja pomocou kladných rezerv hrán vo finálnom reziduálnom grafe . Do množiny T priradíme zvyšné. Môže sa však stať, že rezov s minimálnou kapacitou je v sieti viac. Na overenie jedinečnosti, ako je uvedené v článku [18], je možné napríklad vymeniť zdroj s ústím a algoritmus spustiť znovu. Ak sa nájde rovnaké rozdelenie, optimálny rez je jediný (toto platí len pre neorientované grafy).

Sociálne siete sa však vo väčšine prípadov zvyknú reprezentovať neorientovanými grafmi. Vysvetlený algoritmus sa dá jednoducho aplikovať aj na tieto grafy. Stačí každú neorientovanú hranu reprezentovať dvoma navzájom opačne orientovanými hranami. Na takýto graf už možno aplikovať vysvetlené postupy. V ďalších kapitolách sa už budeme primárne zaoberať neorientovanými grafmi.

2.3 Kód a výstup v R

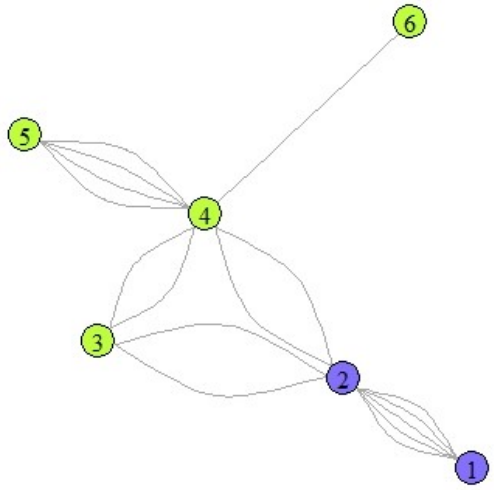
Knižnice softvéru R obsahujú mnoho naprogramovaných algoritmov pre hľadanie komunít v sieťach. Výnimkou nie je ani Ford-Fulkersonov algoritmus. Na ukážku ho budeme demonštrovať na malej sieti pozostávajúcej zo šiestich vrcholov:

```
library(optrees)
nodes <- 1:6
arcs <- matrix(c(1,2,5,  2,3,2,  2,4,2,  4,5,4,  4,6,1  ,  3,4,2),
        byrow=TRUE,  ncol = 3)
```

```
findMinCut(nodes, arcs, algorithm = "Ford-Fulkerson",
source.node = 1, sink.node = 5, directed = FALSE)
```

Informácie o príkazoch sme čerpali z [15]. Prvým vstupom príkazu `findMinCut` z balíka `optrees` je vektor `nodes`. Reprezentuje množinu vrcholov grafu. Druhým vstupom je matica `arcs`. Každý jej riadok predstavuje jednu väzbu. Má 3 stĺpce. Čísla v prvých dvoch indikujú, medzi ktorými dvomi vrcholmi existuje hrana. Tretí stĺpec vyjadruje kapacitu danej hrany. Za `source.node` označíme zdroj toku a za `sink.node` ústie. Ak sa jedná o orientovaný graf, argument `directed` nastavíme na `TRUE`, inak sa graf primárne berie ako neorientovaný. Ako výstup dostávame:

```
$s.cut
[1] 1 2

$t.cut
[1] 3 4 5 6

$max.flow
[1] 4

$cut.set
     ept1 ept2 weight
[1,]    2    3      2
[2,]    2    4      2
```



Obr. 7: Vizuálne znázornenie siete a nájdených komunít podľa FF algoritmu

Pre lepšie ilustráciu siete a získaných komunít poskytujeme obrázok 7. Kapacitu hrán budeme odteraz na obrázkoch znázorňovať viacnásobnými hranami. Spôsob vykreslovania grafov v R demonštrujeme v ďalšej kapitole v podsekcií 3.3.

3 Girvan-Newmanov algoritmus

3.1 Motivácia

Doteraz sme sa venovali Ford-Fulkersonovmu algortimu, ktorý má však jednu veľkú nevýhodu: Rozdeľuje sieť iba na 2 komunity. Taktiež treba zadať ako vstup, ktorý vrchol chceme určiť ako zdroj a ktorý ako ústie. V prípade karate klubu to bolo jednoduché rozhodnutie a na danú situáciu sa algoritmus dal bez problémov aplikovať. V sociálnych sieťach, hlavne s veľkým počtom vrcholov, však často vzniká väčší počet komunít a neraz sú aj hierarchicky organizované. Dobrým príkladom môže byť škola, v ktorej sú žiaci rozdelení do ročníkov, ročníky do tried a v triedach môžeme tiež nájsť určité "skupinkovanie". V takýchto prípadoch je dobré použiť algortimy, uspôsobené na hierarchické zhlukovanie siete.

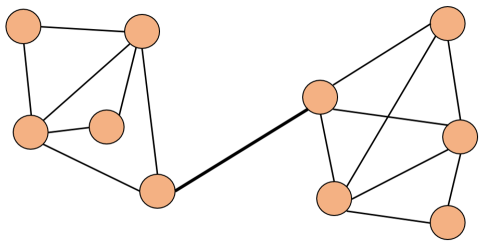
Medzi takéto algoritmy patrí aj trieda tzv. *rozkladných* (ang. *divisive*). Ich základnou myšlienkou je odstraňovanie hrán z grafu, o ktorých máme predpoklad, že sú medzikomunitné, teda spájajú dva rôzne zhluky vrcholov. Takéto hrany majú dôležitú úlohu z hľadiska štruktúry siete a majú vysokú centralitu (dôležitosť). Je známych viacero prístupov, ako centralitu hrany merať.

My sa v tejto kapitole budeme venovať Girvan-Newmanovmu algoritmu (ďalej už len GN), ktorý zohral v rozkladnej triede dôležitú úlohu. Dá sa o ňom dočítať napríklad v článku [7, str. 23] alebo [14]. Jeho podstatnou časťou je definovanie miery centrality hrany. Girvan a Newman navrhli tri spôsoby. My sa budeme zaoberať prvou z nich, ktorá nesie názov *edge betweenness* (ponechávame anglický názov, ďalej už len *betweenness*). Treba podotknúť, že bola pôvodne navrhnutá pre neohodnotené grafy, avšak neskôr si ukážeme, ako sa dá jednoducho transoformovať aj pre ohodnotené.

Táto miera nám pre každú hranu vyjadruje frekvenciu, s ktorou sa daná hrana vyskytuje v najkratších cestách, ktoré sa hľadajú pre všetky dvojice vrcholov grafu. Pod najkratšou cestou medzi dvoma vrcholmi prirodzene rozumieme cestu pozostávajúcu z najmenšieho počtu hrán, spájajúcu dané vrcholy. Takáto miera bola inšpirovaná *vertex betweenness*, vymyslená Freemanom v roku 1977 [8], ktorá sa rátala pre vrcholy grafu.

Ak hrana spája dva klastre, je prirodzené očakávať od nej veľkú mieru *betweenness*, pretože bude často súčasťou najkratších ciest medzi vrcholmi, z ktorých jeden vrchol je

v jednom klastri a druhý v druhom, viď obrázok 8 - hrubo vyznačená čiara spájajúca zjavne dve skupiny vrcholov bude mať vysokú centralitu.



Obr. 8: Ilustrácia hrany s vysokou centralitou

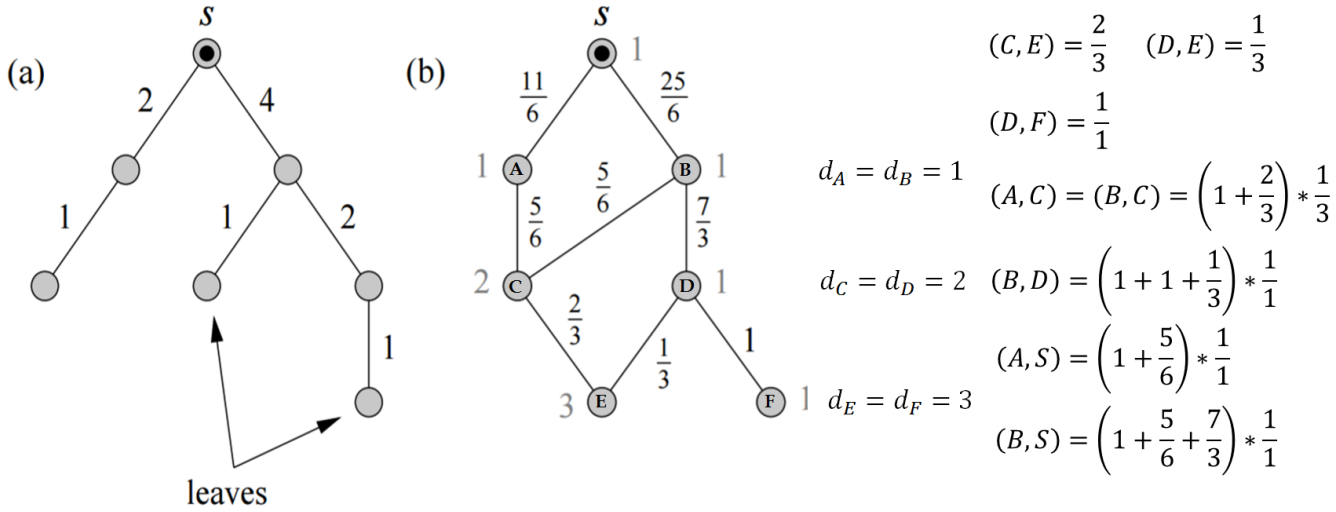
Avšak je dôležité si uvedomiť, že môže nastať prípad, že máme dve hrany spájajúce dve komunity - jedna z nich bude mať vysokú spomínanú mieru, pretože má v nejakom zmysle lepšiu pozíciu ako tá druhá, teda väčšina najkratších ciest bude prechádzať práve ňou. To môže spôsobiť, že druhá hrana, ktorá je tiež veľmi dôležitá, pretože spája dve komunity, nenadobúda vysokú hodnotu *betweenness*. Z toho dôvodu vždy po nájdení hrany s najvyššou hodnotou a jej odstránení musíme *betweenness* pre ostávajúce hrany znovu prepočítať. Pri danom postupe v ďalšej iterácií už bude táto druhá hrana nadobúdať vysokú hodnotu a je možné ju správne identifikovať ako medzikomunitnú.

Algoritmus bude teda každou iteráciou vymazávať jednu hranu a znova prepočítavať hodnotu *betweenness* pre všetky ostávajúce hrany. Ak nastane zhoda, môžu sa odstrániť všetky hrany s najvyššou hodnotou alebo sa vyberie náhodne jedna. Takýmto iteračným vymazávaním hrán budeme postupne graf rozpájať, počas čoho sa v niektorých iteráciach budú od seba oddelia skupiny vrcholov. Takto počas procesu vzniknú viaceré rozdelenia siete, až nakoniec dostaneme rozdelenie o n jednoprvkových komunitách.

3.2 Vysvetlenie algoritmu

Akým algoritmom sa dá *betweenness* vypočítať? Vysvetlíme metódu, ktorú opísali Girvan a Newman v článku [14].

Najprv vybereme jeden vrchol, označíme ako zdroj S a budeme hľadať najkratšie cesty z S do ostatných vrcholov. Tento proces výpočtu sme demonštrovali na malej sieti na obrázku 9. Tento obrázok sme prebrali z článku [14, FIG. 4]a doplnili ho o konkrétny postup pri výpočtoch, vysvetlený nižšie. V prípade, že sa v grafe nenachádza žiadny cyklus, hľadanie najkratších ciest a vypočítanie hodnoty *betweenness* je vcelku jednoduchý proces. Takýmto prípadom je situácia naznačená na obrázku 9(a). Situácia sa začne komplikovať, ak sa jedná o cyklický graf, viď obrázok 9(b) a môže vzniknúť viacero najkratších ciest. V takom prípade sa váha týchto ciest predeľuje ich počtom.



Obr. 9: Výpočet hodnoty *betweenness* pri určenom zdroji S

Pri výpočte budeme postupovať nasledovným spôsobom:

- začiatočnému vrcholu s priradť vzdialenosť $d_s = 0$ a váhu $w_s = 1$
- každému susediacemu vrcholu i priradť $d_i = d_s + 1 = 1$ a $w_i = 1$
- pre každý, vrchol j susediaci s predošlými vrcholmi i urob jednu z troch možností:
  - ak vrcholu j zatiaľ nebola priradená vzdialenosť, priradť $d_j = d_i + 1$ a váhu $w_j = w_i$
  - ak už vrcholu j bol priradená vzdialenosť a platí $d_j = d_i + 1$, potom navýš váhu $w_j = w_j + w_i$
  - ak už vrcholu j bol priradená vzdialenosť avšak $d_j < d_i + 1$, nič nerob

- Zopakuj od tretieho kroku, až kým všetky vrcholy majú priradené vzdialenosti.

