Univerza *v Ljubljani*
Fakulteta *za računalništvo*
in informatiko

Lovro Šubelj

ODKRIVANJE SKUPIN VOZLIŠČ V VELIKIH REALNIH OMREŽJIH NA OSNOVI IZMENJAVE OZNAK

Doktorska disertacija

Mentor: izr. prof. dr. Marko Bajec

Ljubljana, 2013

Zahvaljujem se mentorju, mami in očetu ter vsem ostalim, ki so kakorkoli pripomogli k izdelavi pričujoče doktorske disertacije.

— Lovro Šubelj, Ljubljana, 2013

Povzetek

Svetovni splet, nevronska mreža, "Facebook" in vodovod so vse primeri kompleksnih omrežij, sestavljenih iz velikega števila med seboj povezanih komponent (t.j. vozlišč). Pri tem pa je v različnih realnih omrežjih moč opaziti karakteristične vzorce povezovanja, ki jih značilno ločijo od regularnega ali naključnega sveta. Proučevanje zgradbe velikih realnih omrežij je tako trenutno izjemno aktivno področje v fiziki, matematiki, računalništvu in informatiki, ter tudi drugod, uporaba za-to razvitih metod in pristopov pa predstavlja temelj številnih uspešnih podjetij (npr. "Google").

Pričujoča doktorska disertacija se v prvi vrsti ukvarja z razvojem pristopov za odkrivanje karakterističnih skupin vozlišč v velikih kompleksnih omrežjih (npr. tesno povezanih skupin vozlišč). Predstavljene so različne metode in tehnike odkrivanja skupin na osnovi izmenjave oznak med vozlišči, ki močno izboljšajo robustnost in natančnost izmenjave, hkrati pa pristop razširijo še na odkrivanje splošnejših skupin vozlišč. Predlagane razširitve in izboljšave tako skupaj predstavljajo celovito rešitev za odkrivanje skupin, pri tem pa za razliko od večine drugih pristopov ne zahtevajo nikakršnega predhodnega znanja o zgradbi omrežja. Uspešnost na sintetičnih in realnih omrežjih z znano zgradbo je vsaj primerljiva z najboljšimi obstoječimi pristopi, dočim je asimptotična časovna zahtevnost blizu idealne. Poleg tega so predlagani pristopi enostavni za razumevanje in implementacijo, ter omogočajo vključitev poljubnega dodatnega znanja o proučevani domeni. Poudarimo, da se z omenjenimi lastnostmi lahko ponašajo le redki drugi pristopi v literaturi.

Pričujoča disertacija se osredotoča tudi na proučevanje skupin vozlišč v omrežjih programja, ki verjetno predstavlja eno od najkompleksnejših človeških stvaritev, a je le malo znanega o zgradbi kakovostnega programja. Analize kažejo, da programska omrežja vsebujejo izrazite skupine vozlišč, ki razmeroma dobro pojasnijo njihovo zgradbo, hkrati pa hierarhije skupin delno sovpadajo z organizacijo pripadajočih programskih knjižnic. Na podlagi slednjega so predstavljeni tudi različni praktični primeri uporabe odkrivanja skupin v programskem inženirstvu.

Doktorska disertacija temelji na štirih objavljenih delih, ki skupaj tvorijo jedro besedila, ter so v disertacijo vložena v enaki obliki, kot v publikaciji, ki jih je izdala.

Ključne besede

teorija ali analiza omrežij, skupine vozlišč, skupnosti, moduli, odkrivanje skupin, izmenjava oznak, realna omrežja, programska omrežja, programsko inženirstvo

Abstract

Detecting groups of nodes in large real-world networks using label propagation

The World Wide Web, wiring of a neural system, "Facebook" and a plumbing are all examples of complex networks composed of a large number of interconnected components denoted nodes. Many such real-world networks reveal characteristic patterns of connectedness that are far from regular or random. Analysis of the structure of large real-world networks is thus an active research field in physics, mathematics, computer science and informatics, whereas network analysis also represents the foundations of many successful companies (e.g., "Google").

Present doctoral thesis focuses on the development of approaches for detection of characteristic groups of nodes in large complex networks. We present different group detection methods and techniques based on label propagation that improve its robustness and accuracy, and extend the approach to more general groups of nodes. Presented advances represent a complete solution for group detection. In contrast to most other approaches, ours do not require any prior knowledge about the structure of the network. Performance on synthetic and real-world networks with known structure is at least comparable to current state-of-the-art approaches, while the asymptotic complexity is near ideal. Furthermore, presented approaches allow simple comprehension and implementation, and also a straight-forward incorporation of arbitrary knowledge about the underlying domain. Note that the above-mentioned properties are met by only a few other approaches in the literature.

The doctoral thesis also focuses on the analysis of groups of nodes in software networks. Complex software is probably one of the most sophisticated human-made systems, however, only little is known about the actual structure of high-quality software projects. Analyses show that software networks contain significant groups of nodes that are relatively well depicted in the network structure, whereas the group

hierarchy also roughly coincides with the organization of the corresponding program libraries. Based on the latter, we present different practical applications of group detection in software engineering.

Doctoral thesis is based on four published papers that together constitute the core of the thesis, and are included in the same form as published.

Keywords

network theory or analysis, node groups, communities, modules, group detection, label propagation, real-world networks, software networks, software engineering

Kazalo

1

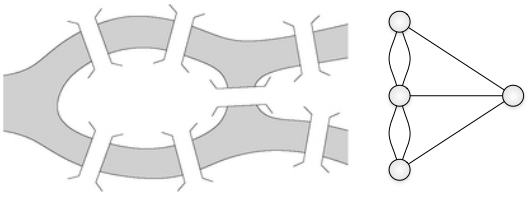
Uvod

Pred več kot tristo leti je prebivalce pruskega Königsberga, danes Kaliningrad v Rusiji, mučila naslednja matematična zagonetka. Skozi mesto teče reka Pregel, med bregovoma reke pa ležita dva otoka. Preko reke je v tistem času vodilo sedem mostov, kot to prikazuje slika 1.1(a). Prebivalci so se vrsto let spraševali, ali je moč prečkati vse mostove v mestu tako, da vsakega od njih obiščemo natanko enkrat? Se je pri tem moč vrniti na začetno mesto? Omenjeni vprašanji danes poznamo kot "Problem königsberških mostov" (angl. "Königsberg bridge problem").

Rešitev je leta 1736 podal švicarski matematik Leonard Euler [1]. Pri tem pa je bila ključnega pomena ustrezna predstavitev problema. Euler je namreč opazil, da je pot, ki jo opravimo po kopnem, za rešitev nepomembna. Povedano drugače, pomemben je le vrstni red, v katerem prečkamo mostove. Bregova reke in otoka je zato predstavil zgolj s štirimi točkami, mostove pa s črtami med njimi. Dobljenemu objektu pravimo matematični *graf* in je prikazan na sliki 1.1(b). Točke in črte grafa označujemo z izrazoma *vozlišča* (angl. *vertex, node*) in *povezave* (angl. *edge, link*), številu povezav, ki so krajišče nekega vozlišča, pa pravimo *stopnja* (angl. *degree*).

Problem prečkanja mostov je ekvivalenten sprehodu po grafu, kjer vsako povezavo obiščemo natanko enkrat. Euler pa je pri tem poudaril, da mora tak sprehod vsako obiskano vozlišče tudi zapustiti. Tako neposredno sledi, da imajo vsa vozlišča v grafu sodo stopnjo. Natančneje, obstajata lahko le dve vozlišči z liho stopnjo, ki predstavljata začetek in konec sprehoda. Ker so v zgornjem grafu štiri vozlišča z liho stopnjo, "Problem königsberških mostov" nima rešitve. (Rešitev obstaja, v kolikor na poljubno mesto dodamo most.)

Eulerjevo delo navadno smatramo za začetek *teorije grafov* (angl. *graph theory*), področja matematične kombinatorike, ki se ukvarja s proučevanjem grafov. S soro-



(a) (b)

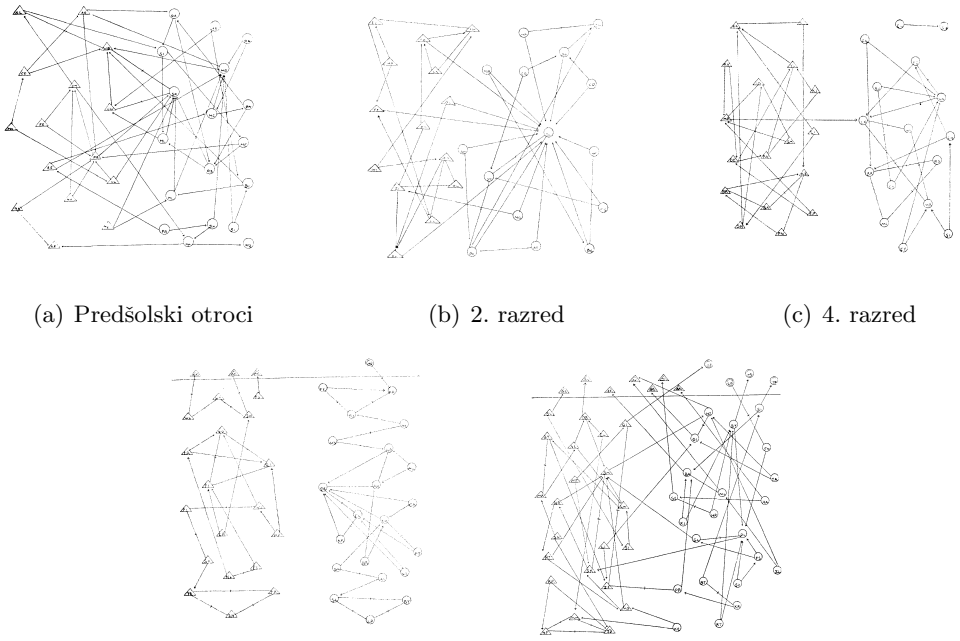
Slika 1.1: "Problem königsberških mostov" [1]. (a) Mostovi preko reke Pregel v mestu Königsberg v 18. stoletju. (b) Pripadajoč matematični graf s štirimi vozlišči (bregova reke in otoka) ter sedmimi povezavami (mostovi). Slika (a) je povzeta po [2].

dnimi problemi se je približno stoletje kasneje ukvarjal tudi irski fizik William R. Hamilton ter še utrdil področje. Nemški fizik Gustav R. Kirchhoff je leta 1845 s svojimi zakoni postavil osnove analize električnih vezij, kemik F. August Kekulé pa je dobrih deset let po tem nakazal uporabo grafov še v molekularni kemiji. Teorija grafov se tako že v tistem času izkaže kot nepogrešljivo orodje za proučevanje kompleksnih sistemov, sestavljenih iz velikega števila med seboj povezanih komponent. *Podrobnejši pregled sodobne teorije grafov najdemo v [3, 4].*

V 30. letih preteklega stoletja so podobno zanimanje pokazali tudi v družboslovnih znanostih. Romunski psiholog Jacob L. Moreno, ki je proučeval medsebojno delovanje med posamezniki v skupinah, je namreč v svoji knjigi leta 1934 objavil prve primere analize odnosov v družbi z uporabo grafom podobnih struktur (glej sliko 1.2), ki jih je poimenoval *sociogrami* [5] (angl. *sociogram*). Slednje štejemo za začetek *sociometrije* (angl. *sociometry*), ki med drugim predstavlja tudi temelj sodobne analize spletnih družbenih medijev (npr. "Facebook").

Podobno je v 70. letih ameriški sociolog Wayne Zachary proučeval odnose med člani karate kluba na severnoameriški univerzi [6]. Na podlagi neposrednega opazovanja je zgradil eno od danes verjetno najprepoznavnejših mrež prijateljstev, ki je prikazana na sliki 1.3(a). Tekom dvoletne raziskave je namreč prišlo do spora o višini članarine med lastnikom kluba in glavnim inštruktorjem, kar je povzročilo razpad kluba na dva dela. Pri tem pa je posebej zanimivo dejstvo, da je delitev v dve skupini jasno vidna že na sliki 1.3(a), ko spora, vsaj na načelni ravni, sploh še ni bilo. Omenimo, da se pričujoča doktorska disertacija med drugim ukvarja z razvojem pristopov za odkrivanje takih in podobnih skupin vozlišč.

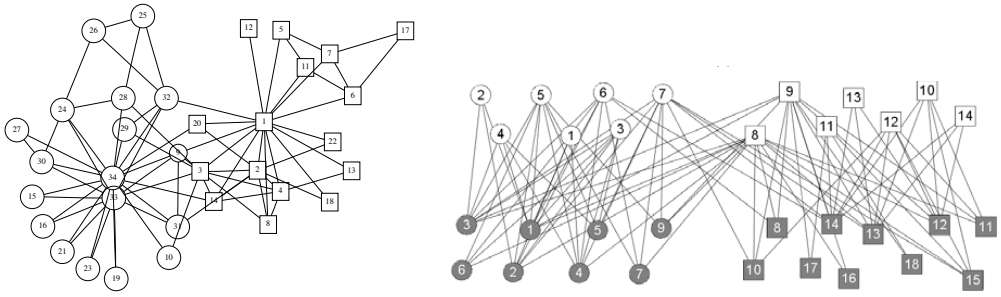
Poleg enostavnih mrež so bile pogosto proučevane tudi *dvodelne* (angl. *two-mode*) mreže (glej sliko 1.3(b)), ki jih predstavimo z *bipartitnimi grafi* (angl. *bipartite*



(a) Predšolski otroci (b) 2. razred (c) 4. razred

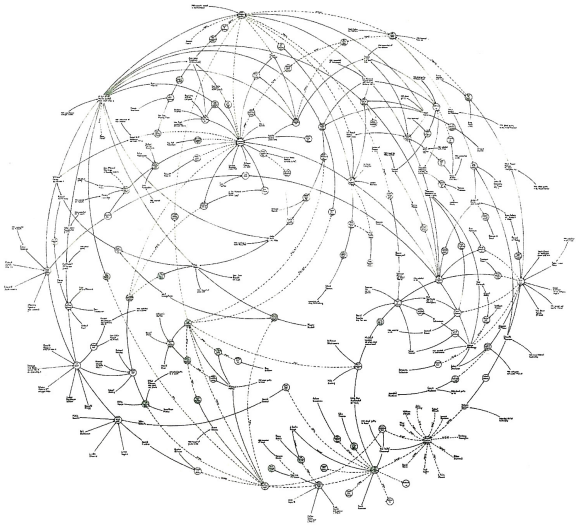
(d) 6. razred (e) 8. razred

Slika 1.2: Mreže prijateljstev med predšolskimi in osnovnošolskimi otroci, ki prikazujejo odnos med spoloma tekom odraščanja [5]. Vozlišča na levi strani ustrezajo dečkom (trikotniki), vozlišča na desni pa deklicam (točke). Slike so povzete po [5].



(a) (b)

Slika 1.3: (a) Mreža prijateljstev med člani univerzitetnega karate kluba [6]. Oblika vozlišč ustreza delitvi kluba po sporu med lastnikom in inštruktorjem. (b) Dvodelna mreža žensk (spodaj), ki so se udeležile istih družabnih dogodkov [7] (zgoraj). Oblika vozlišč ustreza razvrstitvi žensk in dogodkov v dve skupini. Sliki sta povzeti po [8, 9].

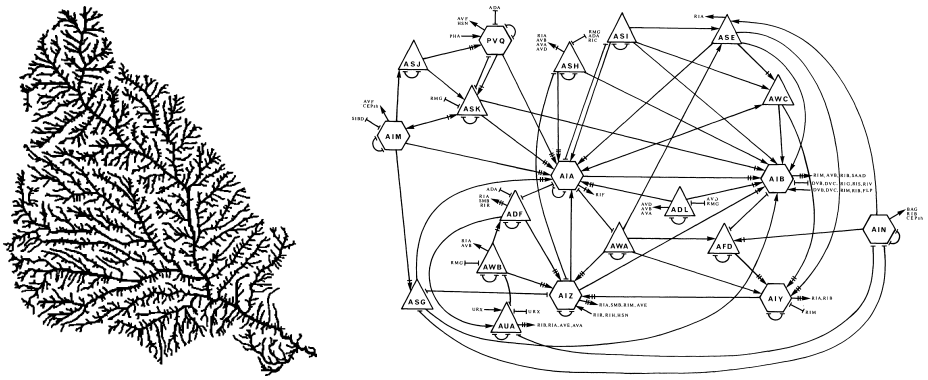


Slika 1.4: Povezave med različnimi osebami in organizacijami vpletenimi v politični škandal v 80. letih v Ameriki [17]. Slika je povzeta po [17].

graphs). Na razvoj sociometrije so pomembno vplivali še ameriški psiholog Stanley Milgram [10] ter sociologa Linton C. Freeman [11] in Mark Granovetter [12].
Podrobnejši pregled sodobne analiza socialnih mrež najdemo v [13, 14].

V času po drugi svetovni vojni je teorija grafov postala tudi pomembno področje *operacijskih raziskav* (angl. *operations research*). Pri tem so bili razviti številni učinkoviti algoritmi, ki so v uporabi še danes. Ameriški matematik Frank Harary in drugi so začeli razvijati teorijo *označenih grafov* [15] (angl. *signed graphs*), v katerih imajo povezave lahko pozitiven ali negativen predznak. Z vpeljavo verjetnostnih modelov v teorijo grafov pa sta leta 1959 madžarska matematika Paul Erdős in Alfréd Rényi osnovala področje *naključnih grafov* [16] (angl. *random graphs*).
Podrobnejši pregled sodobne teorije naključnih grafov najdemo v [2].

Grafi so tudi najnaravnejša predstavitev citiranj med npr. patenti ali znanstvenimi deli. Ker so v tem primeru povezave usmerjene, dobimo *usmerjene grafe* (angl. *directed graphs*). Leta 1965 je tako britanski fizik Derek J. de Solla Price objavil prvo analizo citiranj med skoraj milijon znanstvenimi deli [18], pri čimer je uporabil ročno zgrajeno bibliografsko zbirko. Slednja je kasneje postala "Science Citation Index" (SCI), ki je danes eden od najpopolnejših bibliografskih virov na svetu in hkrati eno od meril pri vrednotenju znanstvene uspešnosti. Z zbiranjem in proučevanjem bibliografskih podatkov pa se je v tem času začelo razvijati področje informacijskih ved, ki mu pravimo *bibliometrika* (angl. *bibliometrics*).



(a) (b)

Slika 1.5: (a) Rečno omrežje v kitajski provinci Shanxi [21]. (b) Del mreže nevronov pesne ogorčice "Caenorhabditis elegans" [20] (t.j. črv). Sliki sta povzeti po [20, 21].

Predstavitev z grafi se je izkazala tudi pri proučevanju rečnih ter drugih transportnih omrežij [19] (slika 1.5(a)). Poleg vsega omenjenega pa so v zadnjih desetletjih preteklega stoletja teorijo grafov s pridom začeli uporabljali tudi v organski kemiji in nevrologiji [20] (slika 1.5(b)), pri proučevanju gospodarskih in političnih škandalov [17] (slika 1.4) ter še ponekod drugje.

Ob koncu stoletja je področje doživelo silovit razmah. Z razvojem računalniške tehnologije in pojavom svetovnega spleta, ter različnih komunikacijskih omrežij, so znanstveniki začeli proučevati veliko večje grafe kot v primerih doslej. Dočim je slednje navadno sestavljalo največ nekaj deset ali sto vozlišč, ki jih brez težav narišemo in "analiziramo" s prostim očesom, pa so spletna, komunikacijska in nekatera druga sodobna omrežja vsebovala tudi do nekaj sto tisoč in več vozlišč ter povezav. V središču pozornosti niso bila več npr. posamezna vozlišča, temveč splošne lastnosti velikih realnih omrežij. Tako sta leta 1998 fizik Duncan J. Watts in matematik Steven H. Strogatz pokazala, da je v različnih realnih omrežjih opaziti fenomen *malega sveta* [22] (angl. *small-world*). Podobno sta leto kasneje fizika Albert-László Barabási in Réka Albert odkrila ter pojasnila obstoj *brezlestvičnih omrežij* [23] (angl. *scale-free networks*). Omenjena dosežka danes smatramo za začetek *teorije* ali *analize omrežij* (angl. *network theory, analysis*). (Omrežja malega sveta ter brezlestvična omrežja sta dejansko "opazila" že Milgram in de Solla Price zgoraj, pri tem pa oba pojma natančneje pojasnimo v razdelku 1.1.)

Sodobna analiza omrežij, za razliko od klasične teorije grafov, v prvi vrsti želi pojasniti zgradbo velikih kompleksnih omrežij, ki jih najdemo v realnem svetu. V

doktorski disertaciji tako dosledno ločimo med pojmoma graf in *omrežje* (angl. *network*). Dočim je omrežje abstrakten pojem, ki naj si ga vsak razlaga po svoje (npr. svetovni splet, nevronska mreža, vodovod itd.), pa je graf formalna predstavitev omrežja, ki jo privzamemo pri analizi. Seveda lahko pri tem morebitno dodatno znanje predstavimo z *oznakami* (angl. *label*) vozlišč ali utežmi na povezavah, pri čimer dobimo *utežene grafe* (angl. *weighted graphs*). Omrežje si sicer lahko predstavljamo tudi kot graf z dodatnim znanjem. Pri tem pa je potrebno poudariti, da moč teorije omrežij ne temelji na kompleksnosti posameznih povezav med komponentami proučevanega sistema, ali celo komponent samih (t.j. vozlišč), temveč na kompleksnih vzorcih medsebojnega povezovanja večjega števila komponent. Enostavnost predstavitve ter vedno lažja dostopnost velikih realnih omrežij pa sta verjetno dva ključna razloga za izjemno priljubljenost področja v zadnjih letih. Teorija omrežij je namreč v tem trenutku aktivno področje v fiziki, matematiki, računalništvu in informatiki, biologiji, družboslovnih znanostih, informacijskih vedah ter tudi drugje.

Podrobnejši pregled teorije omrežij najdemo v razdelku 1.1 in [2, 24–26].

Večino realnih omrežij je moč razvrstiti v zgolj štiri skupine (glej prve štiri spodaj), kljub temu pa je število vseh proučevanih omrežij danes preveliko za celovit pregled. V nadaljevanju tako zgolj kratko predstavimo značilne vrste realnih omrežij:

socialna — v *socialnih* ali *družbenih omrežjih* (angl. *social networks*) vozlišča predstavljajo ljudi ali živali (lahko tudi drugo), povezave pa ustrezajo proučevanim odnosom ali interakcijam med njimi. Sem sodijo klasične mreže prijateljstev (angl. offline social), spletna socialna ali družabna omrežja (angl. online social), dvodelna omrežja pripadnosti (angl. affiliation) ter omrežja sodelovanj med znanstveniki in filmskimi igralci (angl. author, actor collaboration);

informacijska — v *informacijskih omrežjih* (angl. *networks of information*) vozlišča ustrezajo nekim podatkovnim virom, povezave pa ponazarjajo tok informacij skozi proučevan sistem. Sem sodijo zgoraj omenjena omrežja citiranj (angl. citation), svetovni splet (angl. web (graph)), omrežja vsak z vsakim (angl. peer-to-peer) ter komunikacijska omrežja (angl. communication);

tehnološka — *tehnološka omrežja* (angl. *technological networks*) navadno predstavljajo neko umetno infrastrukturo, ki je podvržena tehnološkim ali drugim omejitvam. Sem sodijo internet (angl. Internet (map)), telefonsko (angl. telephone) in električno omrežje (angl. power (grid)), različna transportna omrežja (angl. transport) ter omrežja zgrajena iz programske kode (angl. software), ki

jih med drugim obravnavamo v pričujoči doktorski disertaciji;

biološka — *biološka omrežja* (angl. *biological networks*) ponazarjajo neko obliko interakcije med geni, celicami, nevroni itd. v živih organizmih. Sem sodijo gensko regulatorna (angl. gene regulatory), metabolična (angl. metabolic), beljakovinska (angl. protein-protein interaction) in nevronska omrežja (angl. neural);

ekološka — najznačilnejši predstavnik *ekoloških omrežij* (angl. *ecological networks*) so omrežja povezav "plenilec-plen" v nekem ekosistemu (angl. food (web));

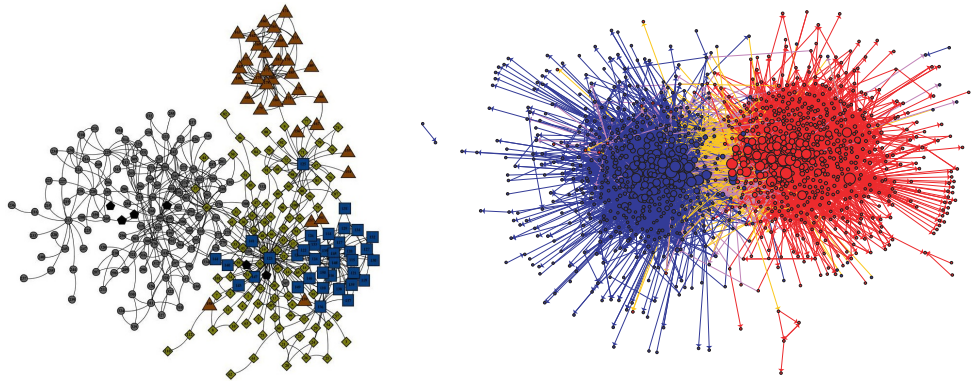
leksikalna — *leksikalna omrežja* (angl. *lexical networks*) predstavljajo strukturo nekega besedila kot je npr. knjiga ali spletna stran. Sem sodijo omrežja sosednosti besed (angl. word adjacency) ter semantična omrežja (angl. semantic);

ekonomska — med *ekonomska omrežja* (angl. *economic networks*) sodijo omrežja trgovanj med državami (angl. financial, trade), bančnih posojil (angl. bank, credit) ter industrijske proizvodnje (angl. production).

Mnoga realna omrežja ni moč razvrstiti v nobeno od omenjenih skupin. Mednje sodijo npr. omrežja živilskih sestavin (angl. ingredient) ali okusov (angl. flavor) ter klimatska (angl. climate) in potresna omrežja (angl. earthquake).

Zgornja delitev ni stroga, saj številna realna omrežja sodijo v različne skupine. Tako so omrežja med pošiljatelji in prejemniki elektronskih sporočil (angl. e-mail communication) obenem informacijska, saj ponazarjajo tok širjenja neke pomembne informacije ali virusa med posamezniki, ter hkrati socialna, saj si elektronska sporočila večinoma izmenjujemo s prijatelji, znanci, sodelavci itd. Podobno so omrežja zgrajena na podlagi objektno-usmerjene programske kode (glej sliko 1.6(a)) na eni strani tehnološka, saj so podrejena logičnim omejitvam in sintaktičnim pravilom programskega jezika, ter na drugi strani informacijska, saj je npr. funkcija v programu zgolj transformacija podatkov oziroma informacij iz ene oblike v drugo.

Tehnološko omrežje interneta je bilo gotovo med prvimi, ki so v literaturi požela obilo zanimanja (glej sliko 1.7). Sicer pa so bila v preteklih letih najpogosteje proučevana različna socialna in informacijska omrežja, kot so sodobna družabna omrežja in svetovni splet (glej sliko 1.6(b)), ter tudi biološka (npr. genetska, beljakovinska) in nekatera druga omrežja (npr. ekonomska). Zaradi vsesplošnosti omrežij v resničnem in navideznem svetu pa se v literaturi v zadnjem času pojavljajo številne nove vrste omrežij, ki ponujajo zanimive praktične primere uporabe (npr. omrežja športnih dogodkov ali glasbenih tokov). Analiza omrežij se namreč pogosto uporablja skupaj z metodami in pristopi *podatkovnega rudarjenja* (angl. *data mining*),



(a) (b)

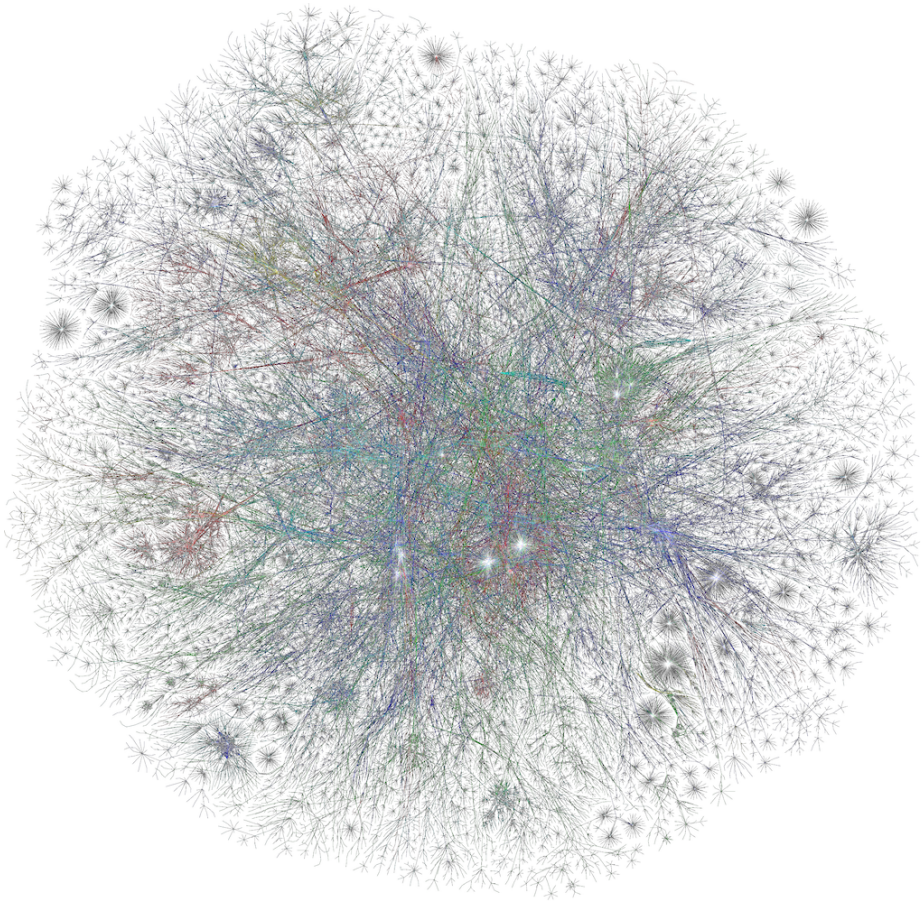
Slika 1.6: (a) Odvisnosti med programskimi razredi javanske knjižnice za delo z omrežji "JUNG" [27]. Barva vozlišč ponazarja štiri visokonivojske pakete knjižnice, oznake pa ustrezajo identifikatorjem pripadajočih razredov (glej sliko 2.6). (Oznake so bolje vidne v elektronski različici doktorske disertacije). (b) Spletni dnevniki (angl. blog) na temo predsedniških volitev v Ameriki leta 2004 [28]. Omrežje vidno razpade na dve skupini vozlišč, ki ustrezata delitvi na levo (modro) in desno (rdeče) usmerjene politične dnevnike. Slika (b) je povzeta po [28].

odkrivanja znanj iz besedil (angl. *text mining*) ter drugih področij *umetne inteligence* (angl. *artificial intelligence*), kar omogoča razvoj izjemno učinkovitih aplikacij.