Vypočítané váhy pri každom vrchole sú presne to čo potrebujeme. Vyjadrujú počet najkratších ciest zo zdroja k danému vrcholu i. Budú následne využité v ďalšom dopočítavaní:

- nájdi každý "konečný"vrchol t, cez ktorý už neprechádza nijaká najkratšia cesta z vrchola s do ľubovoľného vrcholu v grafe
- každej hrane vychádzajúcej z vrchola t smerujúcej do susedného vrchola i priraď hodnotu w_i/w_t
- začínajúc vrcholmi, ktoré sú najďalej od zdroja s (teda vrcholy najnižšie položené, ak graf zakreslíme schématicky ako na *obrázku 7*) pokračujúc smerom vyššie k zdroju postupne počítaj nasledovné: hrane spájajúcej vrchol i a j pričom i je bližšie k zdroju ako vrchol j priraď hodnotu 1 plus suma už priradených hodnôt hrán, ktoré spájajú daný vrchol j s inými vrcholmi z nižšej úrovni, celé prenásobené hodnotou w_i/w_j
- Opakuj od tretieho kroku pokým sa nedosiahne vrchol s

Už len treba zopakovať tento proces pre všetkých n vrcholov ako zdroje a sčítať všetky hodnoty danej hrany, ktoré v týchto výpočtoch nadubdla. Dostávame mieru *betweenness* pre každú hranu grafu.

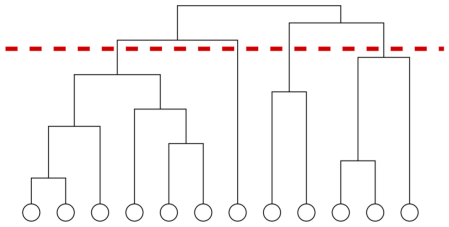
Tento algoritmus bol doteraz vysvetlený len pre neohodnotené grafy. Newman v článku [12] však ponúka hneď viacero prístupov, ako vhodne rozšíriť daný algoritmus aj pre tie ohodnotené. Jedným z nich bolo po vypočítaní hodnoty *betweenness* podľa vyššie uvedeného algoritmu jej jednoduché predelenie kapacitou danej hrany. Vzhľadom k tomu, že algoritmus ďalej odstraňuje v každej iterácií hranu s najvyššiou hodnotou *betweenness*, ide o snahu vyvarovať sa tak skorému vymazaniu hrany, ktorej by mala byť prikladaná výrazne vyššia váha, pretože predstavuje silnú väzbu medzi prílušnými objektami. Vzhľadom k tomu, že sa po predelení veľkou kapacitou hodnota *betweenness* vyrazne zmenší, spôsobí to pokles pravdepodobnosti, že táto hrana bude vymazaná v danej iterácií.

Sila väzby sa však dá prezentovať nielen pomocou priradenia kapacít, ale i vytvorením toľkých hrán medzi dvojicou vrcholou, koľko bola tej pôvodnej hrany kapacita. Tento prístup sme už spomenuli v kapitole 1. V takomto grafe s viacnásobnými hranami má už každá hrana rovnakú silu väzieb (kapacita je rovná 1) teda jedná sa o neohodnotený graf. Naň už môžno aplikovať vysvetlený algoritmus. Práve s touto druhou možnosťou rošírenia pre ohodnotené grafy zvyknú rátať zabudované algoritmy v programoch.

Celý GN algoritmus môžeme zhrnúť nasledovne:

- výpočet hodnoty *betweenness* pre všetky hrany
- odstránenie hrany s navyššou hodnotou
- prepočítanie hodnôt pre ostávajúce hrany
- opakovanie od druhého kroku, až kým sa neodstránia všetky hrany

Proces delenia vrcholov do menších skupín sa dá výstižne zachytiť napríklad dendrogramom, viď ilustračný obrázok 10 prebratý z článku [14, FIG. 2].



Obr. 10: Dendrogram zachycujúci proces delenia

Na záver, kedže počas procesu dostávame viacero rozdelení siete (podľa toho, kde prislúchajúci dendrogram predelíme) ostáva otázkou, ktoré rozdelenie budeme považovať za optimálne. V článku [14] uvádzajú, že za optimálne budeme považovať to rozdelenie, ktoré dosahuje najvyššiu modularitu, ktorú sme definovali v kapitole 1.

3.3 Kód a výstup z R

Na spustenie GN algoritmu budeme potrebovať aj balíček `igraph`. Informácie o príkazoch z tohto balíka sme čerpali z [16].

```
library(optrees)
library(igraph)
nodes <- 1:6
arcs <- matrix(c(1,2,5, 2,3,2, 2,4,2, 4,5,4, 4,6,1 , 3,4,2),
        byrow=TRUE, ncol = 3)
mat <- ArcList2Cmat(nodes, arcs, directed=FALSE)
graf <- graph_from_adjacency_matrix(mat, mode = "undirected",
        diag=FALSE)
fc <- cluster_edge_betweenness(graf, weights=E(graf)$weight,
      directed=FALSE, edge.betweenness=TRUE, merges=TRUE,
      bridges=TRUE, modularity=TRUE, membership=TRUE)
```

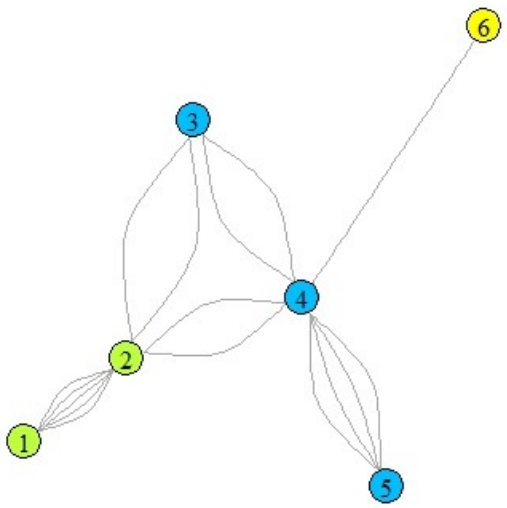
Maticu `arcs` vieme príkazom `ArcList2Cmat` jednoducho pretransformovať na maticu susednosti. Graf vytvoríme príkazom `graph_from_adjacency_matrix`, kde atribút `diag` rozhoduje, či môžu v grafe existovať aj hrany vychádzajúce a vchádzajúce do toho istého vrchola. Je dôležité podotknúť, že pri grafe vytvorenom z matice susednosti sú kapacity prezentované viacnásobnými hranami.

Na spustenie algoritmu využijeme príkaz `cluster_edge_betweenness`, do ktorého zadávame graf, vektor kapacít k jednotlivým hranám (avšak pri využití matice susednosti sú už kapacity započítané ako viacnásobnosť hrany a tento vstup už netreba zadávať). Atribút `directed` nastavíme prirodzene podľa toho, či sa jedná o orientovaný alebo neorientovaný graf. Ak nastavíme ostatné atribúty na hodnotu `TRUE`, z príkazu bude možné vytiahnúť nasledovné informácie. Atribút `edge.betweenness` vráti vektor hodnôt *betweenness*, ktoré hrany nadobudli v čase jeho odstránenia. Z atribútu `merges` je možné získať maticu zlúčenia, ktorá je opisom procesu zlučovania komunít (GN algoritmus je síce rozkladným algoritmom, avšak stačí zobrať proces od konca).Atribút `bridges` vráti zoznam počtu iterácií, pred ktorými došlo k oddeleniu subgrafu od ostatných. Atribút `modularity` vracia vektor modularít, ktoré nadobúda graf pri jednotlivých rozdeleniach do komunít, ktoré počas procesu nastali. Posledný atribút `membership` vyjadruje príslušnosť jednotlivých vrcholov k daným komunitám z optimálneho rozdelenia. Ako výstup dostávame:

```
IGRAPH clustering edge betweenness, groups: 3, mod: 0.21
+ groups:

$`1`
[1] 1 2

$`2`
[1] 3 4 5

$`3`
[1] 6
```

Na vykreslenie obrázka 11 treba využiť príkaz:

```
plot(graf, vertex.color=c("greenyellow","deepskyblue",
"yellow")[fc$membership])
```



Obr. 11: Vizuálne znázornenie siete a nájdených komunít podľa GN algoritmu

4 Greedy algoritmy na hľadanie maxima modularity

4.1 Motivácia

Pojem modularita sme už predstavili v prvej kapitole a taktiež bol využitý ako ukazovateľ optimálneho rozdelenia spomedzi vzniknutých rozdelení siete v predchádzajúcom algoritme. Pre ešte lepšie pochopenie a ilustráciu, prečo by bolo prínosné sa maximalizáciou modularity zaoberať, vzhľadom na problematiku detekovania komunít v sieťach, uvedieme nasledovnú myšlienku, uvedenú v článku [7, str. 16].

Pôvodne sme rátali modularitu (pre neohodnotené grafy) ako

$$Q = \frac{1}{2m} \sum_{i,j} \Big(A_{ij} - \frac{k_i k_j}{2m} \Big) \delta(C_i, C_j). \tag{24}$$

Keďže do súčtu sa reálne dostanú len výrazy s indexami i a j takými, že príslušné vrcholy patria do rovnakej komunity, je možné tento výpočet prepísať na

$$Q = \sum_{c=1}^{n_c} \left[\frac{l_c}{m} - \Big(\frac{d_c}{2m} \Big)^2 \right], \tag{25}$$

kde n_c sa rovná počtu komunít rozdelenia, l_c sa rovná sume počtu hrán, ktoré spájajú vrcholy komunity c a d_c predstavuje súčet stupňov vrcholov danej komunity.

Ďalej môžeme maximálnu hodnotu modularity pre danú sieť vyjadriť ako

$$Q_{max} = \max_P \left\{ \sum_{c=1}^{n_c} \left[\frac{l_c}{m} - \Big(\frac{d_c}{2m} \Big)^2 \right] \right\}, \tag{26}$$

kde pod $\max_P$ budeme rozumieť maximum cez všetky možné rozdelenia P daného grafu. Výraz $\frac{d_c^2}{4m}$ vyjadruje očakávanú hodnotu počtu hrán v danej komunite v nulovom modely - označíme $Ex(l_c)$. Ďalej môžeme upraviť na

$$\begin{aligned} Q_{max} &= \frac{1}{m} \max_P \left\{ \sum_{c=1}^{n_c} \Big[l_c - Ex(l_c) \Big] \right\}, \\ &= -\frac{1}{m} \min_P \left\{ -\sum_{c=1}^{n_c} \Big[l_c - Ex(l_c) \Big] \right\}. \end{aligned} \tag{27}$$

Pričítaním a odčítaním hodnoty m dostávame

$$Q_{max} = -\frac{1}{m} \min_{P} \left\{ \left(m - \sum_{c=1}^{n_c} l_c \right) - \left(m - \sum_{c=1}^{n_c} Ex(l_c) \right) \right\}, \tag{28}$$

kde výraz $(m - \sum_{c=1}^{n_c} l_c)$ môžeme interpretovať ako veľkosť rezu, ktorý delí sieť do daných klastrov. Všimnime si totiž, že od celkového počtu hrán m odčítavame hrany, ktoré spájajú vrcholy len z rovnakých klastrov a teda vyjadruje počet medzikomunitných hrán. Zvyšný výraz predstavuje, podobne, očakávanú veľkosť rezu pre náhodný graf. Z toho nám vyplýva, že pri maximalizovaní modularity hľadáme také rozdelenie siete P, ktoré minimalizuje rozdiel medzi veľkosťou rezu v našom grafe a veľkosťou rezu v náhodnom grafe, od ktorého sa neočakáva veľká miera zhlukovania. Teda zjednodušene povedané, toto rozdelenie by malo mať čo možno najmenšiu veľkosť rezov v pôvodnom grafe (podobne ako vo FF algoritme) a čo možno najväššiu v náhodnom. Takéto rozdelenie sa potom javí celkom ideálne, pretože v skutočnosti nám stačí "pretrhnúť"relatívne malý počet hrán medzi komunitami, ako by sme od takej siete s rovnakými stupňami vrcholov očakávali.

V skutočnosti úloha maximalizácie modularity je veľmi náročným problémom. Pre graf s n vrcholmi pri rozdelení do g klastrov vieme vypočítať počet možných rozdelení tzv. *stirlingovým číslom druhého druhu* $S(n, g)$ (viac sa o ňom možno dočítať napríklad v [17]). Celkový počet možných rozdelení grafu sa rovná $\sum_{g=1}^{n} S(n, g)$. Tento súčet nie je bližšie špecifikovaný, avšak už pre prvé dva členy $S(n, 1) + S(n, 2)$, rovnajúce sa 2^{n-1} pozorujeme minimálne exponenciálny nárast. Z tohto dôvodu maximálizácia cez všetky možné rozdelenia pre grafy vysokým počtom vrcholov bude časovo náročná.

Existuje viacero spôsobov, ako aproximovať maximálnu modularitu, ktoré sú spomenuté napr. v [7, str. 27-34]. My sa budeme zaoberať Newmanovým algoritmom z článku [13] a jeho neskoršiou modifikáciou z článku [2].