Podobno je tudi velikost proučevanih omrežij v literaturi vse večja. Le-ta danes navadno vsebujejo od nekaj tisoč do nekaj sto tisoč vozlišč in povezav. Na standardnem računalniku je moč analizirati omrežja z nekje do sto milijonov povezav (z učinkovitimi algoritmi). Za primerjavo, med trenutno največja prosto dostopna omrežja sodi spletno omrežje "Yahoo!" z več kot milijardo vozlišč ter skoraj sedem milijard povezav [30], pripadajoča datoteka pa zasede 120GB fizičnega pomnilnika.

Danes so na voljo tudi številna orodja za proučevanje omrežij. Omenimo program "Pajek" [31, 32] slovenskih avtorjev ter "Gephi" [33] in "NWB" [34], knjižnice za delo z omrežji "JUNG" [27], "SNAP" [35] in "NetworkX" [36] (programski jeziki "Java", "C++" in "Python") ter dodatek za "Microsoft Excel" imenovan "NodeXL" [37].

V nadaljevanju natančneje opredelimo znanstveno področje (razdelek 1.1) ter predstavimo problemsko domeno doktorske disertacije (razdelek 1.2). Sledi opis izvirnih prispevkov k znanosti (razdelek 1.3) ter pregled vsebine (razdelek 1.4).



Slika 1.7: Omrežje več milijonov internetnih poti med podomrežji (angl. subnet) razreda "C" zbrano v letu 2003 [29]. Barve povezav ustrezajo različnim delom sveta, dočim postavitev vozlišč ne predstavlja dejanske geografske lokacije pripadajočih omrežnih naprav (ali skupin naprav). Slika je dostopna preko [29].

1.1 Znanstveno področje

Teorija omrežij se v prvi vrsti ukvarja s proučevanjem splošnih lastnosti in zgradbe velikih realnih omrežij, ter z razvojem za to potrebnih metod in pristopov [24]. Kljub veliki raznolikosti imajo realna omrežja namreč številne skupne lastnosti, ki jih značilno ločijo od enostavnih modelov naključnih omrežij [8, 22, 23, 38]. Omrežja lahko proučujemo na nivoju posameznih vozlišč ali povezav, na nivoju (karakterističnih) skupin vozlišč ali povezav ter na nivoju omrežja kot celote.

V preteklosti so se znanstveniki najpogosteje osredotočali na slednje. Sem sodi že prej omenjen pojav malega sveta [22], ki med drugim pravi, da je povprečna *razdalja* (angl. *distance*) med vozlišči (t.j. število povezav v najkrajši poti) v realnih omrežjih zelo majhna. Na primer, povprečna razdalja med 700 milijoni uporabnikov družabnega omrežja "Facebook" je v letu 2011 znašala zgolj 4,7 [39]. Podobno so mnoga realna omrežja brezlestvična [23], kar pomeni, da porazdelitev stopenj vozlišč sledi *potenčnemu zakonu* (angl. *power-law*). Posledica le-tega je pojav vozlišč z izredno visoko stopnjo imenovanih *zvezdišča* [40] (angl. *hub*), ki so jasno vidna tudi na sliki 1.7. Omenimo še, da so realna omrežja tudi izjemno *odporna* (angl. *robustness*) na naključne izpade vozlišč oziroma povezav, a hkrati občutljiva na namerne napade nad omrežjem [38]. Tako je internetno omrežje na sliki 1.7 izredno odporno na naključne izpade strojne opreme, dočim pa lahko napad na že manjše število ključnih komponent omrežja popolnoma ohromi internetni promet.

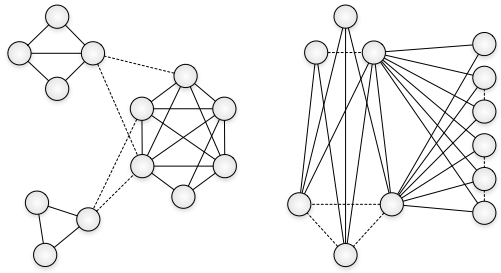
Podrobnejši pregled karakterističnih lastnosti realnih omrežij najdemo v [2, 24].

Dobre kandidate za napad predstavljajo zvezdišča in vozlišča z visoko *središčnostjo* [11, 41] (angl. *centrality*). Obstaja več mer središčnosti, ki upoštevajo npr. najkrajše poti ali pa središčnost *sosedov* (angl. *neighbour*), t.j. neposredno povezana vozlišča. Simetrična mera za primer povezav je *vmesnost* [8] (angl. *betweenness*).

Podrobnejši pregled karakterističnih vozlišč in povezav v realnih omrežjih najdemo v [2, 14].

Pogosto nas zanima tudi gostota omrežja v okolici nekega vozlišča, kar merimo s *koeficientom nakopičenosti vozlišča* [22] ali *omrežja* [22, 42] (angl. *node, network clustering coefficient*). Slednji je v primeru mrež prijateljstev kar enak verjetnosti, da je prijatelj prijatelja prav tako prijatelj. Omenjena verjetnost je v večini realnih omrežij relativno velika, kar pomeni, da so le-ta lokalno zelo *gosta* (angl. *dense*). Kljub temu pa velja, da so realna omrežja v splošnem izjemno *redka* [43, 44] (angl. *sparse*), kar je tudi eden ključnih dejavnikov pri razvoju učinkovitih algoritmov.

Ena od možnih razlag zgornjega "protislovja" je pojav karakterističnih *skupin vozlišč* (angl. *node group*) v realnih omrežjih. Leta 2002 sta namreč fizika Michelle



(a) (b)

Slika 1.8: Enostavna primera omrežij s tremi (a) skupnostmi (t.j. skupine tesno povezanih vozlišč) oziroma (b) moduli (t.j. skupine vozlišč s skupnimi sosedi). Povezave, ki sovpadajo z razvrstitvijo v skupine, so predstavljene s polno črto.

Girvan in Mark E. J. Newman odkrila, da številna socialna in biološka omrežja, ter tudi nekatera druga, vsebujejo *skupnosti* ali *komune* [8] (angl. *community*). Le-te navadno razumemo kot skupine tesno povezanih vozlišč, ki so med seboj dobro ločene (glej sliko 1.8(a)). Skupnosti ustrezajo posameznikom s sorodnimi interesi v socialnih omrežjih [45] (glej sliko 1.3(a)), stranem s podobno tematiko v spletnih omrežjih [28] (glej sliko 1.6(b)) ter beljakovinskim kompleksom v bioloških omrežjih [46]. V realnih omrežjih so skupnosti pogosto tudi *prekrivajoče* [46] (angl. *overlapping*), t.j. vozlišča lahko pripadajo več skupnostim hkrati, *samo-podobne* [47] (angl. *self-similar*), skupnosti so t.j. (približno) podobne delu samega sebe, ter med seboj povezane v kompleksne *hierarhije* [48] (angl. *hierarchical*).

Poudarimo, da skupnosti ni moč natančno opredeliti, saj se le-te v različnih vrstah omrežij izražajo drugače [49]. Kljub temu pa si pri delu lahko pomagamo z naslednjimi definicijami. *Krepka skupnost* [50, 51] (angl. *strong community*) je skupina vozlišč, kjer ima vsako vozlišče več povezav znotraj skupine kot izven nje. V kolikor slednje zahtevamo zgolj na nivoju skupine dobimo *šibke skupnosti* [51] (angl. *weak community*). Na drugi strani pa "idealni" skupnosti pravimo *klika* (angl. *clique*) in je definirana kot največja skupina med seboj povezanih vozlišč. Podobno je *k-klika* (angl. *k-clique*) največja skupina vozlišč, ki so na razdalji največ k. Sorodne zgornjim so še definicije *k-pleksa* (angl. *k-plex*), *k-jedra* (angl. *k-core*) in drugih.

Za proučevanje skupnosti v realnih omrežjih, ter posledično uporabo v praktičnih aplikacijah, je izjemno pomemben razvoj učinkovitih postopkov za njihovo odkrivanje. *Odkrivanje skupnosti* (angl. *community detection*) je bilo tako v zadnjem desetletju eno najpopularnejših področij v teoriji omrežij [52, 53]. Problem je NP-

težek [54], v posebnih primerih NP-poln [55] (t.j. učinkovit algoritem verjetno ne obstaja), kljub temu pa v literaturi najdemo na stotine različnih hevrističnih metod in pristopov [54, 56]. Med njimi pristopi, ki izhajajo iz teorije grafov [57], spektralni algoritmi [58, 59], statistični pristopi [60, 61], optimizacijske [62, 63] in več-resolucijske metode [64, 65], pristopi, ki temeljijo na dinamičnih procesih nad omrežji [66, 67] ter še mnogi drugi [51, 68]. Številni izmed predlaganih pristopov optimizirajo *modularnost* Q [69] (angl. *modularity*), ki meri kvaliteto skupnosti napram izbranemu modelu naključnih grafov [16]. Čeprav je v pomanjkanju drugih mer modularnost Q v literaturi postala "de facto" standard v odkrivanju skupnosti, pa je zaradi številnih pomanjkljivosti njena raba danes odsvetovana [70, 71].

Poleg skupnosti pa v omrežjih najdemo tudi druge karakteristične skupine vozlišč. Leta 2012 sva računalničarja Lovro Šubelj in Marko Bajec pokazala, da velika večina realnih omrežij vsebuje tudi jasno definirane *module* [72, 73] (angl. *module*). Slednji so skupine vozlišč, ki niso nujno povezana, a so podobno povezana z (ostalimi) vozlišči v omrežju (glej sliko 1.8(b)). Moduli sicer ustrezajo posameznikom s skupnimi interesi v dvodelnih socialnih omrežjih (glej sliko 1.3(a)), dokumentom s podobno vsebino v spletnih omrežjih, besedam iste vrste v leksikalnih omrežjih ter razredom z enako vlogo v programskih omrežjih [60, 72, 73]. Podobno kot skupnosti se moduli v realnih omrežjih verjetno prekrivajo ter tvorijo hierarhije [73, 74]. (Moduli so bili v posameznih omrežjih dejansko "opaženi" že pred tem.)

Terminologija skupin vozlišč v literaturi ni popolnoma enotna. Moduli so znani tudi kot angl. *link-pattern community*, dočim so skupnosti v tem primeru angl. *(link-density) community* [72, 75]. Poleg tega so splošne skupine vozlišč ponekod poimenovane angl. *module*, pri čimer so moduli angl. *functional module, class* [73, 76]. V sociometriji oziroma analizi socialnih mrež so skupnosti in moduli proučevani kot skupine *strukturno* [77], *regularno* [78, 79] in *splošno ekvivalentnih* [80] (angl. *structural, regular, generalized equivalence*) vozlišč, pri čimer sta dve vozlišči regularno (strukturno) ekvivalentni, če sta povezani z vozlišči, ki so prav tako ekvivalentna (enaka). Z odkrivanjem skupin ekvivalentnih vozlišč se ukvarja *bločno modeliranje* [81] (angl. *blockmodeling*) z začetki v 70. letih preteklega stoletja.

Poudarimo, da moduli pravzaprav vsebujejo skupnosti kot poseben primer, zato je odkrivanje le-teh, oziroma *odkrivanje skupin* (angl. *group detection*) vozlišč v splošnem, prav tako NP-težek problem. Za razliko od prej pa so metode in pristopi za odkrivanje splošnih skupin mnogo manj pogosti, kar lahko pripišemo povečani zahtevnosti ter pomanjkanju modelov oziroma znanja o skupinah v realnih omrežjih [72, 73]. V literaturi sicer najdemo metode na osnovi *gručenja* (angl. *clustering*) podat-

kov [82], statistične pristope [60, 83], optimizacijske [84] in kompresijske metode [85], matrične algoritme [61, 86], bločno modeliranje [80, 87] z razširitvami [88, 89], dinamične pristope [72, 73] ter še nekatere druge metode [90, 91].

Poleg skupin vozlišč lahko omrežja delimo tudi glede na *skupine povezav* [92, 93] (angl. *link community*). Le-te so bile sicer predlagane kot odgovor na vprašanje, kako prekrivajoče skupine povezati v hierarhije. Poleg omenjenega so bili v preteklosti pogosto proučevani tudi manjši vzorci vozlišč in povezav, kot so npr. *motifi* [94] (angl. *motif*) in *grafleti* [95] (angl. *graphlet*). Slednje smatramo za gradnike nekaterih tehnoloških in bioloških omrežij, pri tem pa so vzorci dveh oziroma treh vozlišč v analizi socialnih mrež poznani kot *diade* in *triade* [13, 14] (angl. *dyads, triads*).

Podrobnejši pregled karakterističnih skupin v realnih omrežjih najdemo v [13, 53, 54, 96].

Splošne lastnosti realnih omrežij so pogosto proučevane vzporedno z razvojem modelov naključnih grafov, ki smo jih omenili v prejšnjem razdelku. Čeprav predvsem želimo, da so dobljena naključna omrežja čimbolj podobna realnim, pri čimer se opremo na znane lastnosti, pa glavni namen razvoja modelov ni gradnja omrežij, temveč pojasnitev zgradbe realnih omrežij, oziroma pojavov, ki so prisotni v ozadju. Na drugi strani pa imajo lastnosti omrežij tudi močan vpliv na različne *dinamične procese* (angl. *network dynamics*), ki se odvijajo nad omrežji [97]. Sem sodi npr. proučevanje širjenja *epidemij* (angl. *epidemics*) v socialnih omrežjih ter *kaskadnih izpadov* (angl. *cascade failures*) preobremenjene mrežne opreme na internetu [98].

Podrobnejši pregled dinamičnih procesov v realnih omrežjih najdemo v [97, 98].

Številni od prej omenjenih pristopov odkrivanja skupin imajo vsaj kvadratno asimptotično časovno zahtevnost, zatorej niso primerni za proučevanje današnjih velikih omrežij. Poleg tega je implementacija algoritmov pogosto obsežna in zapletena, mnogi pristopi pa zahtevajo tudi določeno predznanje o omrežju (npr. število skupin). Slednjega v primeru velikih realnih omrežij danes ni moč natančno določiti [83]. V pričujoči doktorski disertaciji zato predstavimo različne metode in tehnike odkrivanja splošnih skupin vozlišč (t.j. skupnosti in modulov), ki temeljijo na izmenjavi oznak med vozlišči [66]. Časovna zahtevnost predlaganih pristopov je blizu idealne (t.j. linearne), le-ti pa so enostavni za implementacijo in ne zahtevajo nikakršnega predhodnega znanja (npr. število ali vrsto skupin). Uspešnost na sintetični in realnih omrežjih z znano zgradbo je primerljiva z najboljšimi pristopi v literaturi [72, 99].

Podrobnejši pregled prispevkov k znanosti najdemo v razdelku 1.3.