4.2 Vysvetlenie algoritmov

4.2.1 Základná myšlienka

Newman ponúkol v článku [13] spôsob aproximácie optima modularity, využijúc tzv. *greedy* techniku. Algoritmus v každej iterácií na základe definovanej podmienky rozhodne, akým spôsobom upraviť, resp. vytvoriť ďalšie rozdelenie siete s cieľom čo

najväčšieho priblíženia k maximálnej hodnote modularity. Výsledné riešenie je teda len aproximáciou a môže existovať rozdelenie s ešte väčšou modularitou, ktoré algoritmus nenájde, vzhľadom k tomu, že algoritmus nepreskúma všetky možné rozdelenia. Avšak, ako vo svojom článku uvádza, algoritmus bol v rôznych aplikáciach pomerne úspešný a jednoznačne ním zjednodušil výpočet oproti počítaniu modularity pre všetky možné rozdelenia, ktoré by sa pre grafy s počtom vrcholov nad $30 - 40$ stávalo nemožným v konečnom čase. Vo svojom neskoršom článku [2] spolu s Clausetom a Moorom, ponechajúc základnú myšlienku tohto algoritmu, modifikovali algoritmus zefektívniac výpočet zbavením zbytočných operácií (modifikovaný algoritmus budeme ďalej nazývať CNM).

Oba algoritmy radíme k triede *aglomeratívnych* algoritmov - t.j. na začiatku každý vrchol predstavuje jednu komunitu a na základe definovanej podmienky sa iteratívne zlučujú do väčších komunít.

Základnou myšlienkou týchto algoritmov bude v každej iterácií vypočítať zmenu modularity $\triangle Q_{ij}$ po zlúčení komunity i a j pre všetky páry komunít, medzi ktorými existuje aspoň jedna väzba. Po spojení takého páru komunít, ktoré nemajú ani jednu spoločnú hranu, modularita nikdy nevzrastie[2] , preto kvôli zjednodušeniu výpočtov a ušetreniu času takúto hodnotu $\triangle Q_{ij}$ nebudeme ani počítať. Následne zlúčíme klastre s najväčším prírastkom modularity, resp. s najmenším poklesom[3] a proces opakujeme, až kým budú všetky vrcholy patriť jednej komunite. Všimnime si, že pri grafe s n vrcholmi je potrebných $n-1$ iterácií, aby sme z n komunít dostali postupným zlučovaním nakoniec jednu. Proces zlučovania môže byť opäť ilustrovaní napríklad dendrogramom. Na záver, podobne ako pri GN algoritme, za optimálne rozdelenie prirodzene vyberieme rozdelenie s najväčšou dosiahnutou modularitou.

4.2.2 Pôvodný algoritmus

Pre zrozumiteľnosť a zjednodušenie popisu zavedieme nové ústredné premenné, ako sú uvádzané v článku [2]. Vytvoríme maticu e, ktorej zložka

[2]Platnosť spomínaného pozorovania bude objasnená v podsekcií 4.2.2 po zavedení vhodného značenia.

[3]V podsekcií 4.2.3 ľahko ukážeme, že akonáhle raz modularita v nejakej iterácií klesne, bude klesať aj v ďalších a teda najvyššia modularita v týchto algoritmoch je dosiahnutá pred spomínanou iteráciou.

$$e_{ij} = \frac{1}{2m} \sum_{u,v} A_{uv} \delta(C_u, i) \delta(C_v, j) \tag{29}$$

sa dá interpretovať ako pomer počtu hrán spájajúcich komunitu i s komunitou j ku počtu koncov všetkých hrán v grafe (teda $2m$). Tieto premenné vytvárajú symetrickú štvorcovú maticu e rozmerov n_c, kde n_c sa rovná počtu komunít. Ďalšiou dôležitou premennou bude vektor a a jeho i-ty člen

$$a_i = \frac{1}{2m} \sum_{u} k_u \delta(C_u, i) \tag{30}$$

bude predstavovať pomer súčtu koncov hrán vrcholov prislúchajúcich komunite i ku počtu všetkých koncov hrán.

Uvedomme si, že platí rovnosť

$$\delta(C_u, C_v) = \sum_{i} \delta(C_u, i) \delta(C_v, i), \tag{31}$$

pretože v prípade, že pravá strana rovnice sa rovná 1 (teda $C_u = C_v$), súčet naľavo obsahuje samé nuly a práve jednu jednotku pre člen, kde $i = C_u = C_v$. V prípade pravej strany rovnej nule v súčte naľavo dostávame samé nuly. Využijúc túto rovnosť, môžeme upraviť vzorec na výpočet modularity:

$$\begin{aligned} Q &= \frac{1}{2m} \sum_{u,v} \left(A_{uv} - \frac{k_u k_v}{2m} \right) \sum_{i} \delta(C_u, i) \delta(C_v, i) \\ &= \sum_{i} \left(\frac{1}{2m} \sum_{u,w} A_{uw} \delta(C_u, i) \delta(C_v, i) - \frac{1}{2m} \sum_{u} k_u \delta(C_u, i) \frac{1}{2m} \sum_{v} k_v \delta(C_v, i) \right) \\ &= \sum_{i} (e_{ii} - a_i^2). \end{aligned} \tag{32}$$

Všimnime si, čo sa deje s maticou e pri zlúčovaní komunity i s komunitou j. Novú komunitu nazvyme j^*, potom si je ľahko podľa interpretácie rovnice (29) uvedomiť, že

$$e_{j^*j^*} = e_{ii} + e_{jj} + e_{ij} + e_{ji} = e_{jj} + e_{ii} + 2e_{ij}, \tag{33}$$

$$e_{j^*k} = e_{ik} + e_{jk}. \tag{34}$$

Z rovníc (33) a (34) vyplýva, že v každej iterácií matica e sčítava príslušné dva riadky a stĺpce prislúchajúce i-tej a j-tej komunite.

Taktiež z interpretácie rovnice (30) môžeme vyvodiť, že vektor a sa v každej iterácií upraví nasledovným spôsobom:

$$a_{j^*} = a_i + a_j. \tag{35}$$

Matica e a vektor a každou iteráciou strácajú jednu dimenziu, vzhľadom k tomu, že klesá počet komunít o jeden. Zmenu modularity po zlúčení i-tej a j-tej komunity sa dá potom vypočítať ako

$$\triangle Q_{ij} = e_{ij} + e_{ji} - 2a_i a_j = 2(e_{ij} - a_i a_j), \tag{36}$$

vychádzajúc pritom z výpočtu modularity z rovnice (32) a z poznatkov, ako sa menia matica e a vektor a. Z rovnice (36) je dobre vidieť, že ak dve komunity neboli prepojené ani jednou hranou (teda $e_{ij} = e_{ji} = 0$), tak zmena modularity po zlúčení takýchto komunít bude vždy záporná.

Algoritmus môžeme zhrnúť nasledovne:

- vypočítaj maticu e, vektor a a počiatočnú modularitu Q (všimnime si, že táto hodnota bude vždy záporná)
- vypočítaj $\triangle Q_{ij}$ pre všetky páry komunít okrem neprepojených párov
- nájdi najväčšiu hodnotu prírastku, zlúč dané dve komunity, uprav maticu e a vektor a podľa popisu vyššie a doterajšiu modularitu Q navýš o hodnotu $\triangle Q_{ij}$
- opakuj od druhého kroku, kým sa počet komunít nezredukuje na jedna

4.2.3 CNM algoritmus

Ako sme už spomínali v podsekcií 4.2.1 , v článku [2] nám Clauset, Newman a Moore ponúkajú modifikáciu predošlého algoritmu s cieľom zbaviť sa zbytočných operácií. Ich motiváciou bolo skrátiť čas potrebný na zbehnutie algoritmu, hlavne v prípade grafov s vysokým počtom vrcholov.

Predstavujú nám inú štruktúru dát, ktorú budú udržiavať v pamäti. Namiesto matice e zaviedli maticu $\triangle Q$, ktorej prvky $\triangle Q_{ij}$ rovno vyjadrujú zmenu modularity v prípade zlúčenia komunity i s komunitou j. Ako už bolo spomenuté, komunity, ktoré nie sú prepojené ani jednou hranou, nemôžu modularitu po ich zlúčení navýšiť. Takýmto dvojiciam komunít na začiatku priradíme v matici $\triangle Q$ nulovú hodnotu. V skutočnosti však po zlúčení spomenutých komunít dôjde k poklesu modularity - pozri rovnicu (36), za predpokladu nenulovosti oboch výrazov a_i a a_j. Na druhej strane, ako sme už v pôvodnom algoritme načrtli, pre nás budú kľúčové dvojice s kladným prírastkom. Na konci tejto podsekcie uvidíme, že akonáhle sa nebudú v matici $\triangle Q$ nachádzať kladné hodnoty, modularita v ďalších rozdeleniach už nemôže narásť. Z tohto dôvodu nemusíme neprepojeným párom komunít venovať veľkú pozornosť, podobne ako v pôvodnom algoritme. Priradíme im rovno nulovú hodnotu. Ušetrí to veľa zbytočných operácií. Odvodiac z poznatku, že na začiatku každý vrchol začína ako samostatná komunita a z rovníc (29), (30) (36), počiatočná matica $\triangle Q$ bude vyzerať nasledovne:

$$\triangle Q_{ij} = \begin{cases} 1/m - k_i k_j / 2m^2 & \text{ak } i \text{ a } j \text{ sú prepojené,} \\ 0 & \text{inak,} \end{cases} \tag{37}$$

V každej iterácií bude opäť snahou zlúčiť komunity s najväčšiou hodnotou $\triangle Q_{ij}$. Na hľadanie tejto hodnoty využijeme tzv. *maximovú haldu H* (angl. *max heap*), binárny strom, kotrý bude obsahovať najväčšie hodnoty každého riadku, resp. stĺpca matice $\triangle Q$ s príslušnými indexami i a j(viac o maximovej halde napríklad v [1]).

Podobne ako v pôvodnom algoritme, aj teraz budeme pracovať s vektorom a, ktorý na začiatku nadobúda tvar

$$a_i = \frac{k_i}{2m}. \tag{38}$$

Opäť popisuje rovnaký jav a bude sa aj rovnako meniť každou iteráciou, ako bolo popísané v pôvodnom algoritme.

CNM algoritmus môžeme zhrnúť nasledovne:

- vypočítaj počiatočnú modularitu Q, maticu $\triangle Q$, vektor a a vlož do maximovej haldy H najväčšie hodnoty z každého riadku matice $\triangle Q$

- najväčšiu hodnotu $\triangle Q_{ij}$, nájdenú maximovou haldou H, pripočítaj k doterajšej modularite Q, zlúč príslušnú i-tu a j-tu komunitu a prepočítaj maticu $\triangle Q$ podľa popisu nižšie), vektor a a maximovú haldu H.
- opakuj druhý krok, až kým ostane jedna komunita

Poslednou otázkou ostáva, akým spôsobom upravovať maticu $\triangle Q$ v jednotlivých iteráciach. Ilustrujme to na nasledovnej situácií.

V predošlej iterácií sme zlúčili i-tu s j-tou komunitou. Novovzniknutú komunitu si nazvyme j^*. Ukážeme, ako by sa zmenila modularita v prípade, že zlúčime j^* s k-tou komunitou. Podľa rovníc (34), (35) a (36) vieme, že

$$\begin{aligned} \triangle Q_{j^*k} &= 2(e_{j^*k} - a_{j^*}a_k) = 2(e_{ik} + e_{jk} - (a_i + a_j)a_k) \\ &= 2(e_{ik} - a_i a_k) + 2(e_{jk} - a_j a_k) = \triangle Q_{ik} + \triangle Q_{jk}. \end{aligned} \tag{39}$$

Z toho vyplýva, že zmena modularity po zlúčení už skoršie zlúčenej i-tej a j-tej komunity s komunitou k sa rovná zmene modularity po zlúčení i-tej a k-tej komunity plus zmena modularity po zlúčení j-tej a k-tej.

Pripomeňme si, že na začiatku sme neprepojeným párom komunít zadali nulovú hodnotu, ktorá však reálne neodpovedá zmene modularity v prípade ich zlúčenia. Skutočnú zmenu modularity po spojení počítame rovnicou (36), a preto treba postupovať nasledovne:

$$\triangle Q_{j^*k} = \triangle Q_{ik} - 2a_j a_k, \tag{40}$$

ak k bolo prepojené s i, ale nebolo prepojené s j. Podobne

$$\triangle Q_{j^*k} = \triangle Q_{jk} - 2a_i a_k, \tag{41}$$

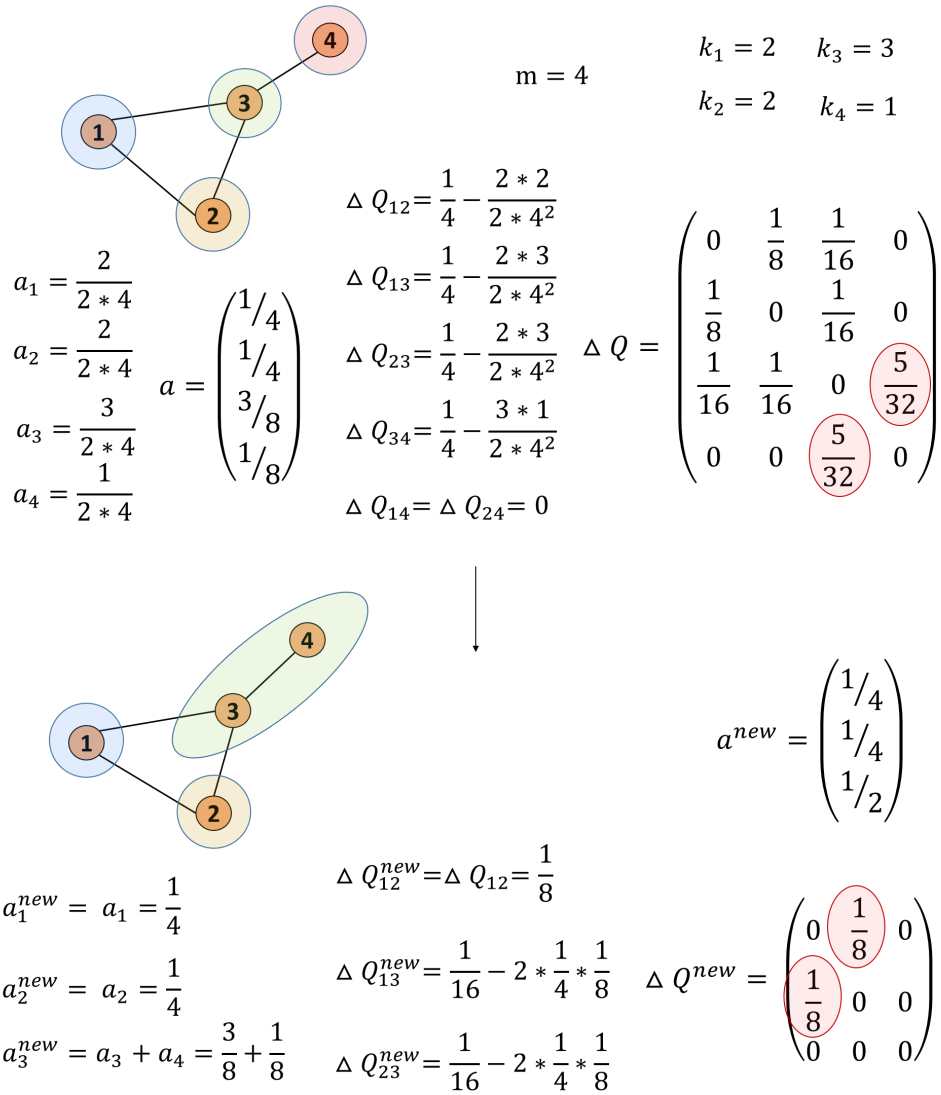
ak k bolo prepojené s j, ale nebolo prepojené s i. V prípade, že k nie je prepojené ani s i ani s j nastavíme

$$\triangle Q_{j^*k} = 0. \tag{42}$$

Práve z týchto úprav, ako sa mení matica $\triangle Q$ môžeme pozorovať, že akonáhle už v matici nemáme ani jednu kladnú hodnotu, v ďalších iteráciach je nemožné dostať

kladný prírastok modularity. Z toho vyplýva, že rozdelenie nájdené v iterácií po zlúčení posledných dvoch komunít s kladným prírastkom dosahuje v rámci rozdelení, ktoré sme počas algoritmu dostali, najvyššiu hodnotu modularity a budeme ho pokladať za optimálne. Teoreticky by sa dalo proces v tomto okamihu zastaviť, avšak v článku uvádzajú, že sa dokončí až po rozdelenie s jednou komunitou. Tento proces môže byť teda následne zachytení dedrogramom.

Pre lepšiu predstavu poskytujeme praktickú ukážku jednej iterácie pre danú sieť, pričom sa budeme pri výpočtoch riadiť pravidlami uvedenými vyššie.



Obr. 12: Praktické znázornenie jednej iterácie v CNM algoritme

Na záver treba podotknúť, že podobne ako v prípade GN algoritmu, aj tieto algoritmy sú jednoduchým rozšírením aplikovateľné na ohodnotené grafy. Modularita bude počítaná pomocou rovnice (10) z kapitoly 1. V matici susednosti namiesto jednotiek budú vystupovať príslušné hodnoty kapacít hrán spájajúcich dané dvojice vrcholov. Stupeň vrchola bude rovný súčtu kapacít hrán vychádzajúcich z daného vrchola a počet všetkých hrán sa bude rovnať súčtu všetkých kapacít hrán.

4.3 Kód a výstup v R

Príkaz sa podobá príkazu pre GN algoritmus. Treba však spomenúť, že graf nesmie byť vytvorený maticou susednosti, pretože príkaz nevie spracovať viacnásobné hrany. Graf je možné vytvoriť napríklad nasledovne:

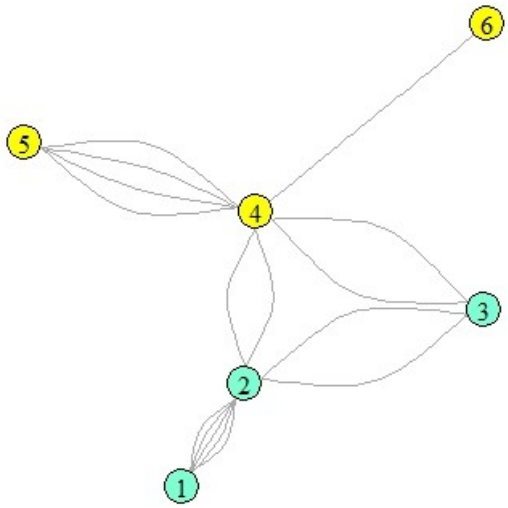
```
library(igraph)
arcs <- matrix(c(1,2,5, 2,3,2, 2,4,2, 4,5,4, 4,6,1 , 3,4,2),
        byrow=TRUE, ncol = 3)
graf <- make_undirected_graph(t(arcs[,1:2]), 6)
E(graf)$weight <- c(arcs[,3])
cluster_fast_greedy(graf, merges = TRUE, modularity = TRUE,
membership = TRUE, weights = E(graf)$weight)
```

Príkaz `cluster_fast_greedy` obsahuje atribúty vysvetlené v podsekcií 3.3.

Ako výstup dostávame:

```
IGRAPH clustering fast greedy, groups: 2, mod: 0.24
+ groups:
$`1`
[1] 1 2 3

$`2`
[1] 4 5 6
```

Získané komunity demonštrujeme na obrázku 13, ktorý možno dostať analogickým spôsobom pomocou príkazu uvedenom v podsekcií 3.3 .



Obr. 13: Vizuálne znázornenie siete a nájdených komunít podľa CNM algoritmu

5 Iné prístupy

Z predošlých kapitol vidíme, že sa na danú problematiku dá nazerať z viacerých uhlov pohľadu. V článku [7] je predstavených mnoho ďalších prístupov. Niektoré si teraz uvedieme.

Pri FF algoritme bolo ústredným javom minimalizovanie kapacity rezu. Týmto javom sa zaberá aj *spectrálna bisekcia*, založena na využití vhodných vlastností Laplaceovej matice[4]. Kapacita rezu sa teraz dá vypočítať ako

$$c(S, V\backslash S) = \frac{1}{4} s^T L s, \tag{43}$$

kde elementy vektora s sú rovné 1, ak prislúchajúci vrchol patrí prvej množine alebo -1, ak patrí do opačnej. S využitím vlastných vektorov v_i Laplaceovej matice (normalizovaných na dĺžku 2) pri prepísaní vektora s na $\sum_i a_i v_i$ vieme kapacitu rezu určiť výrazom

$$c(S, V\backslash S) = \sum_i a_i^2 \lambda_i, \tag{44}$$

kde λ_i je príslušnou vlastnou hodnotou Laplaceovej matice. Spektrálna bisekcia sa teda zaoberá minimalizovaním vzniknutého výrazu z rovnice (44). Výsledné dva klastre však nemusia mať požadované proporčné vlastnosti, preto vznikli viaceré stratégie, snažiace sa tomuto javu čo najviac zabrániť - viac v [7, str. 18].

Jedným z prvých algoritmov zaoberajúcim sa minimalizovaním kapacity rezov je aj *Kernighan-Linov algoritmus* [7, str. 17]. Pôvodne bol navrhnutý za účelom optimálneho rozmiestnenia elektronických obvodov na dosky. Snahou bolo umiestniť komponenty na dosky tak, aby bolo čo najmenej prepojení medzi doskami. Za vstup je treba zadať prvotné rozdelenie siete do dvoch klastrov o rovnakej veľkosti, z ktorého algoritmus bude vychádzať. Iteratívnym spôsobom dochádza k výmenám podmnožín vrcholov medzi klastrami za účelom minimalizovania kapacity rezu. Tento algoritmus je rozšíriteľný aj na iný požadovaný počet klastrov a taktiež na ich rôzne požadované veľkosti. Nevýhodou je, že výsledné rozdelenie býva silno závislé od prvotného vstupu,

[4]Laplaceova matica L je definovaná predpisom $D - A$, kde D je diagonálna matica, ktorej hodnoty odpovedajú jednotlivých stupňom vrcholov grafu a matica A je maticou susednosti, resp. maticou W s príslušnými kapacitami.

preto sa tento algoritmus často používa ako vylepšenie rozdelenia nájdeného inou metódou. Viac informácií v [10].

Pri hľadaní komunít (teda zhlukov v grafe reprezentujúcom nejakú sieť) ide teda o snahu nájsť objekty, ktoré sú si v niečom podobné, blízke. Predstavme si, že máme dáta - dátové objekty s určitými vlasnosťami, ktoré je potrebné roztriediť do klastrov tak, aby sa v klastroch nachádzali dáta podobných vlastností. Jedným z možných spôsobov je previesť tieto dátové objekty do metrického priestoru, kde jednotlivé súradnice odpovedajú zadaným vlastnostiam tohto objektu. Body reprezentujúce podobné dáta sa budú v tomto priestore nachádzať blízko vedľa seba - vzdialenosť medzi dvojicou bodov bude teda predstavovať mieru rozdielnosti príslušných objektov. Cieľom triedy metód *partitional clustering* je rozdeliť objekty do k klastrov, minimalizovaním nákladovej funkcie založenej na vzdialenostiach medzi bodmi a *centroidmi*, vhodne definovanými v priestore. Je viacero spôsobov definovania tejto nákladovej fukncie - pozri [7, str. 20]. Jedna z najpopulárnejších metód z tejto triedy zvaná *k-means clustering* od MacQueena využíva predpis

$$\sum_{i=1}^{k} \sum_{x_j \in S_i} ||x_j - c_i||^2, \tag{45}$$

kde S_i predstavuje množinu bodov tvoriacu i-ty klaster a c_i reprezentuje jej príslušný centroid. Tento problém rieši *Lloydov algoritmus*, kde sa na začiatku určia prvotné centroidy. V prvej iterácií sa body zatriedia k najbližšiemu centroidu. Z takto vzniknutých klastrov odhaneme ich stredy a tie potom určíme za nové centroidy do ďalšej iterácie. Takto pokračujeme, pokým sa roztriedenie do klastrov neustáli. Avšak opäť je výsledok silno ovplyvnený prvotným určením centroidov. Je teda vhodné previesť viacero spustení algoritmu s rôznym určením prvotných centroidov.

Ďalšiou triedou metód je tzv. *spektrálne zhlukovanie*. Na jej využitie potrebujeme definovať funkciu podobnosti S, ktorá priradí nejakú hodnotu každej dvojici objektov. Od tejto funkcie očakávame nezápornosť a symetrickosť. Príkladom môže byť veľkosť prieniku spoločných susedov[5]týchto dvoch objektov (vrcholov v grafe) predelený veľkosťou ich zjednotenia [7, str. 12]. Vypočítame maticu S, ktorej element S_{ij} sa bude rovnať hodnote priradenej funkciou podobnosti pre príslušné objekty i a j. Touto

[5]Pojem sused vrchola označuje taký vrchol v grafe, ktorý má s daným vrcholom spoločnú hranu.

maticou môže byť napríklad aj samotná matica susednosti. Či už ide o dátové objekty, resp. body v priestore alebo o vrcholy grafu, v týchto metódach ich transformujeme na body v priestore, ktorých súradnice budú elementami vlastných vektorov matice S (poprípade jej modifikácie). Takto získané body už ďalej možno analyzovať metódami naznačenými v predošlom odseku, napríklad $k - means clustering$. Na čo je to dobré? Prečo nevyužiť tieto metódy rovno? Touto tranformáciou sa totiž stávajú vlastnosti zhlukov ľahšie pozorovateľné. Viac o spektrálnom zhlukovaní v [7, str. 20-23].

Vyššie spomínané metódy a algoritmy majú veľkú nevýhodu v tom, že ako vstup musia mať zadaný počet klastrov, do koľkých majú objekty zatriediť. Ako vieme, algoritmy z kapitol 3 a 4 patria do triedy rozkladných, resp. aglomeratívnych metód, ktoré sú vhodné pri hierarchickom zhlukovaní siete a tento vstup nepotrebujú. Medzi aglomeratívne patrí aj metóda *single linkage clustering*. V každej iterácií spája dva klastre s najvyššou podobnosťou, pričom podobnosť i-teho s j-tym klastrom má definovanú ako minimálnu podobnosť z podobností všetkých dvojíc objektov, z ktorých jeden je z i-teho a druhý z j-teho klastra. Metóda *complete linkage clustering* má definovanú podobnosť dvoch klastrov, práve naopak, maximom z týchto podobností. Metóda *average linkage clustering* ju definuje strednou cestou - ako ich priemer.