Doktorska disertacija se poleg razvoja pristopov za odkrivanje skupin vozlišč osredotoča tudi na proučevanje le-teh v programskih omrežjih (glej sliko 1.6(a)), katerih zgradba še ni dokončno pojasnjena. Za predstavitev programja so bila pre-

dlagana različna omrežja sodelovanj [100, 101] (angl. method, class, package collaboration network) in odvisnosti [102, 103] (angl. class, inter-package dependency networks), diagrami poteka [104] (angl. mirror graph) in arhitekturni diagrami [105] (angl. architecture map), omrežja sklopljenosti [106] (angl. coupling network) ter druga [107, 108]. V večini programskih omrežij je opaziti pojav malega sveta [100, 103] ter ponavljajoče se motife [109], omrežja pa so tudi hierarhična [100, 103] ter brezlestvična v kolikor jih predstavimo z usmerjenimi grafi [105, 107, 110]. V preteklosti je bila proučevana tudi odpornost in *odvisnosti* [111, 112] (angl. *mixing*) med stopnjami vozlišč [100, 106], poleg tega pa so bili predstavljeni še številni naključni modeli [100, 105, 113, 114]. Za odkrivanje ključnih vozlišč so bile predlagane mere na osnovi *naključnih sprehodov* [115, 116] (angl. *random walk*). Na drugi strani pa so bile skupine vozlišč v programskih omrežjih le redko proučevane [72, 108, 117, 118].

Podrobnejši pregled karakterističnih lastnosti programskih omrežjih najdemo v [119, 120].

Poudarimo, da je programje ena od najkompleksnejših človeških stvaritev, kljub temu pa je danes le malo znanega o zgradbi kakovostnega programja [104]. Analiza karakterističnih skupin vozlišč v programskih omrežij tako lahko pomembno pripomore k odkrivanju zakonitosti kompleksnega programja ter posledično privede do razvoja praktičnih aplikacij v *programskem inženirstvu* (angl. *software engineering*) in tudi na drugih področjih. Na primer, različne teorije nakazujejo, da skupine v omrežjih odvisnosti med komponentami objektno-umerjenega programja sovpadajo s programskimi knjižnicami, zbirkami oziroma paketi [110, 121], na čimer temeljijo tudi primeri uporabe, ki jih predstavimo v pričujoči disertaciji.

Podrobnejši pregled prispevkov k znanosti najdemo v razdelku 1.3.

1.2 Problemska domena

Doktorska disertacija se ukvarja predvsem z odkrivanjem in proučevanjem karakterističnih skupin vozlišč v velikih realnih omrežjih. V prvi vrsti so predlagane različne metode in tehnike odkrivanja skupin na osnovi *izmenjave oznak* (angl. *label propagation*), ki so jo sicer kot prvi leta 2007 predlagali Usha N. Raghavan in sodelavci [66]. Pri slednjem si sosednja vozlišča v omrežju med seboj izmenjujejo oznake, ki predstavljajo različne skupnosti. Vsako vozlišče privzame oznako, ki si jo deli večina njegovih sosedov, celoten postopek pa ponavljamo, dokler ni doseženo ravnovesno stanje [122]. Pristop ima skoraj linearno asimptotično časovno zahtevnost [66, 99], poleg tega pa zaradi svoje algoritmične enostavnosti omogoča preprosto implementacijo ter dopušča uporabo poljubnega domenskega predznanja [72, 73, 99, 123].

Pristop izmenjave oznak je bil razširjen tudi na prekrivajoče [124, 125] in hierarhične skupnosti [99, 126], skupnosti z visoko modularnostjo Q [127, 128], dvodelna in druga omrežja [9, 129], *tekoče* (angl. *stream*) obdelave [123, 130] ter tudi vzporedne računalniške arhitekture [131, 132]. V literaturi pa najdemo še številne druge izboljšave osnovnega pristopa [123, 133–135] ter tudi primere uporabe v analizi odpornosti in *stiskanju omrežij* [136, 137] (angl. *network compression*).

Kljub svoji učinkovitosti in enostavnosti, ter številnim izboljšavam, ki so bile predlagane v literaturi, pa ima pristop izmenjave oznak naslednje pomanjkljivosti:

robustnost — zaradi naključnega vrstnega reda obdelave (in posodabljanja) vozlišč, pristop že v manjših realnih omrežjih vrne večje število razvrstitev v skupine [122], ki so med seboj tudi relativno različne [138];

natančnost — v omrežjih z manj jasno zgradbo je natančnost pristopa izmenjave oznak slabša od najboljših pristopov v literaturi [99]. Podobno pristop v velikih realnih omrežjih pogosto večino vozlišč razvrsti v eno skupino [123]; in

splošnost — pristop izmenjave oznak je omejen na odkrivanje tesno povezanih skupin vozlič, zatorej lahko razkrije zgolj skupnosti v omrežju [72]. Poleg tega pristop morebitne module v realnih omrežju združi v skupnosti [73].

Pričujoča disertacija se osredotoča predvsem na odpravo omenjenih pomanjkljivosti izmenjave oznak, pri čimer ohranimo visoko (časovno) učinkovitost, algoritmično enostavnost ter splošno razširljivost pristopa. Delo temelji na predpostavki, da je slednje moč doseči z ustrezno strategijo preferenčnih vozlišč [123], hierarhičnim preiskovanjem ter analogijo med različnimi skupinami vozlišč v omrežju [72, 73]. Metode in tehnike odkrivanja skupin vozlišč, ki so predlagane v disertaciji, tako predstavljajo tri samostojne izvirne prispevke k znanosti (glej razdelek 1.3).

Zadnji, četrti, prispevek pa predstavlja analiza skupin vozlišč v kompleksnih programskih omrežjih. Pri tem privzamemo omrežja odvisnosti med razredi [103, 117] objektno-usmerjenega programja, v katerih naj bi morebitne skupine sovpadale s paketi pripadajočih programskih knjižnic [110, 121]. Delo temelji na predpostavki, da skupine vozlišč deloma ustrezajo programskim paketom, dočim pa je te navadno moč prerazporediti v bolj učinkovito obliko (t.j. bolj modularno ali funkcionalno obliko). Na podlagi dognanj analize lastnosti skupin pa so v disertaciji predstavljeni tudi različni primeri uporabe v programskem inženirstvu.

V disertaciji se omejimo na neprekrivajoče skupine ter predpostavimo, da je omrežja moč predstaviti z neusmerjenimi grafi. V nasprotnem primeru smeri pove-

zav enostavno zanemarimo. Dopuščamo pa uteži na povezavah ter več povezav med vozlišči, ne pa tudi povezav s samim seboj ali *zank* (angl. *loop*). Za razliko od *enostavnih grafov* (angl. *simple graph*), slednjim pravimo *multigrafi* (angl. *multigraph*).

1.3 Prispevki k znanosti

Doktorska disertacija predstavlja naslednje izvirne prispevke k znanosti:

— **metode in tehnike** odkrivanja skupin vozlišč v omrežjih, **ki izboljšajo robustnost pristopa izmenjave oznak**. Le-te temeljijo na povezavi med vrstnim redom izmenjave ter širjenjem oznak po omrežju. Uspešnost na sintetičnih in realnih omrežjih je primerljiva z drugimi pristopi v literaturi;
Prispevek je predstavljen v poglavju 3 ter objavljen v [138, 139].

— **metode in tehnike** odkrivanja skupin vozlišč v omrežjih, **ki izboljšajo natančnost pristopa izmenjave oznak**. Le-te temeljijo na različnih strategijah preferenčnih vozlišč ter hierarhičnemu preiskovanju omrežja. Uspešnost na sintetičnih in realnih omrežjih je primerljiva z najboljšimi pristopi v literaturi;
Prispevek je predstavljen v poglavju 4 ter objavljen v [73, 99, 126].

— **metode in tehnike** odkrivanja skupin vozlišč v omrežjih, **ki razširijo pristop izmenjave oznak na splošnejše skupine**. Le-te temeljijo na analogiji med različnimi skupinami vozlišč ter njihovimi lastnostmi. Uspešnost na sintetičnih in realnih omrežjih je primerljiva z najboljšimi pristopi v literaturi; in
Prispevek je predstavljen v poglavju 5 ter objavljen v [72, 73, 139].

— **analiza skupin vozlišč v omrežjih** odvisnosti med komponentami objektno-usmerjenega **programja**. Na podlagi rezultatov in dognanj analize so predlagani tudi različni **primeri uporabe v programskem inženirstvu**.
Prispevek je predstavljen v poglavju 6 ter objavljen v [72, 73, 117, 120].

1.4 Pregled disertacije

Pričujoča doktorska disertacija temelji na štirih objavljenih delih, ki skupaj tvorijo jedro besedila disertacije. V nadaljevanju tako v poglavju 2 najprej predstavimo vsebino, teoretično ozadje, zasledovane hipoteze ter znanstveni doprinos objavljenih del, dočim so sama dela v disertacijo vložena kot poglavja 3, 4, 5 in 6. Dela zaporedoma ustrezajo prispevkom k znanosti v razdelku 1.3 ter so vključena v enaki obliki,

kot v publikaciji, ki jih je izdala. Na koncu sledi še kratka razprava objavljenih del v poglavju 7 ter zaključek z izhodišči za nadaljnje delo v poglavju 8.

Podrobnejši pregled disertacije se nahaja znotraj posameznih poglavij.

2

Pregled objavljenih del

Doktorska disertacija temelji na štirih objavljenih delih s področja proučevanja in odkrivanja karakterističnih skupin vozlišč v velikih realnih omrežjih, ki so v disertacijo vložena kot poglavja 3, 4, 5 in 6. Kot smo omenili že v poglavju 1, se tri dela osredotočajo na izboljšave pristopa za odkrivanje skupin, ki temelji na izmenjavi oznak med vozlišči [66]. Pri tem prvi dve deli predlagata različne metode in tehnike, ki močno izboljšajo robustnost in natančnost izmenjave oznak [99, 138], tretje delo pa predstavi še razširitev osnovnega pristopa na bolj splošne skupine [72] (t.j. skupnosti in module). Asimptotična časovna zahtevnost predlaganih pristopov je blizu idealne (t.j. linearne), le-ti pa so enostavni za implementacijo ter ne zahtevajo predhodnega znanja o zgradbi omrežja (npr. število ali vrsto skupin). Poleg tega je uspešnost na sintetičnih in realnih omrežjih (vsaj) primerljiva z najboljšimi pristopi v literaturi. Dela tako skupaj predstavljajo celovit pristop k odkrivanju skupin vozlišč v velikih realnih omrežjih ter hkrati pomemben doprinos k znanosti.

Predstavljeno delo je v razširjeni obliki objavljeno v [72, 73, 99, 126, 138, 139].

Zadnje, četrto, delo pa se ukvarja s proučevanjem karakterističnih skupin vozlišč v omrežjih odvisnosti med razredi kompleksnega objektno-usmerjenega programja [117], katerih zgradba še ni dokončno pojasnjena. Zaradi splošnosti so poleg izmenjave oznak uporabljeni tudi drugi pristopi odkrivanja skupin v literaturi. Rezultati analize večjega števila programskih omrežij sicer kažejo, da le-ta vsebujejo izrazite skupine vozlišč, ki lahko pojasnijo njihovo zgradbo, poleg tega pa skupine tudi (delno) sovpadajo s paketi pripadajočih programskih knjižnic. Pri tem pa je moč pakete prerazporediti v bolj učinkovito obliko (t.j. modularno ali funkcionalno), iz česar sledijo tudi različni primeri uporabe v programskem inženirstvu.

Predstavljeno delo je v razširjeni obliki objavljeno v [72, 73, 117, 120, 139].

V pričujočem poglavju kratko predstavimo vsebino, teoretično ozadje, hipoteze ter doprinos objavljenih del, začenši z odkrivanjem skupin vozlišč v omrežjih.

Objavljena dela so v disertacijo vložena kot poglavja 3, 4, 5 in 6.[1]

Odkrivanje skupin vozlišč v omrežjih

Naj bo omrežje predstavljeno kot neusmerjen utežen multigraf. Pri tem pa lahko brez izgube za splošnost spodaj, več povezav med parom vozlišč združimo, dočim je utež nove povezave enaka vsoti uteži prvotnih povezav (oziroma številu povezav v neuteženih omrežjih). Posledično dobimo neusmerjen utežen enostaven graf $G\,(V, E)$, kjer je $V = \{v_1, v_2 \dots v_n\}$ množica vozlišč ter E množica povezav, $m = |E|$,

$$E \subseteq \{\{v_i, v_j\} \mid v_i, v_j \in V \wedge i \neq j\}. \tag{2.1}$$

Naj bo $w_{ij} \in \mathbb{R}$ utež na povezavi med vozlišči v_i, v_j ter N_i množica sosedov v_i. Pri tem je $|N_i|$ stopnja vozlišča v_i, kar označimo z d_i. Podobno naj bo w_i stopnja vozlišča, pri kateri upoštevamo tudi uteži na povezavah, $w_i = \sum_{v_j \in N_i} w_{ij}$. Skupine vozlišč v omrežju predstavimo z oznakami vozlišč $g_i \in \{1, 2 \dots n\}$ tako, da si vozlišča v neki skupini delijo enako oznako. Vsakemu vozlišču pripada natanko ena oznaka, saj se omejimo na neprekrivajoče skupine. Poleg tega za skupnosti (module) zahtevamo še, da so vozlišča v skupini povezana (na razdalji največ dva). V nasprotnem primeru nepovezane skupine obravnavamo kot več različnih skupin.

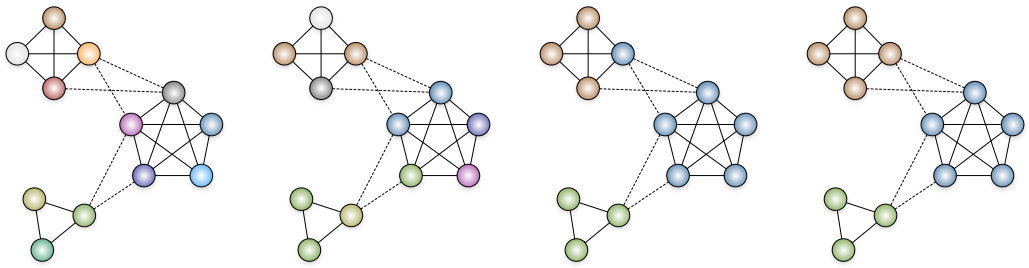
V nadaljevanju najprej predstavimo prvotni pristop odkrivanja skupnosti v velikih omrežjih na osnovi izmenjave oznak med vozlišči [66]. Sledijo izboljšave in razširitve pristopa, ki so bile predstavljene v objavljenih delih [72, 99, 138]. Poudarimo, da kljub temu, da slednje vse izpeljemo iz osnovnega pristopa izmenjave oznak, lahko različne izboljšave in razširitve med seboj neodvisno združujemo.

Objavljena dela so v disertacijo vložena kot poglavja 3, 4 in 5.

Izmenjava oznak med vozlišči

Osnovni pristop izmenjave oznak temelji na naslednji enostavni iteraciji [66]. Na začetku vsako vozlišče v_i razvrstimo v svojo skupino kot $g_i = i$. Nato pa na vsakem

[1] Dela so bila objavljena v revijah, ki jih indeksira SCI, avtor disertacije pa povsod nastopa kot edini prvi avtor. Vsebinski prispevek avtorja je obsegal določitev predpostavk in hipotez; zasnovo, razvoj in implementacijo algoritmov; zasnovo in izvedbo eksperimentov; analizo in interpretacijo rezultatov ter pisanje del.