Veľkou triedou metód sú metódy a algoritmy maximalizujúce modularitu. Na tie sme už odkazovali v kapitole 4. Aj pre tieto účely sa môže využiť napríklad spektrálna optimalizácia, pomocou vlastných vektorov a vlastných hodnôt takej matice B, že $B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$ ale i veľa ďalších metód [7, str. 27-34].

6 Aplikácia algoritmov na reálne sociálne siete

V tejto kapitole uvedieme dva príklady sociálnych sietí, vzniknutých z vlastných dát. Vysvetlíme, ako sme ohodnocovali vzťahy medzi nimi. Na siete následne aplikujeme algortimy, ktorým sme sa venovali v kapitolách 2 - 4 a nájdené rozdelenia porovnáme.

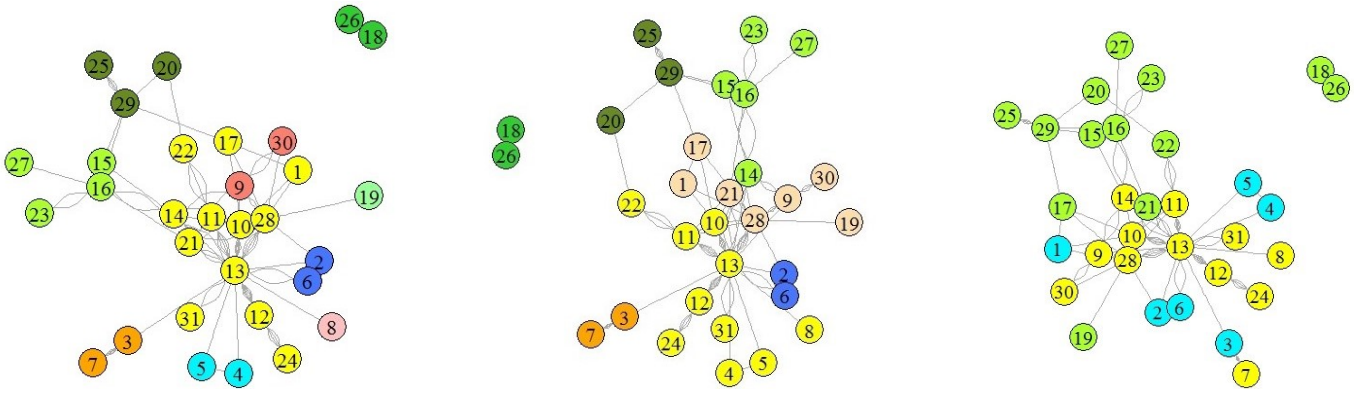
6.1 Spoluatorstvá publikácií

Sociálna sieť pozostáva z pracovníkov Katedry aplikovanej matematiky a štatistiky. Zoznam pracovníkov sme čerpali zo stránky [6]. Kapacity ich väzieb sú prezentované počtom spoločnej publikačnej činnosti, takže vzťahy sú symetrické, graf je ohodnotený, neorientovaný. Brali sme do úvahy publikácie publikované do konca roku 2016. Informácie o publikáciach sme čerpali zo stránky [5]. V sieti sú zahrnutí len tí pracovníci, ktorý mali aspoň jednu spoločnú publikáciu s niekým z katedry.

Katedra sa delí na 3 oddelenia - Oddelenie aplikovanej matematiky, Oddelenie ekonomických a finančných modelov a Oddelenie štatistiky a poistnej matematiky. Doktorandov sme priradili k tomu oddeleniu, z ktorého majú školiteľa. Pre zaujímavosť porovnáme výsledné komunity nájdené algoritmami so skutočným zaradením pracovníkov do oddelení. Uvidíme teda, či majú pracovníci silnú tendenciu publikovať najma s členmi z rovnakého oddelenia.

6.1.1 Analýza rozdelení podľa GN a CNM algoritmov

Na obrázku 14 sú znázornené výsledné komunity podľa jednotlivých algoritmov.



Obr. 14: Zľava: rozdelenie podľa GN, rozdelenie podľa CNM, rozdelenie podľa oddelení

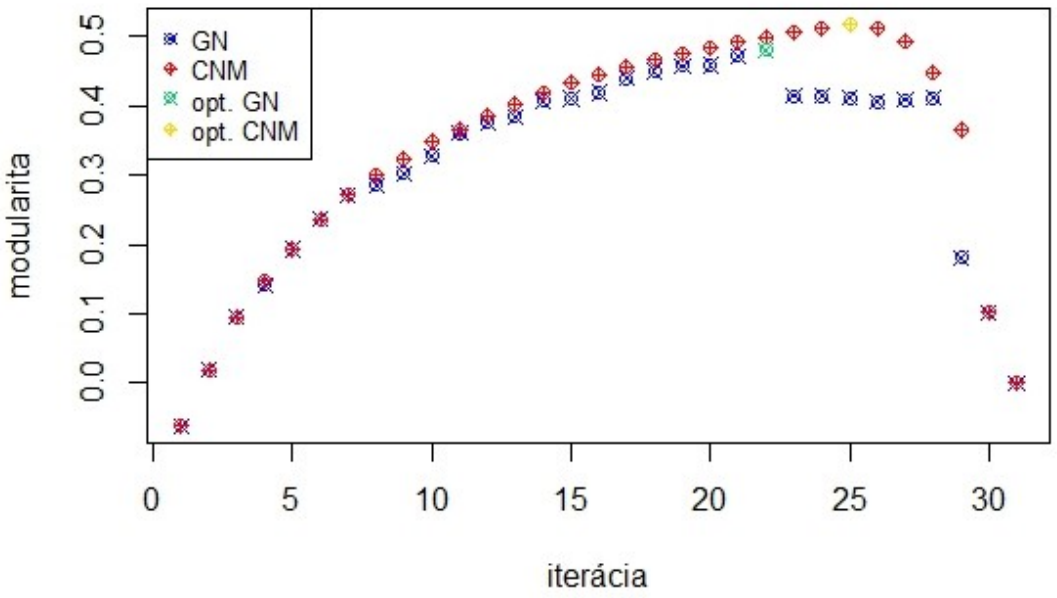
Priradenie mien k číslam a príslušné dendrogramy zachytávajúce proces GN a CNM algoritmov možno nájsť v prílohách A a B.

Ako porovnať, ktoré rozdelenie vykazuje silnejšie črty vlastností, ktoré prislúchajú komunitám? V tabuľke 1 sa nachádzajú hodnôty jednotlivých funkcií kvalít, ktoré sme predstavili v kapitole 1.

algoritmus	modularita	coverage	conductance
GN	0,4802	0,7946	0.5021
CNM	0,5162	0,7411	0.6693

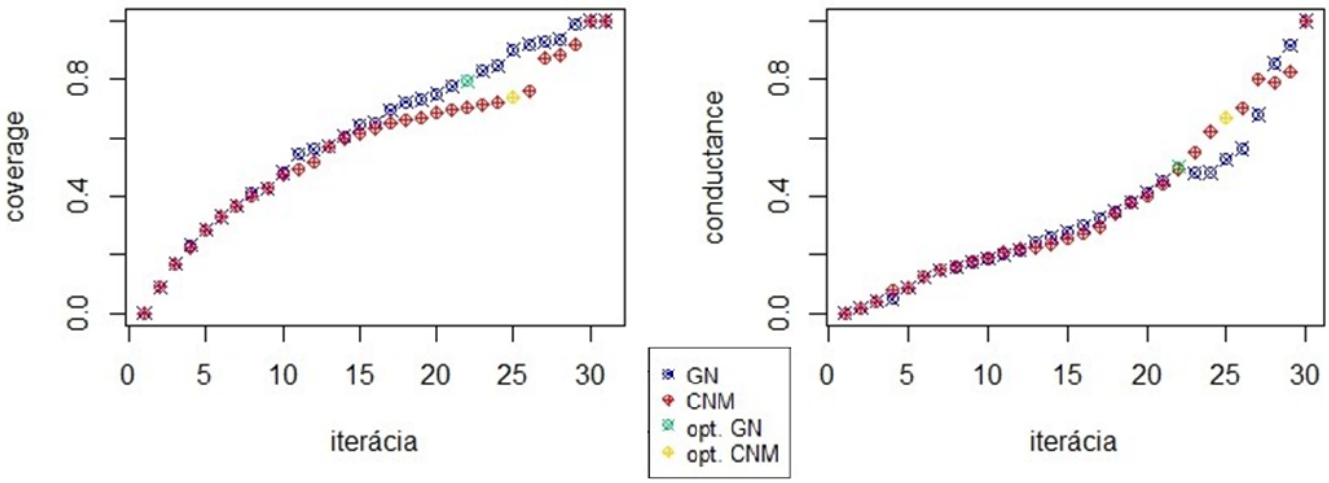
Tabuľka 1: Tabuľka hodnôt rôznych funkcií kvality

Nie je takým prekvapením, že rozdelenie podľa CNM dosiahlo vyššiu hodnotu modularity. Práve modularita bola totiž ústrednou vlastnosťou rozdelenia, ktorú sa CNM algoritmus vo všeobecnosti snaží maximalizovať. Na grafe 15 môžeme sledovať vývoj modularity počas procesu zlučovania. Vidíme, že GN algoritmu sa ani raz počas procesu nepodarilo "predbehnúť"CNM.



Obr. 15: Priebeh dosahovania rôznej úrovne modularity počas procesu zlučovania

Taktiež pre zaujímavosť uvádzame grafy zachytávajúce vývoj hodnôt *coverage* a *conductance* počas procesu zlučovania u oboch algoritmov - viď. obrázok 16.



Obr. 16: Priebeh dosahovania rôznej úrovne *coverage* (naľavo) a *conductance* (napravo) počas procesu zlučovania

Nakoniec sme preskúmali, nakoľko môžeme považovať vzniknuté rozdelenia siete za podobné medzi sebou a taktiež ich porovnať s reálnym rozdelením do oddelení. Voľným okom môžeme z obrázku 14 vypozorovať, že niektoré klastre z rozdelenia GN a CNM sa líšia len v malom počte vrcholov, resp. niektoré sú dokonca zhodné. Celkovú podobnosť dvoch rozdelení vyjadríme pomocou Randovho indexu z kapitoly 1. Dostávame nasledovné hodnoty:

	GN a CNM	GN a oddelenia	CNM a oddelenia
Randov index	0.8280	0.6753	0.6495

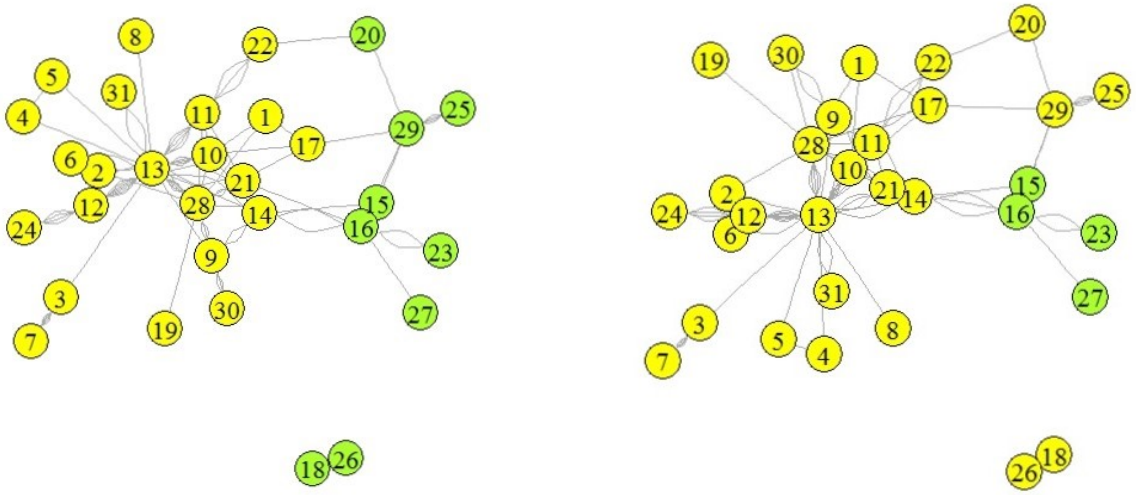
Tabuľka 2: Podobnosť rozdelení

Vidíme, že rozdelenia vzniknuté GN a CNM algoritmami sú výrazne podobnejšie s porovnaním so skutočným zadelením do oddelení. Pridáme fakt, že hodnota modularity pre rozdelenie siete podľa oddelení dosahuje približne 0.3423, čo je výrazne menej ako pre nájdené rozdelenia podľa algoritmov, avšak stále je to vcelku slušná hodnota. Z toho možno usúdiť, že pracovníci z jednotlivých oddelení striktne nepublikujú len spolu

s tým istým oddelením a prichádza aj k spolupráci s ostatnými oddeleniami. Avšak istá väčšia miera hustoty väzieb medzi oddeleniami tam je. Taktiež z tabuľky možno vyčítať, že na rozdelenie do oddelení sa podľa Randovho indexu ponáša o čosi viac rozdelenie nájdené GN algoritmom ako rozdelenie nájdené CNM.

6.1.2 Aplikácia FF algoritmu

Z hľadiska charakteru siete je FF algoritmus trochu ťažšie aplikovateľný. Treba podľa nejakej úvahy určiť dve osoby, ktoré označíme za zdroj a ústie siete. V Zacharyho karate klube bola voľba jasná. V našom prípade prvou myšlienkou bolo postaviť proti sebe vždy vedúcich z dvoch katedier. Výsledky však neboli príliš uspokojivé, pretože niektorí vedúci nemali dostatočnú štruktúru spoluautorstiev, vyžadovanú pre správne fungovanie FF algoritmu. Ako výsledok sme zväčša dostali oddelenie jedného vedúceho (alebo sa k nemu pridal maximálne jeden ďalší pracovník) od ostatných. Z tohto dôvodu sme sa rozhodli zvoliť za ústredné osoby také, ktoré sú samozrejme z rôznych komunít, ale majú silnú štruktúru väzieb (majú vysoký stupeň vrchola). Pozreli sme sa teda, kto má najviac väzieb z jednotlivých oddelení. Taktiež z obrázka 14 je pozorovať, že Oddelenie aplikovanej matematiky je už na pohľad málo kompaktná a prepojená. Rozhodli sme sa teda zvoliť osoby s najvyšším počtom väzieb z Oddelenia ekonomických a finančných modelov (13 - Ševčovič) a Oddelenia štatistiky a poistnej matematiky (16 - Harman). Výsledok bol nasledovný:



Obr. 17: Rozdelenie siete pre zdroj 13 a ústie 16 (naľavo) a zdroj 16 a ústie 13 (napravo)

Pozorujeme, že rez s minimálnou kapacitou nie je jedinečný, pretože pri obrátení zdroja s ústim sme dostali iný. Tento fakt znamená, že skupina vrcholov 20, 25 a 29 a viditeľne aj skupina vrcholov 18 a 26 sú prepojené s oboma zvyšnými časťami klastrov rovnakým počtom väzieb, reps. neprepojené vôbec. Veľkosť kapacity minimálneho rezu vyšla rovná 6.

Všimnime si, že rez 1 správne klasifikoval až 9 z 13 pracovníkov z Oddelenia štatistiky a poistnej matematiky. Pozrime sa teda, ako dopadli rezy podľa kritérií jednotlivých funkcií kvalít a podobonosťou s oddeleniami:

	modularita	coverage	conductance	podobnosť s oddeleniami
Rez 1	0.3302	0.9464	0.8125	0.6129
Rez 2	0.1913	0.9464	0.6842	0.4194

Tabuľka 3: Porovnanie jednotlivých hodnôt získaných rezov

Rez 1 dosiahol slušnú úroveň modularity, podobnú pre rozdelenie podľa oddelení. Za zmienku stojí aj rovnaká hodnota *coverage* v oboch rezoch. Tento jav nie je náhodný. *Coverage* (podiel počtu vnútrokomunitných hrán ku počtu všetkých hrán) sa rovná v skutočnosti komplementu podielu medzikomunitných hrán ku všetkým hranám. My však vieme, že počet týchto hrán v oboch rezoch sa rovná (kapacita oboch rezov je rovnaká - najmenšia možná). Dôvodom, prečo je hodnota *coverage* taká vysoká, je výrazne vďaka spomínanému fenoménu rásť pri klesajúcom počte klastrov (viď definícia 1.3), preto sa v tomto smere nedá úplne porovnať s *coverage* rozdelení podľa GN a CNM.

6.2 Sociálna sieť zo spolužiakov v ročníku

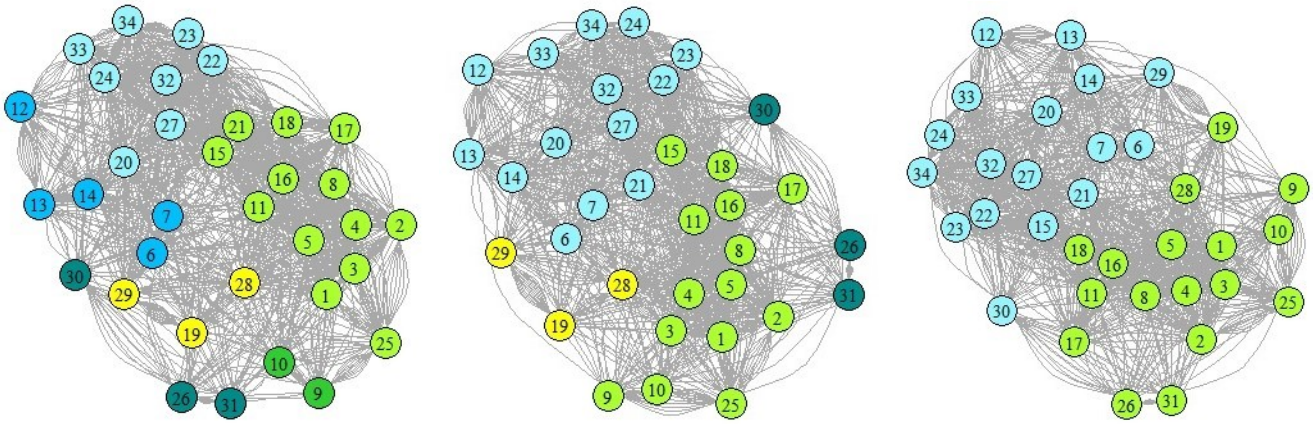
Druhá sieť, ktorou sa budeme zaoberať pozostáva zo študentov tretieho ročníka zo študijného programu Ekonomickej a finančnej matematiky (EFM). Kapacitu jednotlivých väzieb sme ohodnotili na základe odpovedí z dotazníku (viac v prílohe C). Opäť predpokladáme symetrické vzťahy - vzniknutý graf je neorientovaný, ohodnotený.

V tomto prípade budeme pre zaujímavosť porovnávať nájdené rozdelenia do komunít so skutočným zadelením študentov do krúžkov v zimnom semestri[6]tohto roku.

[6]Mohli sme sa zatriediť podľa vlastných preferencií. V letnom semestri už krúžky neboli ustálené.

6.2.1 Analýza rozdelení podľa GN a CNM algoritmov

Opäť uvádzame výsledné rozdelenia do komunít podľa oboch algoritmov a zadelenie do krúžkov v zimnom semestri - viď. obrázok 18. V prílohe D sú poskytnuté proces zachytávajúce dendrogramy príslušných algoritmov danej sieti.



Obr. 18: Zľava: rozdelenie podľa GN, rozdelenie podľa CNM, rozdelenie podľa krúžkov

Ako môžeme vidieť, vzniknuté komunity v oboch prípadoch celkom dobre oddeľujú krúžky, do ktorých sme sa zimný semester tohto roku zadelili. Všimnime si zároveň, že podobne, ako v prípade predošlej sociálnej siete, aj v tomto prípade GN algoritmus rozdelil sieť na väčší počet komunít ako CNM. Pre porovnanie opäť uvádzame tabuľku s hodnotami, dosiahnutými v jednotlivých funkciách kvality.

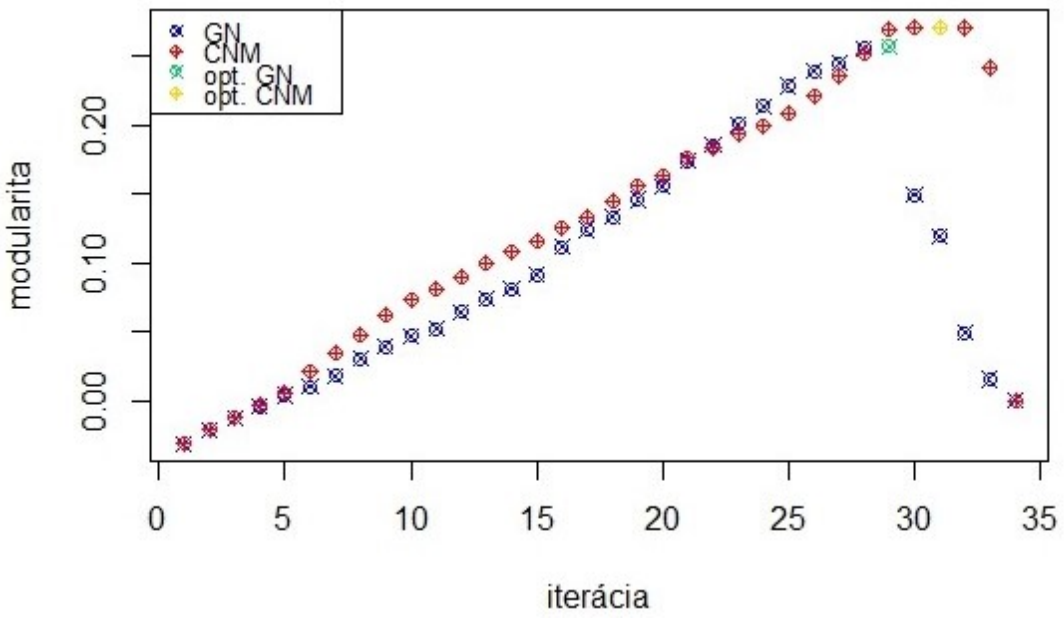
algoritmus	modularita	coverage	conductance
GN	0.2572	0.5574	0.2859
CNM	0.2716	0.6677	0.3815

Tabuľka 4: Tabuľka hodnôt rôznych funkcií kvality

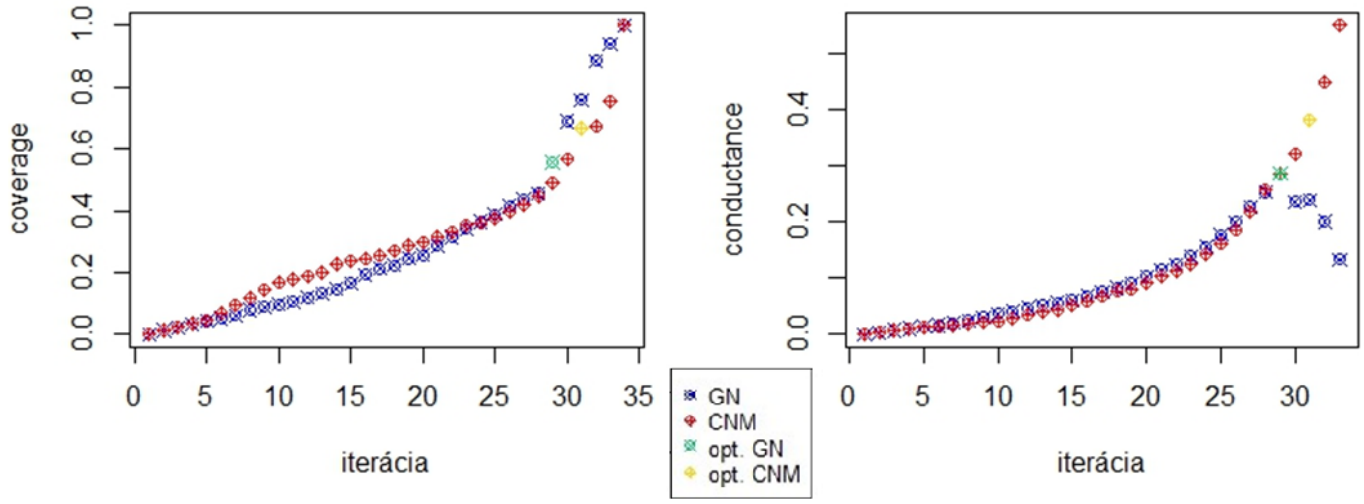
Z tabuľky môžeme vyčítať, že rozdelenie nájdené pomocou CNM algoritmu má poďla týchto troch kritérií vhodnejšie vlastnosti na detekovanie komunít ako rozdelenie podľa GN. Vo všeobecnosti však zároveň pozorujeme, že úroveň hodnôt z tejto tabuľky je v porovnaní s úrovňou hodnôt z tabuľky 1 z predošlej siete o dosť nižšia. Tento fakt poukazuje na to, že druhá skúmaná sieť nevykazuje takú vysoku mieru zhlukovania - teda vrcholy sú vo väčšej miere poprepájané navzájom aj mimo nájdených klastrov.

Tento fenomén je jednoducho pozorovateľný už rovno z obrázku 18, kde vidieť, že sieť má vysokú hustotu väzieb vrámci, ale i mimo klastrov.

Pozrime sa na priebeh hodnôt funkcií kvality, ktoré sa nadobúdali počas procesu zlučovania.



Obr. 19: Priebeh dosahovania rôznej úrovne modularity počas procesu zlučovania



Obr. 20: Priebeh dosahovania rôznej úrovne *coverage* (naľavo) a *conductance* (napravo) počas procesu zlučovania

V tomto prípade modularita v GN algoritme nachvíľu prevýšila hodnoty modula-

rity v CNM algoritme. Z celkového hľadiska však CNM dosahuje vyššiu hodnotu, tak ako sme očakávali. Zaujímavé je pozorovať ľavý graf z obrázka 20. Vidíme, že medzi nájdenými rozdeleniami v GN algoritme má rozdelenie s najvyššiu modularitou zároveň aj navyššiu hodnotu *conductance*.