(a) Začetno stanje (b) 1. korak (c) 2. korak (d) Ravnovesno stanje

Slika 2.1: Prikaz odkrivanja skupin na osnovi izmenjave oznak v omrežju s tremi skupnostmi (glej tekst). Barva vozlišč ustreza oznakam skupin, dočim so povezave, ki sovpadajo z razvrstitvijo v skupine, predstavljene s polno črto.

koraku posodobimo skupine g_i tako, da vozlišča privzamejo oznake, ki si jih deli večina njihovih sosedov (v primeru več oznak, naključno izberemo eno [66, 127]),

$$g_i = \underset{g}{\operatorname{argmax}} \left| \{ v_j \in N_i \mid g_j = g \} \right| . \tag{2.2}$$

Slednje ponavljamo, dokler ni doseženo ravnovesno stanje (t.j. oznake vozlišč se ne spreminjajo [66, 127]). Zaradi tesne povezanosti med vozlišči v skupnostih, oziroma šibke povezanosti sicer, le-ta že po nekaj korakih privzamejo enako oznako skupine (glej sliko 2.1). Ob koncu zato dobljene skupine proglasimo za skupnosti v omrežju.

Časovna zahtevnost koraka iteracije znaša $\mathcal{O}(m)$, saj vsako povezavo obiščemo natanko dvakrat. Na drugi strani pa je število korakov izjemno majhno, saj navadno že po petih korakih več kot 95% vozlišč privzame "pravo" (t.j. končno) oznako [66, 99]. Asimptotična časovna zahtevnost izmenjave oznak je bila sicer ocenjena na $\mathcal{O}(m^{1,23})$ [99], s čimer se lahko primerjajo le redki pristopi v literaturi [54, 140].

Poudarimo, da je pristop izmenjave oznak zgolj (lokalna) optimizacija enačbe (2.2) preko vseh vozlišč v omrežju. Pri tem pa je trivialna globalna rešitev, da vsa vozlišča razvrstimo v eno skupino (enačba (2.2) je tedaj očitno izpolnjena). Vendar se izkaže, da se v omrežjih, ki vsebujejo jasno definirane skupnosti, izmenjava oznak ujame v lokalnem ravnovesnem stanju, v katerem skupine vozlišč ustrezajo skupnostim v omrežju [66, 122]. Dočim slednje navadno smatramo kot slabost lokalnih optimizacijskih algoritmov, pa je le-to ključno pri odkrivanju skupin na osnovi izmenjave oznak (t.j. trivialni globalni rešitvi se izognemo preko lokalnosti pristopa).

Osnovni pristop izmenjave oznak vsa vozlišča obravnava enako, saj imajo vsa enako "moč" širjenja svoje oznake. Slednje opazimo, če v enačbi (2.2) moč množice

nadomestimo z vsoto (pri tem upoštevamo tudi uteži na povezavah),

$$g_i = \underset{g}{\operatorname{argmax}} \sum_{v_j \in N_i} 1 \cdot w_{ij}\delta(g_j, g), \tag{2.3}$$

kjer je δ Kroneckerjev delta. Na proces izmenjave oznak pa lahko sicer vplivamo z uporabo *preferenc vozlišč* f_i [123] (angl. *node preference*),

$$g_i = \underset{g}{\operatorname{argmax}} \sum_{v_j \in N_i} f_j \cdot w_{ij}\delta(g_j, g), \tag{2.4}$$

ki jih nastavimo na poljubno lastnost vozlišč (npr. stopnjo [99, 123]). Na tem pa temeljijo tudi izboljšave izmenjave oznak, ki jih predstavimo v nadaljevanju.

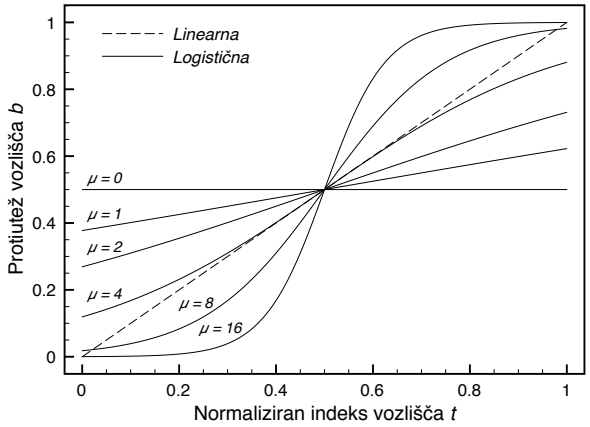
Uravnotežena izmenjava oznak med vozlišči

Prvotno so avtorji izmenjave oznak predpostavili, da se na vsakem koraku iteracije vse oznake skupin vozlišč posodobijo vzporedno (t.j. naenkrat), kar imenujemo *sinhrona izmenjava* [66] (angl. *synchronous propagation*). Vendar se izkaže, da tak pristop ne doseže nujno ravnovesnega stanja v npr. dvodelnih ali zvezdastih omrežjih [66, 141]. Avtorji so zato predlagali zaporedno posodabljanje oznak, pri čimer zaradi splošnosti vozlišča pred vsakim korakom še naključno premešamo. Tako pri posodabljanju oznake vozlišča upoštevamo tudi že posodobljene oznake sosedov, pristop pa imenujemo *asinhorna izmenjava* [66] (angl. *asynchronous propagation*).

Asinhrona izmenjava odpravi zgoraj omenjen problem, vendar pa pristop pri tem postane izjemno nerobusten [122, 128] (t.j. nestabilen). Na primer, v omrežju na sliki 1.3(a) asihrona izmenjava oznak vrne več kot 500 različnih razvrstitev v skupnosti [122], oziroma okrog 200 razvrstitev v tisoč ponovitvah [138, 139], pri čimer ima omrežje zgolj 34 vozlišč. Število razvrstitev je v primeru velikih realnih omrežjih še toliko večje, poleg tega pa so le-te med seboj tudi relativno različne [122, 138].

Slednje je predvsem posledica naključnega vrstnega reda posodabljanja oznak vozlišč na vsakem koraku iteracije [122, 138]. Natančneje, vrstni red na izmenjavo oznak vpliva podobno kot preference vozlišč (glej enačbo (2.4)). Na primer, v kolikor v pristopu oznake vozlišč posodabljamo po padajočih stopnjah, je dobljena razvrstitev v skupine podobna kot, če preference vozlišč nastavimo na njihove stopnje, oznake pa posodabljamo v naključnem vrstnem redu (in obratno). Povedano drugače, vozlišča, katerih oznake posodobimo na začetku koraka, imajo večjo "moč" širjenja svoje oznake kot pa vozlišča, katerih oznake posodobimo proti koncu [138].

Na podlagi zgornje hipoteze kot preference uvedemo *protiuteži vozlišč* b_i (angl. *node balancer*), katerih naloga je uravnotežiti naključni vrstni red posodabljanja



Slika 2.2: Protiuteži (t.j. preference) vozlišč v primeru linearne oziroma logistične uravnotežene izmenjave oznak pri različnih vrednostih parametra μ (glej tekst).

oznak pri asinhroni izmenjavi oznak. Izmenjava oznak v enačbi (2.3) tako postane

$$g_i = \underset{g}{\operatorname{argmax}} \sum_{v_j \in N_i} b_j \cdot w_{ij} \delta(g_j, g), \tag{2.5}$$

kar imenujemo *uravnotežena izmenjava* [138] (angl. *balanced propagation*) oznak.

Naj bo $t_i \in (0, 1]$ normaliziran indeks vozlišča v_i v nekem naključnem vrstnem redu. Protiuteži vozlišč lahko definiramo že s preprosto linearno funkcijo kot $b_i = t_i$ (glej sliko 2.2), dobljen pristop pa je občutno bolj robusten kot prvotna izmenjava oznak. V omrežju na sliki 1.3(a) pristop v tisoč ponovitvah vrne zgolj še približno 20 razvrstitev v skupnosti, število različnih razvrstitev pa je tudi v drugih realnih omrežij za velikostni red manjše kot v primeru osnovnega pristopa [138, 139]. Uspešnost lahko še nekoliko izboljšamo z uporabo logistične funkcije,

$$b_i = \frac{1}{1 + e^{-\mu(t_i - \lambda)}}, \tag{2.6}$$

kjer sta $\lambda, \mu \in \mathbb{R}$ parametra. Zaradi enostavnosti privzamemo $\lambda = 0{,}5$, dočim parameter μ omogoča dodaten nadzor nad pristopom. Pri vrednosti $\mu = 0$ dobimo osnovno izmenjavo oznak v enačbi (2.3) (do konstante natančno), s povečevanjem μ pa naraščajo razlike med protiutežmi vozlišč (glej sliko 2.2). Tako pristop postaja vedno bolj robusten, s čimer se izboljšuje tudi natančnost, vendar pa se pri tem povečuje časovna zahtevnost [138, 139].

Uravnotežena izmenjava oznak predstavlja pomemben doprinos k odkrivanju karakterističnih skupin vozlišč, saj je osnovni pristop zaradi slabše robustnosti manj

primeren za uporabo v velikih realnih omrežjih. Predstavljena je v delu z naslovom "Odkrivanje skupin vozlišč z uravnoteženo izmenjavo oznak" [138], ki je v pričujočo disertacijo vloženo kot poglavje 3, razširjeno razpravo pa najdemo v poglavju 7.

Predstavljeno delo je v razširjeni obliki objavljeno v [138, 139].

Napredna izmenjava oznak med vozlišči

Natančnost osnovnega pristopa izmenjave oznak je v velikih realnih omrežjih navadno slabša kot pa v primeru najboljših pristopov v literaturi [99, 126]. Eden od razlogov za to je, da pristop v omrežjih z manj jasno zgradbo pogosto večino vozlišč razvrsti v eno samo skupino [123, 131]. Podobno kot v prejšnjem razdelku lahko slednje pripišemo naključnemu vrstnemu redu obdelave in posodabljanja oznak vozlišč, problem pa je posledično nekoliko omiljen pri uravnoteženi izmenjavi oznak.

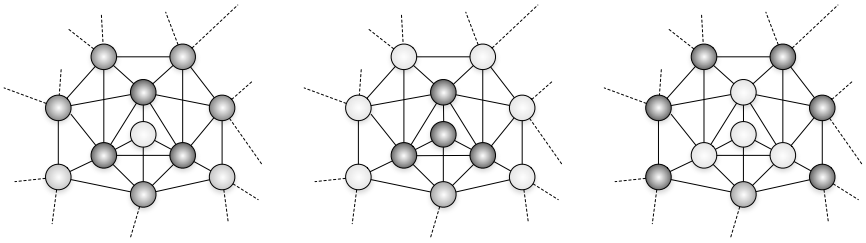
Zgornjemu se lahko izognemo z uporabo *slabljenja oznak* [123] (angl. *label attenuation*). Pri tem kot preference vozlišč uvedemo vrednosti $s_i \in [0, 1]$ (glej enačbi (2.9) in (2.10)), ki predstavljajo razdaljo med vozliščem v_i ter izvorom oznake vozlišča g_i. Tako na začetku $s_i = 1$, ob vsaki spremembi g_i pa vrednosti s_i posodobimo kot

$$s_i = \max_{v_j \in N_i} (s_j \cdot \delta(g_j, g_i)) - \omega, \tag{2.7}$$

kjer je $\omega \in (0, 1)$ parameter. Opazimo, da se pri npr. $\omega = 0{,}5$ oznake skupin očitno lahko širijo največ dva koraka od svojega izvora (glej enačbi (2.9) in (2.10)), s čimer preprečimo, da bi večina vozlišč v omrežju privzela enako oznako. Vendar se izkaže, da izbira vrednosti ω v realnih omrežjih ni enostavna [99, 123].

Manjše vrednosti ω v splošnem ne preprečijo omenjenega problema, na drugi strani pa velike vrednosti lahko onemogočijo naravno izmenjavo oznak med vozlišči. Avtorji slabljenja oznak so zato predlagali, da ω postopoma zmanjšujemo od 0,5 do nič [123], vendar pa, podobno kot prej, izbira koraka zmanjševanja ni enostavna [99]. Poudarimo, da je glavni namen slabljenja oznak preprečitev pojava velike skupine, pri tem pa ne želimo vplivati na sam proces izmenjave oznak. Na podlagi slednjega tako predlagamo različne strategije *dinamičnega slabljenja* [99] (angl. *dynamic attenuation*) oznak, kjer ω povečamo zgolj kadar neka skupina naglo zaseda večji del omrežja. Slednje lahko odkrijemo razmeroma enostavno z npr. analizo števila vozlišč, ki so na zadnjem koraku iteracije privzela novo oznako [99, 126].

Dinamično slabljenje oznak učinkovito prepreči pojav velike skupine, dočim pa lahko natančnost pristopa v realnih omrežjih navadno še izboljšamo. Na primer, v omrežju na sliki 1.3(a) je izjemno uspešen pristop, kjer kot preference uporabimo



(a) Stopnje vozlišč (b) Zadržana izmenjava (c) Nasilna izmenjava

Slika 2.3: Shematski prikaz različnih pristopov odkrivanja skupin vozlišč na osnovi izmenjave oznak (glej tekst). Preference vozlišč so predstavljene z intenziteto barve, dočim so povezave, ki sovpadajo s prikazano skupino, predstavljene s polno črto.

stopnje vozlišč [99, 123] (glej sliko 2.3(a)). Pri tem povečamo "moč" širjenja oznak treh vozlišč z izjemno visoko stopnjo, ki vsa ležijo v *jedrih* (angl. *core*) dveh skupnosti v omrežju, kar omogoči naravno rast skupin od znotraj navzven. Slednje seveda ni mogoče v splošnem, saj v naprej ne poznamo dejanske zgradbe skupin. Kljub temu pa lahko tekom izmenjave oznak le-te ocenimo s trenutnimi skupinami v omrežju.

Na podlagi zgornje hipoteze kot preference vozlišč uvedemo še vrednosti $p_i \in [0, 1]$ (glej enačbi (2.9) in (2.10)), ki naj bodo blizu ena za vozlišča v jedru neke skupine, ter blizu nič za vozlišča na *robovih* (angl. *border*). Vrednosti p_i lahko ocenimo z uporabo naključnih sprehodov, omejenih na posamezne skupine vozlišč. Tako na začetku $p_i = 1/n$, ob vsaki spremembi g_i pa vrednosti p_i posodobimo kot

$$p_i = \sum_{v_j \in N_i} \frac{p_j \cdot w_{ij}\delta(g_j, g_i)}{\sum_{v_k \in N_j} w_{jk}\delta(g_k, g_i)}. \tag{2.8}$$

Omenimo, da je enačba (2.8) podobna meri *PageRank* [142], s katero je v svojem iskalniku zaslovel "Google", ter tudi eni od mer središčnosti [143]. Poudarimo še, da je v splošnem rešitev enačbe (2.8) kar $p_i = \sum_{v_j \in N_i} w_{ij}\delta(g_j, g_i)$ [2], vendar zaradi naključnega posodabljanja vozlišč pri asinhroni izmenjavi oznak slednje ne velja.

V kolikor povečamo "moč" širjenja oznak vozlišč v jedrih skupin, ter poleg tega uporabimo (dinamično) slabljenje, izmenjava oznak v enačbi (2.3) postane

$$g_i = \operatorname*{argmax}_g \sum_{v_j \in N_i} p_j s_j \cdot w_{ij}\delta(g_j, g), \tag{2.9}$$

pristop pa imenujemo *zadržana izmenjava* [99] (angl. *defensive propagation*) oznak. Podobno lahko povečamo "moč" širjenja oznak vozlišč na robovih skupin, pri čimer

dobimo (v tem primeru imenovalec v enačbi (2.8) nadomestimo z w_j)

$$g_i = \underset{g}{\operatorname{argmax}} \sum_{v_j \in N_i} (1 - p_j) s_j \cdot w_{ij} \delta(g_j, g), \tag{2.10}$$

kar imenujemo *napadalna izmenjava* [99] (angl. *offensive propagation*) oznak. Shematski prikaz različnih pristopov izmenjave oznak je podan na sliki 2.3.

Če je zadržana izmenjava oznak bolj uspešna v primeru gostih socialnih, informacijskih ter bioloških omrežij z manj jasnimi skupinami vozlišč, pa je napadalna izmenjava oznak bolj natančna v primeru redkejših tehnoloških in nekaterih drugih omrežij [126]. Sicer zadržana izmenjava oznak odkrije skupine vozlišč tudi, ko so le-te zgolj šibko izražene v zgradbi omrežja, vendar natančnost pristopa pri tem ni najboljša [99]. Na drugi strani pa napadalna izmenjava oznak navadno razkrije skupine v omrežju do vozlišča natančno, a zgolj v primeru, ko so razmeroma jasno izražene [99]. Povedano drugače, zadržana izmenjava ima visok *priklic* (angl. *recall*), napadalna izmenjava pa doseže visoko *natančnost* (angl. *precision*).

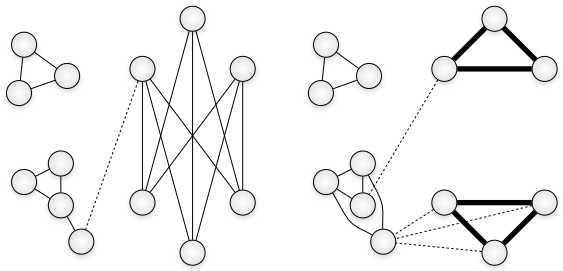
Očitno bi želeli prednosti obeh pristopov, kar dosežemo tako, da grobo začetno razvrstitev v skupine pridobimo z zadržano izmenjavo, skupine pa nato še izpopolnimo z napadalno izmenjavo. Slednje imenujemo *napredna izmenjava* [99] (angl. *diffusion propagation*) oznak, za uporabo v velikih realnih omrežjih pa predlagamo tudi hierarhično preiskovanje [73, 99, 126] (glej poglavje 7). Opisan pristop je še bolj učinkovit kot osnovna izmenjava oznak, pri čimer je bila asimptotična časovna zahtevnost ocenjena na $\mathcal{O}(m^{1,19})$ [99]. Poleg tega je natančnost na sintetičnih in realnih omrežjih (vsaj) primerljiva z najboljšimi pristopi v literaturi [99, 126].

Napredna izmenjava oznak predstavlja pomemben doprinos k odkrivanju karakterističnih skupin vozlišč, saj je osnovni pristop zaradi slabše natančnosti manj primeren za uporabo v velikih realnih omrežjih. Predstavljena je v delu z naslovom "Odkrivanje skupin vozlišč z napredno izmenjavo oznak" [99], ki je v pričujočo disertacijo vloženo kot poglavje 4, razširjeno razpravo dela pa najdemo v poglavju 7.

Predstavljeno delo je v razširjeni obliki objavljeno v [73, 99, 126].

Posplošena izmenjava oznak med vozlišči

Osnovni pristop izmenjave oznak lahko razkrije zgolj skupnosti v omrežju, saj je omejen na odkrivanje tesno povezanih skupin vozlišč [72]. Povezanost sicer smatramo za eno od ključnih lastnosti skupnosti [54], na drugi strani pa so dobro izraženi moduli vozlišč nepovezani. Poleg tega moduli predstavljajo med seboj odvisne skupine vozlišč [72, 139], pri čimer so skupnosti po definiciji neodvisne. Moduli pa se od



(a) (b)

Slika 2.4: (a) Primer omrežja z dvema skupnostma in dvema moduloma. (b) Omrežje z enako množico vozlišč, ki so povezana v kolikor v prvotnem omrežju obstaja skupno sosednje vozlišče. Povezave, ki sovpadajo z razvrstitvijo v skupine, so predstavljene s polno črto, dočim intenziteta ustreza utežem na povezavah.

skupnosti značilno ločijo še v tem, da so verjetno v ozadju prisotni različni procesi, ki povzročijo pojav posameznih vrst skupin v realnih omrežjih [73, 144]. Tako kljub temu, da skupnosti lahko obravnavamo kot poseben primer modulov [60, 75], v pričujoči doktorski disertaciji dosledno ločimo med obema vrstama skupin vozlišč.

Posplošitev izmenjave oznak na odkrivanje modulov pa je sicer razmeroma enostavna. Namesto, da oznake izmenjujemo med sosednjimi vozlišči, si oznake preko skupnih sosedov izmenjujejo vozlišča na razdalji (največ) dva. Tako odkrivanje modulov dejansko prevedemo na odkrivanje skupnosti, saj je slednje ekvivalentno izmenjavi oznak v omrežju z enako množico vozlišč, ki so povezana, v kolikor obstaja vsaj eno skupno vozlišče v prvotnem omrežju (pri tem upoštevamo tudi uteži na povezavah). V kolikor so vozlišča v modulih res podobno povezana z ostalim omrežjem, jih bomo z opisanim pristopom zanesljivo odkrili (glej sliko 2.4).

Predpostavimo, da za vsako skupino vozlišč g poznamo parameter $\nu_g \in [0, 1]$, ki naj bo blizu ena v primeru skupnosti, ter blizu nič v primeru modulov. Tedaj lahko izmenjavo oznak v enačbi (2.3) razširimo na odkrivanje splošnih skupin vozlišč kot

$$g_i = \underset{g}{\operatorname{argmax}} \left(\nu_g \cdot \sum_{v_j \in N_i} w_{ij} \delta(g_j, g) + (1 - \nu_g) \cdot \sum_{v_j \in N_i} w_{ij}/w_j \sum_{v_k \in N_j} w_{jk} \delta(g_k, g) \right), \tag{2.11}$$

pristop pa imenujemo *posplošena izmenjava* [72] (angl. *general propagation*) oznak. Pri $\nu_g = 1$ dobimo levo stran enačbe (2.11), ki ustreza osnovni izmenjavi oznak za odkrivanje skupnosti, pri $\nu_g = 0$ pa dobimo desno stran, ki je namenjena odkrivanju modulov. (Utež w_j v imenovalcu poskrbi, da sta obe strani sorazmerni z w_i.) Pri

$\nu_g = 0{,}5$ predlagan pristop razkrije tako skupnosti kot module v omrežjih, za to pa ne zahteva nikakršnega predhodnega znanja (npr. število ali vrsto skupin).

Zaradi obravnave vozlišč na razdalji dva od proučevanega, je časovna zahtevnost posameznega koraka posplošene izmenjave oznak enaka $\mathcal{O}(m \cdot \overline{d})$, kjer je $\overline{d}$ povprečna stopnja vozlišč v omrežju. Podobno kot prej je število korakov iteracije navadno razmeroma majhno [72, 139]. Tako lahko na podlagi zgornje analogije med skupnostmi in moduli ocenimo, da je pristop omogoča analizo omrežij z več milijoni vozlišč ter povezav, kar je sicer možno le z redkimi pristopi v literaturi [73].

Kljub temu, da je uspešnost posplošene izmenjave oznak pri $\nu_g = 0{,}5$ že primerljiva z drugimi pristopi [73], je natančnost moč še občutno izboljšati z uporabo primernih vrednosti parametrov ν_g. Le-te lahko podobno kot zgoraj tekom izmenjave oznak ocenimo na podlagi analize lastnosti skupin [72, 139]. Še posebej pa so uspešni pristopi, pri katerih na podlagi analize lastnosti vozlišč omrežja presodimo, kje v omrežju je moč pričakovati določeno vrsto skupin [73, 139]. Tako se izkaže, da npr. module v realnih omrežjih pogosto najdemo v delih z nizko nakopičenostjo [145], prav nasprotno pa velja za skupnosti [73, 144]. Podobno so stopnje povezanih vozlišč v primeru skupnosti (modulov) navadno v močni (nasprotni) korelaciji [73, 146].

V kolikor pri odkrivanju skupin s posplošeno izmenjavo oznak uporabimo tudi izboljšave predstavljene v prejšnih razdelkih [99, 138], je uspešnost na sintetičnih in realnih omrežjih (vsaj) primerljiva z najboljšimi pristopi v literaturi [73, 139]. Pri tem pa poudarimo, da praktično vsi drugi znani pristopi zahtevajo število skupin vnaprej, kar v velikih realnih omrežjih danes ni moč natančno določiti [83].

Posplošena izmenjava oznak predstavlja pomemben doprinos k odkrivanju karakterističnih skupin vozlišč, saj je osnovni pristop zaradi omejitve na skupnosti manj primeren za uporabo v splošnih realnih omrežjih. Predstavljena je v delu z naslovom "Odkrivanje skupin vozlišč s posplošeno izmenjavo oznak" [72], ki je v disertacijo vloženo kot poglavje 5, razširjeno razpravo dela pa najdemo v poglavju 7.

Predstavljeno delo je v razširjeni obliki objavljeno v [72, 73, 139].

Skupine vozlišč v programskih omrežjih

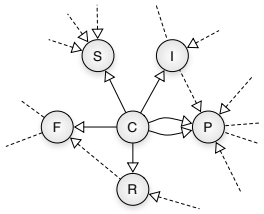
Doktorska disertacija se poleg razvoja pristopov za odkrivanje karakterističnih skupin vozlišč osredotoča tudi na proučevanjem le-teh v programskih omrežjih. V pričujočem razdelku tako kratko predstavimo vsebino, teoretično ozadje, zasledovane hipoteze ter znanstveni doprinos dela z naslovom "Analiza in uporaba skupin v programskih omrežjih" [117]. Le-to je v disertacijo vloženo kot poglavje 6, razširjeno

```
class C extends S implements I {

  F field;

  public C(P parameter) {
    ...
  }

  public R function(P parameter) {
    ...
    return R;
  }
}
```



(a) (b)

Slika 2.5: (a) Primer razreda v programskem jeziku "Java". (b) Pripadajoč del omrežja programskih odvisnosti, kjer oznake vozlišč ponazarjajo imena razredov. Povezave, ki ustrezajo odvisnostim razreda "C", so predstavljene s polno črto.

razpravo pa najdemo v poglavju 7. Poudarimo, da se omenjeno delo sicer ukvarja zgolj s proučevanjem skupnosti v programskih omrežjih, pri čimer pa je analiza in razprava o modulih objavljena v [72, 73, 120, 139] (glej npr. poglavje 5).

Splošne lastnosti programskih omrežij so bile v literaturi že dodobra raziskane, na drugi strani pa je le malo znanega o različnih skupinah vozlišč [72, 117, 118, 147]. Modularna zgradba, ki nakazuje na skupnosti v omrežju, je bila sicer proučevana s stališča različnih dinamičnih procesov ter praktične uporabe programja [108, 118], o slednjem pa so razpravljali še nekateri drugi avtorji [100, 103]. Pri tem omenimo tudi, da na obstoj skupnosti v programskih omrežjih nakazujejo številne teorije [110, 121]. Na primer, *šibka-sklopljenost* in *močna-kohezija* [148] (angl. *minimum-coupling, maximum-cohesion*) različnih konstruktov objektno-usmerjenega programja bi se očitno morala jasno odražati v modularni zgradbi omrežij [121, 149].

V objavljenem delu privzamemo omrežja odvisnosti med razredi [103, 117] različnih javanskih knjižnic ter tudi samega programskega jezika "Java". Pri tem so *odvisnosti* (angl. *dependence*) različnih vrst (glej sliko 2.5), v splošnem pa usmerjena povezava med vozliščema, ki predstavljata razreda C_i, C_j, pomeni, da je pri implementaciji C_i potreben C_j (t.j. C_i je odvisen od C_j). Delo sicer temelji na hipotezi, da upoštevajoč zgornje teorije skupnosti vozlišč v omrežjih odvisnosti med razredi neke knjižnice verjetno sovpadajo s pripadajočimi programskimi paketi [117, 121].

Analiza pokaže, da programska omrežja res vsebujejo skupnosti z visoko modularnostjo Q [69], ki so razmeroma jasno definirane v strukturi omrežja [117]. Pričakovano to ne velja za programski jezik sam, saj je zaradi splošne namembnosti razredov pripadajoče omrežje veliko gostejše kot primerljiva druga omrežja. Sicer

pa se izkaže, da v ostalih analiziranih omrežij odkrite skupnosti deloma ustrezajo programskim paketom, vendar ujemanje ni tako izrazito, kot bi pričakovali [72, 117].

Številni paketi so namreč nepovezane skupine vozlišč v omrežjih, ki imajo posledično izjemno nizko modularnost Q [73, 117]. Čeprav bi slednje lahko pripisali neustrezni predstavitvi programja ali dejstvu, da so nekatere od proučevanih knjižnic še v razvoju, pa velja, da je tudi porazdelitev velikosti skupin v primeru programskih paketov značilno različna od skupnosti v omrežjih [117]. Poudarimo, da je velikost skupin neodvisna od same predstavitve programja (t.j. povezav), zatorej zgolj s skupnostmi očitno ni moč dokončno pojasniti zgradbe programskih omrežij.

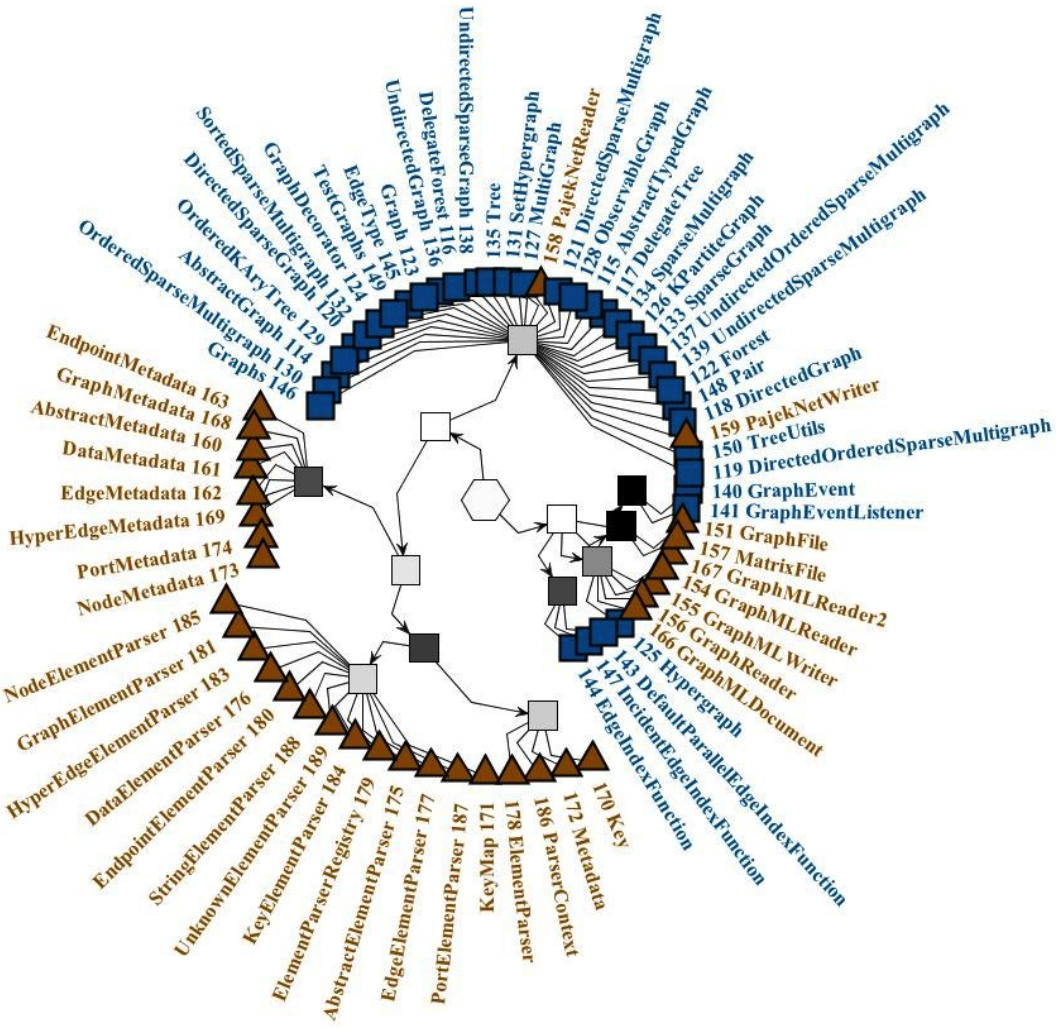
Ta pa gotovo vsebujejo tudi module vozlišč [72, 73]. V programskih knjižnicah namreč pogosto najdemo razrede z enako funkcionalnostjo, ki predstavljajo zgolj različne rešitve nekega problema. Omenjeni razredi pri implementaciji navadno uporabljajo enak nabor drugih razredov, kar se odraža kot podobno povezane skupine vozlišč v omrežjih (t.j. moduli). Tako ni presenetljivo, da hierarhije splošnih skupin vozlišč veliko bolje povzamejo zgradbo programskih omrežij [72, 139] ter hierarhijo paketov pripadajočih programskih knjižnic [73, 120], kot pa le skupnosti (glej poglavje 7). Natančneje, zgornji nivoji hierarhije praviloma ustrezajo (večjim) skupnostim, dočim module navadno najdemo na najnižjem nivoju [73] (glej sliko 2.6).