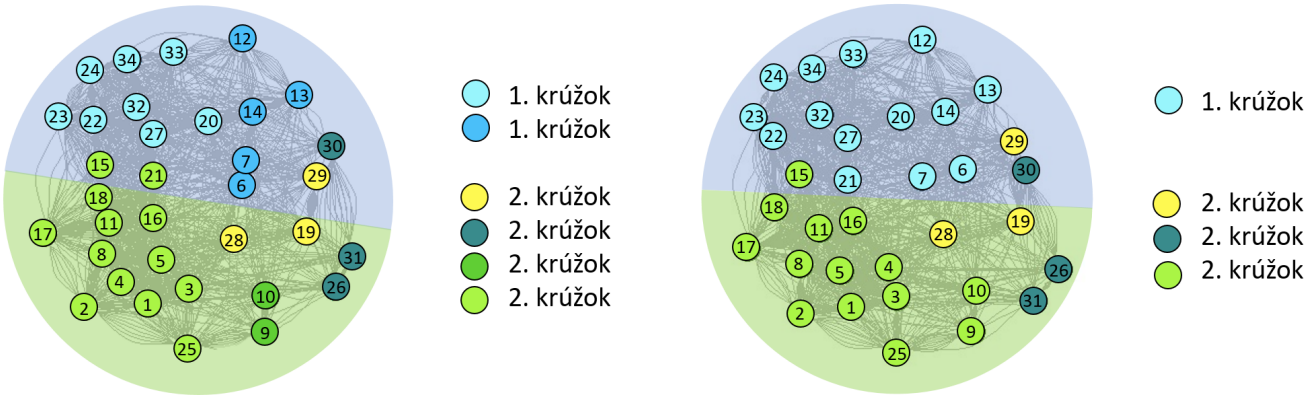
Na záver porovnáme podobnosti jednotlivých rozdelení:

	GN a CNM	GN a krúžky	CNM a krúžky
Randov index	0.8413547	0.6417112	0.7896613

Tabuľka 5: Podobnosť rozdelení druhej siete

Tentoraz sa podľa Randovho indexu na rozdelenie podľa krúžkov podobá viac rozdelenie získané CNM algoritmom. Je to z veľkej časti ovplyvnené aj počtom klastrov vo výsledných rozdeleniach.

Z praktického hľadiska sa môžeme na to pozrieť aj inak. Majme dve referentky zo študijného oddelenia a každá má na starosti jeden z týchto krúžkov. Obe majú k dispozícií výsledky z práve jedného algoritmu. Taktiež samozrejme poznajú rozdelenie do krúžkov z predošlého (zimného) semestra. Ich snahou bude vytvoriť z ročníku dva krúžky, pričom nechcú rozpojiť predpokladané komunity (spokojnosť študentov) a zároveň si želajú, aby väčšina študentov ostala v rovnakom krúžku ako v zimnom semestri (menšia byrokracia). Študijné referentky teda vytvoria nasledovné zadelenia:



Obr. 21: Zľava: zadelenie podľa GN, zadelenie podľa CNM (modrý podklad - prvý krúžok v zimnom semestri, zelený podklad- druhý krúžok v zimnom semestri)

Ktoré z týchto dvoch zadelení si nakoniec na študijnom oddelení vyberú? Po porov-

naní usúdia, že zadelenie tej referentky, ktorá mala k dispozícií výsledky z CNM algoritmu, je výhodnejšie. Inak zatriedilo len študentov 15, 29 a 30, kdežto v prvom zadelení došlo ešte k nezhodnému zatriedeniu študenta 21. Teda či už podľa Randovho indexu alebo aj po pospájaní rozdelení do dvoch klastrov, rozdelenie podobnejšie pôvodnému našiel CNM algoritmus.

Vo všeobecnosti však možno pozorovať, že iba malý počet študentov bol súčasťou opačného krúžku ako väčšina členov jeho komunity, teda v zimnom semestri sme sa rozdelili do krúžkov výrazne podľa kamarátstiev (za predpokladu správne ohodnotených vzťahov).

6.2.2 Aplikácia FF algoritmu

Na prvý pohľad by sa mohlo zdať, že FF algoritmus by mohol byť na túto sieť dobre aplikovateľný, pretože ako výstup dostaneme dva klastre, ktoré by mohli za predpokladu podobných veľkostí reprezentovať dva krúžky. Opäť však nastáva kľúčová otázka, ako správne zvoliť zdroj a ústie danej siete. Zistili sme však, že akokoľvek boli zvolené, vždy dochádzalo k nasledovnému javu. Jeden vrchol (zdroj alebo ústie podľa toho, ktorý mal menší počet väzieb) sa oddelil od ostatných. Tento jav signalizuje, že sieť dosahuje takú veľkú mieru hustoty väzieb a prepojení medzi vrcholmi, že algoritmu sa za každých okolností viac "oplatí" odrezať len jeden vrchol, akoby mal rez prechádzať pomedzi dve väčšie skupiny vrcholov. Sieť teda nevykazuje známky výrazného delenia sa na práve dve komunity.

Záver

V našej práci sme sa venovali problemtike detekovania komunít v sociálnych sieťach. Konkrétne sme sa zamerali na Ford-Fulkersonov, Girvan-Newmanov a *greedy* algoritmus hľadajúci maximálnu modularitu a následne ich aplikovali na sociálne siete.

Jedným z hlavných cieľov bolo prehľadným a objasňujúcim spôsobom vysvetliť fungovanie vybraných algoritmov. Naša práca začína predstavením základných pojmov z teórie grafov a zadefinovaním niektorých dôležitých termínov. V druhej, tretej a štvrtej kapitole sa postupne venujeme vybraným algoritmom. Vysvetľujeme základnú motiváciu, ale i podrobný postup. Nechýbala ani ukážka výpočtov na malorozmerných príkladoch. Taktiež sme čitateľa oboznámili s príkazmi v softvéri R, využiteľnými na spustenie algoritmov. Pre rozšírenie obzoru a predstavy sme v piatej kapitole stručne uviedli iné možné spôsoby nazerania a riešenia danej problematiky.

V praktickej časti sme zo zozbieraných dát namodelovali dve reálne sociálne siete. V prvom prípade šlo o sieť spoločných publikácií pracovníkov z troch oddelení z Katedry aplikovanej matematiky a štatistiky. Ako druhú sme analyzovali sociálnu sieť spolužiakov z tretieho ročníka EFM, ktorá bola modelovaná na základe výpovedí daných študentov. Následne sme pre oba prípady analyzovali nájdené rozdelenia sietí do komunít podľa jednotlivých algoritmov. Zamerali sme sa najmä na GN a CNM algoritmy, ktoré sú si svojim charakterom bližšie a sú vo všeobecnosti komplexnejšie. Výsledné rozdelenia sú znázornené na obrázkoch 14 a 18. Za cieľom zistenia, ako sa im darilo v hľadaní komunít, sme pre jednotlivé rozdelenia vypočítali hodnoty nadobudnuté viacerými funkciami kvalít, viď tabuľky 1 a 4 a porovnali ich. V prvej sieti sme ďalej pre zaujímavosť skúmali jav, či pracovníci katedry mali tendenciu takmer výlučne spoločne publikovať len vrámci jednotlivých oddelení alebo prichádzalo k spoločným publikáciam aj medzi oddeleniami. Prišli sme k záveru, že zatriedenie podľa oddelení sa do istej miery podobá výsledným rozdeleniam do komunít, no napriek tomu reálne zatriedenie nedosahuje takú hodnotu modularity ako nájdené rozdelenia. Tieto rozdelenia podľa algoritmov však v niektorých prípadoch pospájali do rovnakých klastrov aj pracovníkov z rôznych oddelení, teda dochádzalo aj k povšimnuteľnej spolupráci medzi určitými jednotlivcami z rôznych oddelení. V druhej sociálnej sieti študentov EFM sme zistili, že nájdené komunity podľa oboch algoritmov do značnej miery spadajú do

rozdelenia do krúžkov v zimnom semetri. Nakoniec sme v podsekciách 6.1.2 a 6.2.2 aplikovali FF algoritmus. V prípade druhej siete sme zistili, že kvôli vysokej hustote hrán algoritmus nevedel nájsť rozumné rozdelenie na dve komunity.

Možným rozšírením by prirodzene mohlo byť bližšie vysvetlenie iných algoritmov a ich aplikácia na naše siete. Konkrétne by sme navrhli študijným referentkám v prípade požiadavky zatriedenia študentov do veľkostne vyrovnaných krúžkov využiť Kernighan-Linov algoritmus, kde by ako vstup mohlo byť zadané rozdelenie zo zimného semestra. Taktiež by sa dalo zaoberať triedou takých algoritmov, ktoré sú schopné ako výstup dostať prekrývajúce sa komunity. Vďaka týmto algoritmom sa dajú detekovať napríklad dôležité osoby spájajúce viaceré komunity.

Literatúra

[1] Algorithms: Binary Min — Max Heap, dostupné na internete (14.5.2017):

http://algorithms.tutorialhorizon.com/binary-min-max-heap/

[2] Clauset,A., Newman,M. E. J., Moore,C.: *Finding community structure in very large networks* Physical Review E 70 (2004), dostupné na internete(14.5.2017): https://arxiv.org/pdf/cond-mat/0408187.pdf

[3] Complex networks, dostupné na internete (14.5.2017):

https://ifisc.uib-csic.es/users/jramasco/ComplexNets.html

[4] Emmons, S., Kobourov, S., Gallant, M., Börner, K.: *Analysis of Network Clustering Algorithms and Cluster Quality Metrics at Scale*, PLoS ONE 11 (2016), dostupné na internete (14.5.2017):

http://doi.org/10.1371/journal.pone.0159161

[5] Evidencia publikačnej činnosti UK, dostupné na internete (14.5.2017):

http://alis.uniba.sk:9909/search/query?theme=EPC

[6] FMFI: Katedra aplikovanej matematiky a štatistiky, dostupné na internete (14.5.2017): https://fmph.uniba.sk/pracoviska/katedra-aplikovanej-matematiky-a-statistiky/

[7] Fortunato, S.: *Community detection in graphs*, Physics Reports 486 (2010), 75-174

[8] Freeman, L. C.: *A Set of Measures of Centrality* , Sociometry 40 (1977), 35-41

[9] In theory: Maximum Flow, dostupné na internete (9.12.2016): https://lucatrevisan.wordpress.com/2011/02/04/cs261-lecture-9-maximum-flow/

[10] Kernighan, B. W., and S. Lin: *An efficient heuristic procedure for partitioning graphs* , Bell System Tech. J. 49 (1970), 291-307

[11] Knor, M., Niepel, Ľ.: *Kombinátorika a teória grafov*, Vydavateľstvo UK, Bratislava, 2000

[12] Newman,M. E. J.: *Analysis of weighted networks*, Physical Review E 70 (2004), dostupné na internete (14.5.2017):

`https://arxiv.org/pdf/cond-mat/0407503.pdf`

[13] Newman,M. E. J.: *Fast algorithm for detecting communit structure in networks* Physical Review E 69 (2004) dostupné na internete (14.5.2017):

`https://arxiv.org/pdf/cond-mat/0309508.pdf`

[14] Newman, M. E. J., Girvan, M.: *Finding and evaluating community structure in networks*, Physical Review E 69 (2004), dostupné na internete (14.5.2017):

`https://arxiv.org/pdf/cond-mat/0308217.pdf`

[15] Package 'optrees', dostupné na internete (14.5.2017):

`https://cran.r-project.org/web/packages/optrees/optrees.pdf`

[16] Package 'igraph', dostupné na internete (14.5.2017):

`https://cran.r-project.org/web/packages/igraph/igraph.pdf`

[17] Wolfram: Stirling Number of the Second Kind, dostupné na internete (14.5.2017):
`http://mathworld.wolfram.com/StirlingNumberoftheSecondKind.html`

[18] Zachary, W. W.: *An information flow model for conflict and fission in small groups*, Journal of Anthropological Research 33 (1977), 452-473