V praksi lahko karakteristične skupine vozlišč v prvi vrsti uporabimo za povzemanje programja, saj le-te poleg zgradbe paketov pojasnjujejo tudi odvisnosti med različnimi funkcionalnimi komponentami kompleksnega programja [103, 117]. Na drugi strani pa lahko pristope za odkrivanje skupin vozlišč uporabimo za reorganizacijo programskih paketov v bolj učinkovito obliko [117, 120], pri čimer upoštevamo tudi odvisnosti, ki so jih razvijalci morda spregledali. Pakete tako preoblikujemo v bolj modularno (npr. vizualizacija) ali funkcionalno (npr. znanstveno računanje) obliko, kar ustreza skupnostim oziroma modulom v programskim omrežjih.

Poleg omenjenega lahko hierarhije skupine vozlišč uporabimo tudi za napovedovanje odvisnosti razredov nekega programskega paketa oziroma določanje najprimernejšega paketa za še neuvrščene razrede [120]. Kot primer povejmo, da lahko za večino razredov v proučevanih programskih knjižnicah napovemo celotno hierarhijo paketov v približno 80% primerih. (Za druge primere uporabe glej [120].)

Zgornja analiza predstavlja pomemben doprinos k razumevanju karakterističnih skupin vozlišč v kompleksnih programskih omrežjih, hkrati pa omenjeni primeri uporabe odpirajo številne možnosti za praktične aplikacije programskega inženiringa.

Predstavljeno delo je v razširjeni obliki objavljeno v [72, 73, 117, 120, 139].



Slika 2.6: Del hierarhije skupin vozlišč v omrežju odvisnosti programske knjižnice "JUNG" [27] (glej sliko 1.6(a)). Barva vozlišč ponazarja visokonivojske pakete knjižnice, dočim oznake ustrezajo imenom razredov. Intenziteta notranjih vozlišč hierarhije je sorazmerna z gostoto pripadajočega dela omrežja (za več glej [120]).

3

Odkrivanje skupin vozlišč z uravnoteženo izmenjavo oznak

V pričujočem poglavju je vloženo delo z naslovom "Odkrivanje skupin vozlišč z uravnoteženo izmenjavo oznak" [138] (angl. "Robust network community detection using balanced propagation"). Delo predstavi različne metode in tehnike odkrivanja karakterističnih skupin vozlišč v omrežjih na osnovi izmenjave oznak, ki temeljijo na povezavi med vrstnim redom izmenjave in širjenjem oznak po omrežju. Le-te močno izboljšajo robustnost osnovnega pristopa ter pri tem ohranijo njegovo algoritmično enostavnost. Predlagane izboljšave pristopa so preizkušene na številnih sintetičnih in realnih omrežjih z znano zgradbo ter primerjane z drugimi pristopi v literaturi.

Predstavljeno delo je v razširjeni obliki objavljeno v [138, 139].[2]

[2]Privzeta programska koda ter omrežja so dostopna na `http://lovro.lpt.fri.uni-lj.si/`.

Eur. Phys. J. B **81**, 353–362 (2011)
DOI: 10.1140/epjb/e2011-10979-2

THE EUROPEAN PHYSICAL JOURNAL B

Regular Article

Robust network community detection using balanced propagation

L. Šubelj[a] and M. Bajec

University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia

Received 15 December 2010 / Received in final form 24 February 2011
Published online 4 May 2011 – © EDP Sciences, Società Italiana di Fisica, Springer-Verlag 2011

Abstract. Label propagation has proven to be an extremely fast method for detecting communities in large complex networks. Furthermore, due to its simplicity, it is also currently one of the most commonly adopted algorithms in the literature. Despite various subsequent advances, an important issue of the algorithm has not yet been properly addressed. Random (node) update orders within the algorithm severely hamper its robustness, and consequently also the stability of the identified community structure. We note that an update order can be seen as increasing propagation preferences from certain nodes, and propose a balanced propagation that counteracts for the introduced randomness by utilizing node balancers. We have evaluated the proposed approach on synthetic networks with planted partition, and on several real-world networks with community structure. The results confirm that balanced propagation is significantly more robust than label propagation, when the performance of community detection is even improved. Thus, balanced propagation retains high scalability and algorithmic simplicity of label propagation, but improves on its stability and performance.

1 Introduction

Complex real-world networks can comprise local structural modules (i.e., *communities* [1]) that are groups of nodes densely connected within and only loosely connected with the rest of the network. Communities may play important roles in different real-world systems – they can be related to functional modules in biochemical networks [2] or individuals with common interests in social networks [1]. Moreover, community structure also has a strong impact on dynamic processes taking place on such networks [3] and can thus provide an important insight into not only structural organization but also functional behavior of various real-world systems.

As a consequence, analysis of network community structure has been the focus of recent endeavor in different fields of science. There has also been a substantial number of community detection algorithms proposed in the literature over the last years [2,4–13] (for a comprehensive survey see [14]). Nevertheless, due to scalability issues, only a small minority of these algorithms can be applied to large real-world networks with several millions, billions of nodes, edges respectively.

A notable step towards this end was made by Raghavan et al. [7], who employed a simple *label propagation* to reveal significant communities in large real-world networks. Communities are identified by propagating (community) labels among nodes, thus, each node is assigned the label shared by most of its neighbors. Due to very fast structural inference of label propagation, densely connected sets of nodes form a consensus on some particular label after only a few iterations [7,13]. The algorithm thus exhibits near linear complexity, which makes it applicable on networks with millions of nodes in a matter of minutes [13]. The basic algorithm was further analyzed and refined by various authors [13,15–26], when, due to its simplicity, label propagation is also currently one of the most commonly adopted algorithms in the literature.

Despite the above efforts, an important issue of label propagation has not yet been properly addressed. To overcome convergence problems in some types of networks, Raghavan et al. [7] have proposed propagating labels among nodes (i.e., updating nodes' labels) in a random order. Although this updating strategy solves the aforementioned problem, introduction of randomness severely hampers the robustness of the algorithm, and consequently also the stability of the identified community structure. It has been noted that the algorithm reveals a large number of distinct community structures even in smaller networks [7,13,16,19], when these structures are also relatively different among themselves [13,16]. Still, the robustness of the algorithm can also be related to the significance of community structure in a network [13].

We argue that updating the nodes in some particular order can be seen as placing higher *propagation preference* [18] to the nodes that are updated at the beginning, and lower propagation preference to the nodes that are updated towards the end (and updating the nodes in a random order). The order of node updates thus governs the dynamics of the algorithm in a similar manner as

[a] e-mail: lovro.subelj@fri.uni-lj.si

(corresponding) node propagation preferences. This observation allows us to stabilize the label propagation algorithm by utilizing node preferences to counteract (i.e., balance) the randomness introduced by random node updates. The resulting algorithm is denoted *balanced propagation* and differs from label propagation merely in the introduction of *node balancers*.

We have evaluated the proposed algorithm on synthetic benchmark networks with planted partition, and on various real-world networks with community structure. The results confirm that balanced propagation is significantly more robust than simple label propagation, when the performance of community detection is even improved (in most cases). We also apply the algorithm to an entire European road network, which is not considered to reveal clear community structure. Nevertheless, the algorithm accurately identifies communities that correspond to different (geographical) regions of Europe, without any serious issues with stability.

The rest of the article is organized as follows. In Section 2 we formally present label propagation, and review issues and advances relevant for this research. Section 3 introduces balanced propagation and discusses the main rationale behind it. Empirical evaluation with discussion is given in Section 4 and conclusion in Section 5.

2 Label propagation

Let the network be represented by a simple undirected graph $G(N, E)$, where N is the set of nodes and E is the set of edges[1]. Furthermore, let w_{nm} be the weight of the edge incident to nodes $n, m \in N$. Moreover, let c_n denote the community (label) of node $n \in N$ and let $\mathcal{N}(n)$ denote the set of its neighbors.

Basic *label propagation algorithm* (*LPA*) [7] reveals network communities by exploiting the following simple procedure. At first, each node $n \in N$ is labeled with an unique label, $c_n = l_n$. Then, at each iteration, each node adopts the label shared by most of its neighbors (considering also edge weights). Hence,

$$c_n = \underset{l}{\operatorname{argmax}} \sum_{m \in \mathcal{N}^l(n)} w_{nm}, \tag{1}$$

where $\mathcal{N}^l(n)$ is the set of neighbors of $n \in N$ that share label l (ties are broken uniformly at random). Due to the existence of many intra-community edges, relative to the number of inter-community edges, densely connected sets of nodes form a consensus on some particular label after a few iterations. Thus, when the algorithm converges (i.e., equilibrium is reached), disconnected sets of nodes sharing the same label are classified into the same community. Due to extremely fast structural inference of label propagation, the algorithm exhibits near linear time complexity [7,13] (in the number of edges of the network) and can easily scale to networks with millions, or even billions, of nodes and edges [13,25].

Leung et al. [18] have first noticed that label propagation can be substantially improved by increasing *propagation preference* (i.e., propagation strength) from certain nodes. The updating rule of the algorithm (i.e., Eq. (1)) is thus rewritten into

$$c_n = \underset{l}{\operatorname{argmax}} \sum_{m \in \mathcal{N}^l(n)} p_m w_{nm}, \tag{2}$$

where p_n is the preference of node $n \in N$. Adequate node preferences can alter the dynamics of label propagation, in order to guide the algorithm towards a more significant community structure [13]. For the analysis and comparison of different node preference strategies, and corresponding algorithms, see [13,18,25].

Next, we also discuss two main issues of label propagation and its advances. First, consider a bipartite network with two sets of nodes, denoted red and green nodes. Further assume that, at some point of the algorithm, all red nodes share label l_r, and all green nodes share label l_g. Due to bipartite structure, at the next iteration, all red nodes will adopt label l_g, and all green nodes will adopt label l_r. Moreover, at next iteration, all nodes will recover their initial labels, failing the algorithm to converge. It should be noted that such oscillations of labels are not limited to bipartite networks, but occur in various real-world networks that are commonly analyzed in the literature.

To ensure convergence, Raghavan et al. [7] have proposed *asynchronous* updating of nodes. Hence, nodes are no longer updated all together, but sequentially, in some random order. Thus, when node's label is updated, possibly already updated labels of its neighbors are considered (in contrast to *synchronous* updating, where only labels from the previous iteration are considered). Although asynchronous updating eliminates aforementioned oscillations of labels, introduction of randomness severely disturbs the robustness of the algorithm, and consequently also the stability of the identified community structure. The stability of label propagation presents a severe issue for the algorithm, however, it has not yet been properly addressed in the past (to the best of our knowledge).

Second, consider a network with *overlapping communities* [2] and let $n \in N$ be a node that has equally strong connections with two or more such communities. As ties are broken uniformly at random (see Eq. (1)), label c_n would then, in general, constantly change. Furthermore, when many of such nodes exist, the algorithm would obviously never converge. Again, the issue is not limited to networks with overlapping communities.

Two possible solutions have been proposed in the literature. Leung et al. [18] suggested including label c_n into the maximal label consideration (besides merely neighbors' labels), when Raghavan et al. [7] proposed a slightly modified approach. When there are multiple maximal labels (among neighbors' labels), and one of them equals the concerned label c_n, the node retains its label. In contrast to the former, the latter approach considers concerned label only when there indeed exist multiple maximal labels.

[1] In directed networks, each edge is treated as undirected, and in multi-networks, multiple edges among nodes are encoded into edge weights.

Although both presented approaches work well for simple label propagation (i.e., Eq. (1)), this is not necessarily the case for different advances of the algorithm (e.g., Eq. (2)). Still, for the analysis in this article we adopt the approach proposed by Raghavan et al. [7].

In the proceeding section we revisit both issues discussed above, and propose solutions to overcome them.

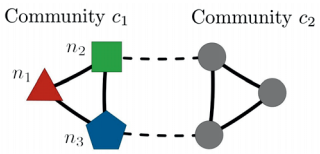


Fig. 1. (Color online) Toy example network with two strong communities (inter-community edges are shown with dashed links). Node colors (shapes) indicate their community labels.

3 Balanced propagation

Label propagation with asynchronous updating accesses the nodes in a random order. In particular, nodes are (re)shuffled before each iteration, in order to address convergence issues in some networks. However, as already discussed in Section 2, this incorporation of randomness severely hampers the robustness of the algorithm.

The issue can be addressed in an ad hoc fashion by simply accessing the nodes in some predefined (deterministic) order. This would clearly stabilize the algorithm, and possibly also perform well on real-world networks. We have conducted several experiments with different update orders, based on various node statistics (i.e., degree and eigenvector centrality [27,28], clustering coefficient [29]). Exact results are omitted, however, they indicate that, although none of these deterministic orders performs well in all networks, best order commonly corresponds to node preference strategy that also performs well. For instance, when ordering the nodes based on their degrees (decreasingly) gives good results, setting propagation preferences to the degrees of the nodes (and updating them in a random order) also performs well (and vice-versa).

Based on the above discussion we pose a hypothesis that the order of node updates within asynchronous label propagation governs algorithm's dynamics in a similar manner as the corresponding node propagation preferences. Intuitively, nodes that are updated at the end of some iteration cannot efficiently propagate their final labels onward, as (most of) their neighbors have already been updated. On the other hand, a node that is considered first can possibly propagate its label to all of its neighbors, and thus form a community. Hence, nodes updated at the beginning exhibit higher propagation strength than those that are considered towards the end.