Prílohy

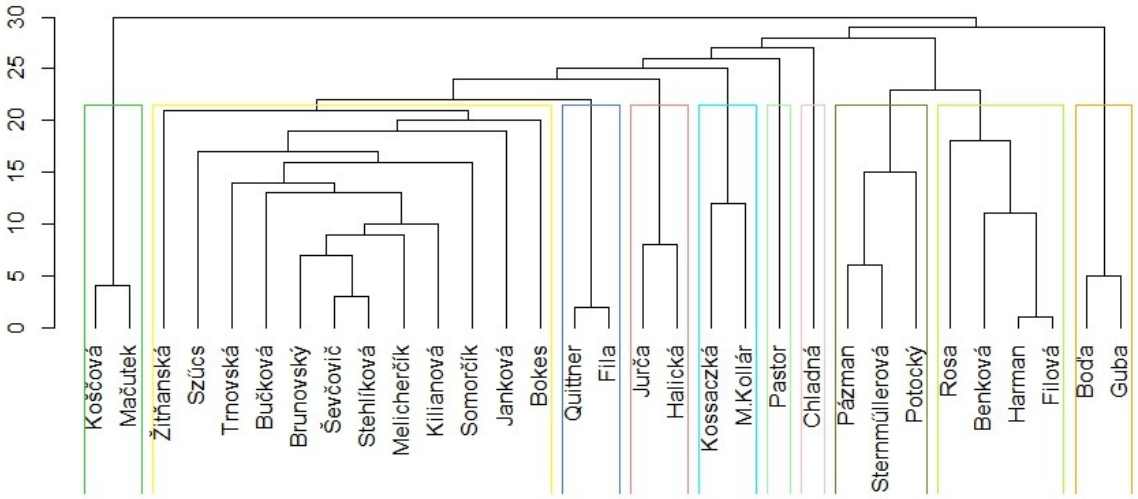
Príloha A

Tabuľka 6: Priradenie čísel k osobám

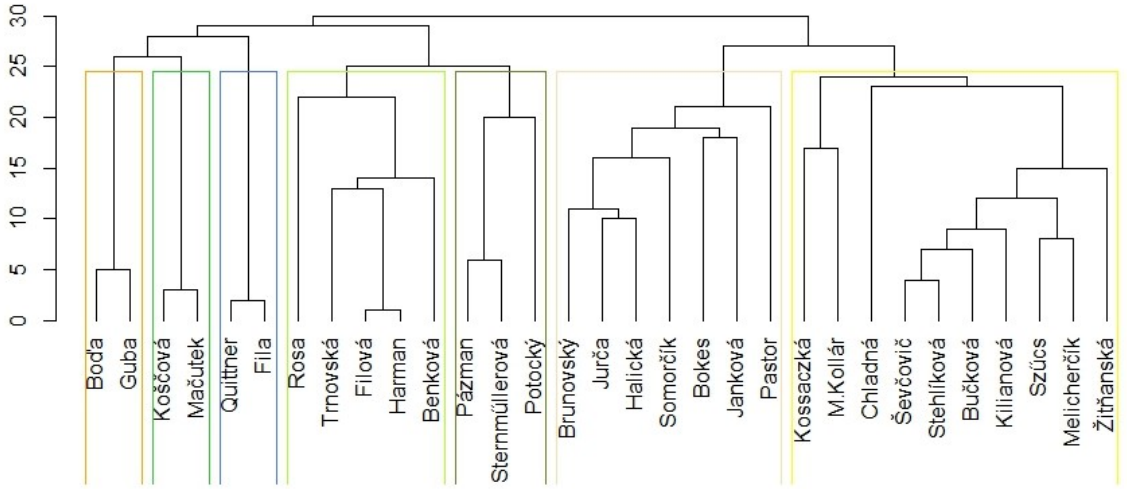
Číslo	Meno	Oddelenie
1	doc. Mgr. Pavol Bokes, PhD.	Oddelenie aplikovanej matematiky
2	prof. RNDr. Marek Fila, DrSc.	Oddelenie aplikovanej matematiky
3	doc. RNDr. Peter Guba, PhD.	Oddelenie aplikovanej matematiky
4	Mgr. Martin Kollár, PhD.	Oddelenie aplikovanej matematiky
5	RNDr. Ľubica Kossaczká, CSc.	Oddelenie aplikovanej matematiky
6	prof. RNDr. Pavol Quittner, DrSc.	Oddelenie aplikovanej matematiky
7	doc. RNDr. Ján Boďa, CSc.	Oddelenie ekonomických a finančných modelov
8	Dr. Zuzana Chladná	Oddelenie ekonomických a finančných modelov
9	doc. RNDr. Margaréta Halická, CSc.	Oddelenie ekonomických a finančných modelov
10	Mgr. Soňa Kilianová, PhD.	Oddelenie ekonomických a finančných modelov
11	doc. Mgr. Igor Melicherčík, PhD.	Oddelenie ekonomických a finančných modelov
12	doc. RNDr. Beáta Stehlíková, PhD.	Oddelenie ekonomických a finančných modelov
13	prof. RNDr. Daniel Ševčovič, CSc.	Oddelenie ekonomických a finančných modelov
14	doc. RNDr. Mária Trnovská, PhD.	Oddelenie ekonomických a finančných modelov
15	Mgr. Lenka Filová, PhD.	Oddelenie štatistiky a poistnej matematiky
16	doc. Mgr. Radoslav Harman, PhD.	Oddelenie štatistiky a poistnej matematiky
17	doc. RNDr. Katarína Janková, CSc.	Oddelenie štatistiky a poistnej matematiky
18	doc. Mgr. Ján Mačutek, PhD.	Oddelenie štatistiky a poistnej matematiky
19	doc. RNDr. Karol Pastor, CSc.	Oddelenie štatistiky a poistnej matematiky
20	doc. RNDr. Rastislav Potocký, CSc.	Oddelenie štatistiky a poistnej matematiky

Číslo	Meno	Oddelenie
21	Mgr. Ján Somorčík, PhD.	Oddelenie štatistiky a poistnej matematiky
22	Mgr. Gábor Szűcs, PhD.	Oddelenie štatistiky a poistnej matematiky
23	Mgr. Eva Benková	Oddelenie štatistiky a poistnej matematiky
24	Mgr. Zuzana Bučková	Oddelenie ekonomických a finančných modelov
25	Mgr. Katarína Sternműllerová	Oddelenie štatistiky a poistnej matematiky
26	Mgr. Michaela Koščová	Oddelenie štatistiky a poistnej matematiky
27	Mgr. Samuel Rosa	Oddelenie štatistiky a poistnej matematiky
28	prof. RNDr. Pavel Brunovský, DrSc.	Oddelenie ekonomických a finančných modelov
29	prof. RNDr. Andrej Pázman, DrSc.	Oddelenie štatistiky a poistnej matematiky
30	Mgr. Ing. Pavol Jurča, PhD.	Oddelenie ekonomických a finančných modelov
31	Mgr. Magdaléna Žitňanská, PhD.	Oddelenie ekonomických a finančných modelov

Príloha B



Obr. 22: Spoluautorstvá publikácií - dendrogram zachytávajúci proces rozpadu v GN algoritme



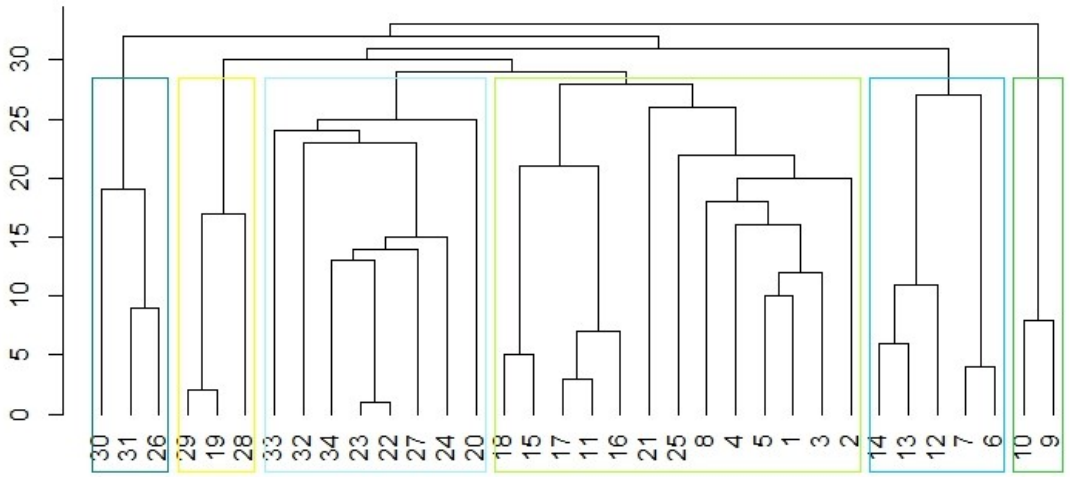
Obr. 23: Spoluautorstvá publikácií - dendrogram zachytávajúci proces zlučovania v CNM algoritme

Príloha C

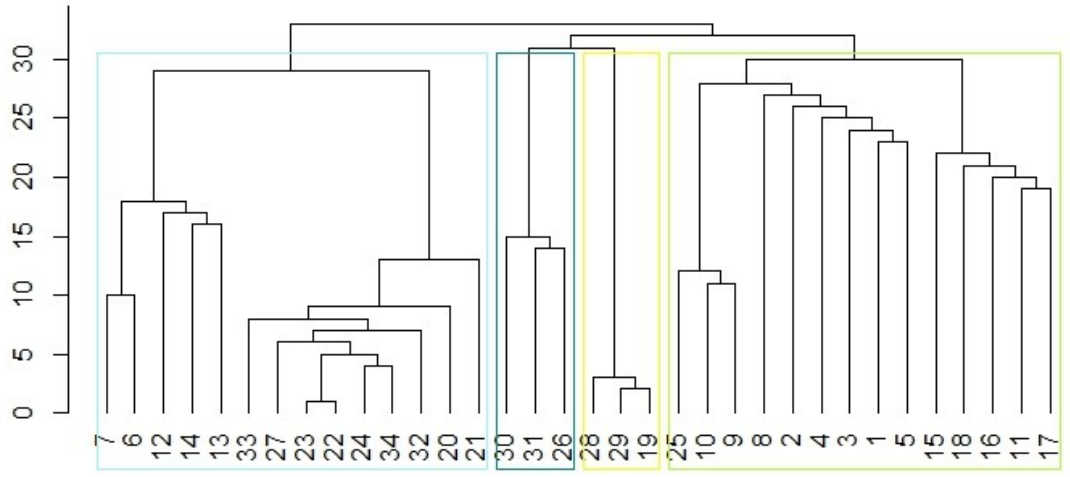
Tabuľka 7: Popis ohodnocovania väzieb medzi spolužiakmi z ročníka

Aspekt	Popis a spôsob ohodnotenia kapacít väzieb
Skupinové projekty	Za každý spoločný projekt - plus 1
Facebook	Ak sa majú v priateľoch - plus 1
Telefónne číslo	Ak má jeden na druhého číslo - plus 1, ak majú oboja navzájom - plus 2
Spolubývajúci na intráku	Ak bývajú osoby spolu na bunke - plus 1 a ak dokonca na izbe - plus 2
Spolužiaci zo strednej	Ak boli spolužiaci na strednej - plus 1
Krúžky	Ak boli za minulé roky spolu v krúžku - plus 1
Obedy	Ohodnotili od $0-3$, ako často s danou osobou obedujú - plus zaokrúhlený priemer týchto dvoch čísel poskytnutých jednou a druhou osobou
Doplňujúce	Bratské puto - plus 2, skupine, čo pravidelne minulé dva roky chodili spolu na lyžiarsky plus 1, spoločný mimoškolský projekt plus 1

Príloha D



Obr. 24: Spolužiaci z EFM - dendrogram zachytávajúci proces rozpadu v GN algoritme



Obr. 25: Spolužiaci z EFM - dendrogram zachytávajúci proces zlučovania v CNM algoritme

Príloha E

Kódy využité v kapitole 6:

```
# COVERAGE, vstupy:
# mat - matica susednosti, resp. s kapacitami
# mem - vektor membership priradujuci kazdemu vrcholu v poradi
#       cislo klastru, do ktoreho bol zatriedeni
coverage <- function(mat, mem)
{
  n <- length(mem)
  A <- 0
  for(i in 1:n)
  {
    for(j in 1:n)
    {
      # ak su z rovnakeho klastra, zapocitame
      # ich prepojenie
      if(mem[i]==mem[j])
      { A = A + mat[i,j] }
    }
  }

  cov <- A / sum(colSums(mat))
  return(cov)
}
```

```
# CONDUCTANCE, vstupy rovnake ako pri coverage
conductance <- function(mat, mem)
{
  n <- length(mem)
  cond <- 0
  nc <- length(table(mem))
  cit <- 0
  kc <- 0
  kcg <-0
  vec <- rep(0, nc)
  for(l in 1:nc)
  {
    for(i in 1:n)
    {
      for(j in 1:n)
      {
        # citatel - pocet prepojeni medzi klastrom l
        # a zvyskom grafu
        if(xor(mem[i]==l,mem[j]==l))
        { cit = cit + mat[i,j] }

        # stupen vrcholov klastra l
        if(mem[i]==l || mem[j]==l)
        { kc = kc + mat[i,j] }

        # stupen vrcholov mimo klastra l
        if(mem[i]!=l || mem[j]!=l)
        { kcg = kcg + mat[i,j] }
      }
    }
    # conductance klastra l
```

```
    vec[l] <- cit/min(kc,kcg)
    cit <- 0
    kc <- 0
    kcg <- 0
  }
  # zaverecny vypocet
  cond <- 1 - 1/nc*sum(vec)
  return(cond)
}


# RANDOV INDEX, vstup dva membership vektory
RandovIndex <- function(mem1, mem2)
{  n <- length(mem1)
  agree <- 0
  for(i in 1:n)
  {
    for(j in 1:n)
    {
      if(i != j)
      {if((mem1[i]==mem1[j] & mem2[i]==mem2[j]) |
      (mem1[i]!=mem1[j] & mem2[i]!=mem2[j]))
        {
          agree <- agree + 1
        }
      }
    }
  }
  rand <- agree/(n*(n-1))
  return(rand)
}
```