We further study the proposed hypothesis on a toy example network in Figure 1. The network consists of two communities, namely c_1 and c_2, that are defined in a *strong sense* [30] (i.e., each node has more intra-community than inter-community edges). Further assume that, at some point of the algorithm, nodes in c_1, namely n_1, n_2 and n_3, are labeled with unique (community) labels, when all nodes in c_2 have already been classified to their right community (see Fig. 1).

We first analyze how different orders of node updates affect the final outcome of the algorithm. When node n_1 is considered first, it will adopt the label of either n_2 or n_3. Due to symmetry, we can assume that it adopts the label of node n_2. No matter which of the nodes n_2 or n_3 is updated next, at the end of this iteration, all nodes in community c_1 will be labeled with the same label (that initially belongs to node n_2). The outcome thus corresponds to the natural community structure of the network.

On the other hand, when node n_1 is updated last, the results can differ. Again, we can assume that node n_2 is considered before node n_3. If node n_2 adopts the label of either n_1 or n_3, the algorithm proceeds similar as above. However, node n_2 can also adopt the label of the second community c_2 (with some probability). In that case, it is straightforward to see that nodes n_1 and n_3 will also adopt the same label, thus, at the end, all nodes in the network will be classified to the same community c_2.

To summarize, if we first consider the core of community c_1 (i.e., node n_1), the label propagation will inevitably lead to the natural community structure of the network. However, if we access the border of community c_1 first (i.e., nodes n_2 and n_3), the algorithm could potentially classify all nodes into the same community (mainly due to the fact that community c_2 is already established). The example shows that even in such simple network, label propagation is extremely sensitive to the order of node updates.

Similar behavior as above can be observed, when we set higher propagation preference to either core or border of community c_1 (and update the nodes in a random order). When core node n_1 has the highest preference in the network, nodes n_2 and n_3 would obviously adopt the label of node n_1. This would unavoidably lead to identification of the natural community structure, no matter the order of updates. However, when higher preference is given to border nodes n_2 and n_3 (i.e., lowest preference is given to node n_1), outcome of the algorithm can again correspond to the trivial community structure, where all nodes are classified into the same community (depends on the preference of other nodes and the order of updates). We thus conclude that, at least for this toy example, order of node updates can be seen as placing higher propagation preference to the nodes that are updated first, and lower propagation preference to the nodes that are updated last.

The latter enables us to stabilize the basic label propagation algorithm. As random node updates cannot be avoided (Sect. 2), node propagation preferences can be utilized to counteract the randomness introduced by random updates. Node preferences are thus employed to balance the algorithm (i.e., *node balancers*) and are set according to the reverse order in which the nodes are assessed. This retains the dynamics of the basic algorithm, but greatly improves its robustness and the stability of the identified community structure.

Let nodes N be ordered in some random way, and let i_n denote the normalized position of node $n \in N$ in this order. Hence,

$$i_n = \frac{\text{index of node } n}{|N|}, \tag{3}$$

where $i_n \in (0, 1]$. Assuming linearity, we introduce node balancers as

$$p_n = i_n, \tag{4}$$

where p_n is the preference of node $n \in N$ (see Eq. (2)). Note that node balancers have to be recomputed at the beginning of each iteration (i.e., after each random shuffling of nodes). The resulting algorithm is else identical to the basic label propagation (with node preferences) and is denoted *balanced propagation algorithm* (*BPA*). Empirical evaluation in Section 4 shows that balanced propagation is not only more stable than basic label propagation, but also improves its community detection. Note also that the revealed community structure could be even further stabilized by, e.g., combining multiple network partitions [31].

We also analyze a variant of the algorithm, where logistic function is used to model the relation between update orders and propagation preferences (the algorithm is denoted BPA_L). Hence, node balancers are set due to

$$p_n = \frac{1}{1 + e^{-\beta(i_n - \alpha)}}, \tag{5}$$

where α and β are parameters of the algorithm. We fix $\alpha = \frac{1}{2}$ and $\beta = 5$ based on some preliminary experiments. Empirical analysis reveals that BPA_L usually performs slightly better than *BPA* (Sect. 4).

Last, we also briefly consider the second main issue of label propagation. As already discussed in Section 2, nodes having equally strong connections with several (overlapping) communities might prevent the algorithm from converging. The problem is even enhanced in the case of balanced propagation, as random node preferences, introduced through random update orders, can extend the issue to cases, where node has only similarly strong connections with different communities. Consequently, solutions proposed in the literature [7,18] do not necessarily overcome the problem in the case of balanced propagation.

Still, the true reason behind these convergence problems is the existence of overlapping communities in real-world networks. However, the purpose of this research is to address issues with random update orders, and not to extend balanced propagation to overlapping communities (see, e.g., [23]). Thus, for the sake of the empirical analysis, we adopt the following simple approach (and limit the analysis to non-overlapping communities).

As the discussed problems of balanced propagation (i.e., *BPA* and BPA_L algorithms) are actually an artifact of node balancers, we simply discard their use, when the algorithm does not converge after at most some maximal number of iterations. Note that this is in fact identical to applying the basic label propagation (i.e., *LPA* algorithm) afterwards, which obviously ensures the algorithm's convergence. We fix the maximum number of iterations to 100, what should suffice for networks with almost a billion edges [13].

4 Experiments and discussion

First, balanced propagation was analyzed, and compared against label propagation, on synthetic benchmark networks with planted partition and on several real-world networks with community structure (Sects. 4.1, 4.2 respectively). We address the stability of the algorithms and also the accuracy of community detection. Next, the proposed algorithm was further applied to a complete European road network, when the results are analyzed and discussed in Section 4.3.

Due to generality, results in the following sections are assessed in terms of different measures of community structure significance. Earlier work commonly reported the *modularity* Q [32] of the identified community structure. Modularity measures the significance of communities due to some *null model* (which is considered to be without community structure). Commonly, a random graph with the same degree sequence is selected for the null model. Hence,

$$Q = \frac{1}{2|E|} \sum_{n,m \in N} \left(A_{nm} - \frac{k_n k_m}{2|E|} \right) \delta(c_n, c_m), \tag{6}$$

where A is the adjacency matrix of the network, k_n is degree of node $n \in N$ and δ is the Kronecker delta. Higher values represent more significant community structure ($Q \in [-1, 1]$), however, recent work shows that modularity has a number of severe deficiencies [33–35] and should not be considered as a reliable indicator of community structure.

For a more adequate assessment of the significance of revealed communities we also adopt the *conductance* Φ [36]. Let $S \subset N$ be some community in the network thus $|S| \leq |N|/2$. Conductance of a set of nodes S is then defined as

$$\Phi = \frac{\sum_{n \in S, m \in \overline{S}} A_{nm}}{\min\{k(S), k(\overline{S})\}}, \tag{7}$$

where $\overline{S}$ is the complement of S and $k(S)$ is the cumulative degree of S (i.e., $k(S) = \sum_{n \in S} k_n$). Conductance thus measures the goodness of community S, or equivalently, the quality of corresponding network cut $(S, \overline{S})$. Lower values represent more significant communities ($\Phi \in [0, 1]$). Nevertheless, conductance cannot be easily extended to an entire community structure of a network. Thus, results are commonly assessed at different scales separately, in the form of *network community profile* (*NCP*) [37] plots. Still, due to simplicity, we also define $\overline{\Phi}$ as the average conductance over all communities in a network.

For networks with known community structure, identified communities are also compared against the true ones.

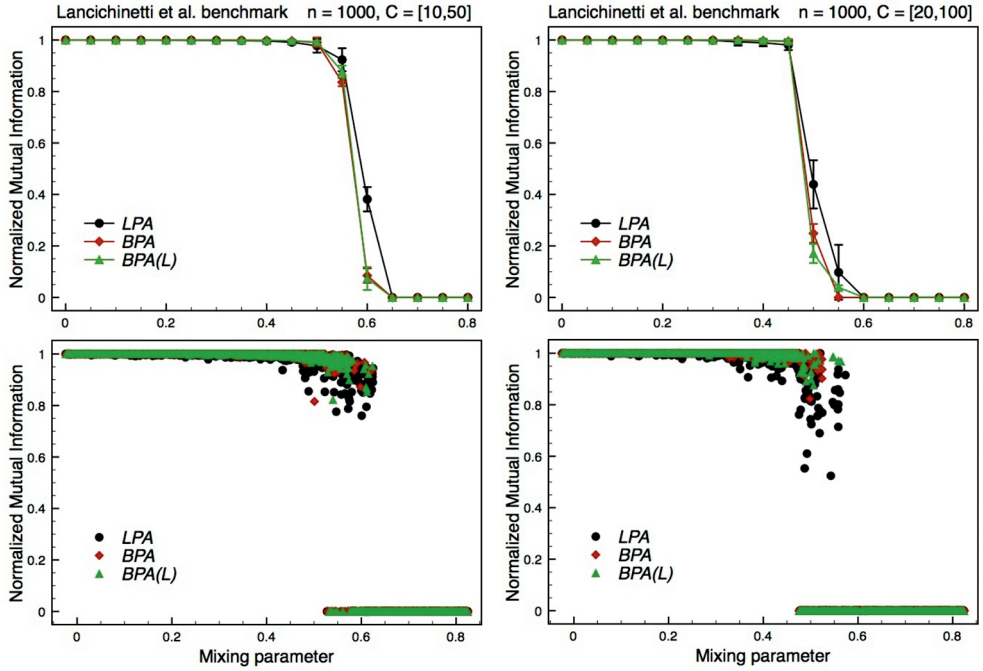


Fig. 2. (Color online) Comparison of balanced and label propagation on synthetic benchmark networks with planted partition [42]. The number of nodes is fixed to 1000 and the sizes of communities vary between $[10, 50]$ and $[20, 100]$ nodes (left, right respectively). We report the averages over 100 realizations and also the scatter plots showing individual runs (top, bottom respectively). For the former, error bars correspond to sample standard deviations computed from only nontrivial partitions (i.e., with $NMI > 0$), and for the latter, a small amount of noise was added along the horizontal axes.

We adopt two measures from the field of information theory [38]. First, *normalized mutual information* (*NMI*) [39], has become a *de facto* standard in the community detection literature. Let $\mathcal{C}$ be a partition (i.e., communities) extracted by some algorithm, and let $\mathcal{P}$ be the known partition for some network (corresponding random variables are C and P respectively). *NMI* of $\mathcal{C}$ and $\mathcal{P}$ is then

$$NMI = \frac{2I(C,P)}{H(C) + H(P)}, \tag{8}$$

where $I(C,P)$ is the mutual information of the partitions (i.e., $I(C,P) = H(C) - H(C|P)$), and $H(C)$, $H(P)$ and $H(C|P)$ are standard and conditional entropies. *NMI* of identical partitions equals 1, and is 0 for independent partitions ($NMI \in [0,1]$).

Second, *variation of information* (*VOI*) [40], has several desirable properties with respect to *NMI*. In particular, it is symmetric local measure that also has the properties of a distance in the space of partitions. *VOI* of $\mathcal{C}$ and $\mathcal{P}$ is defined as

$$VOI = H(C|P) + H(P|C), \tag{9}$$

thus, lower values represent better correlation between partitions. The maximum value of *VOI* depends on the size of the network ($VOI \in [0, \log|N|]$), therefore, for meaningful comparisons, we divide the obtained values with $\log|N|$ [41].

4.1 Synthetic networks with planted partition

We have first analyzed the balanced propagation on a class of synthetic benchmark networks with planted partition [42]. The significance of community structure is controlled by a mixing parameter $\mu \in [0,1]$, where smaller values give clearer community structure. Networks exhibit power-law degree and community size distributions, as commonly observed in real-world networks [43,44]. Power-law exponents α are set to 2 and 1 respectively (i.e., $P(x) \sim x^{-\alpha}$). Moreover, we fix the number of nodes to 1000 and vary the sizes of communities between $[10, 50]$ and $[20, 100]$ nodes. Results are assessed in terms of *NMI* and are shown in Figure 2.

Considering only the average performance (Fig. 2, top), no clear difference between balanced propagation (i.e., *BPA* and BPA_L algorithms) and label propagation (i.e., *LPA* algorithm) is observed. However, scatter plots showing individual runs (Fig. 2, bottom) reveal that there is actually a significant disparity between the approaches. When community structure is only roughly defined (i.e., for $\mu > 0.5$), balanced propagation either relatively accurately identifies communities in the network (i.e., $NMI \approx 1$) or classifies all nodes into a single community (i.e., $NMI = 0$). On the other hand, label propagation also commonly reports community structures, whose correspondence to the actual communities is only marginal (i.e., $NMI \approx 0.75$, $NMI \approx 0.5$ respectively). The latter is particularly apparent in the case of larger communities (note also the difference in error bars).

Table 1. Real-world networks with community structure.

Network	Description	Nodes	Edges
Karate	Zachary's karate club [47]	34	78
Dolphins	Lusseau's dolphins [48]	62	159
Books	Political books [49]	105	441
Football	American football [1]	115	616
Jazz	Jazz musicians [50]	198	2742
Elegans	Nematode *C. elegans* [51]	453	2025
Netsci	Network scientists [52]	1589	2742
Power	U.S. power grid [29]	4941	6594

The results thus confirm that balanced propagation is much more robust than simple label propagation, when the community detection strength of the basic algorithm is largely retained in the refined versions (on average). Still, to obtain results comparable with current state-of-the-art community detection algorithms (see [45]), different advances of the basic approach have to be employed [13,25].

To further address the validity of balanced propagation, we have also applied the algorithms to a random graph à la Erdös-Rényi [46] that (presumably) has no community structure. The number of nodes is again fixed to 1000, when we vary the average degree k between 10 and 100. Both balanced propagation algorithms reveal no community structure in these networks – all nodes are classified into a single community (or multiple communities in the case of disconnected networks) in all 100 realizations of random networks. On the other hand, label propagation also partitions the networks into non-trivial communities, when the average degree is small enough (i.e., for $k \leq 10$).

4.2 Real-world networks with community structure

Balanced propagation was further analyzed on eight real-world networks with community structure (Tab. 1). All these network are commonly employed in the community detection literature, and include different social, biological and technological networks. Due to simplicity, all networks were treated as unweighed and undirected.

We first directly compare the stability of the revealed community structures for balanced and label propagation (i.e., *BPA* and BPA_L, and *LPA* algorithms respectively). We apply the algorithms to each network 1000 times and count the number of distinct community structures obtained. We also measure the pairwise *VOI* of the partitions, to further evaluate the robustness of the algorithms. Due to space complexity, analysis is reduced to smaller networks (with at most hundreds of nodes). Results can be seen in Table 2.

The analysis confirms earlier observations that basic label propagation is relatively unstable, even on smaller networks [7,13,16,19]. However, the latter does not hold for balanced propagation that reveals only a small number of distinct community structures in each network. In most cases, this number is for a scale smaller than in the case of label propagation. Moreover, the pairwise similarity between the structures is also significantly improved, when the same trend is observed if we measure similarity

Table 2. Analysis of the stability of balanced and label propagation. We report the number of distinct community structures obtained over 1000 runs and the average pairwise *VOI* of the corresponding partitions.

Network	Distinct			Pairwise *VOI*		
	LPA	*BPA*	BPA_L	*LPA*	*BPA*	BPA_L
Karate	184	24	**19**	0.276	0.199	**0.192**
Dolphins	525	39	**36**	0.256	0.084	**0.079**
Books	269	37	**29**	0.124	**0.100**	**0.100**
Football	414	180	**154**	0.095	0.093	**0.087**
Jazz	63	22	**20**	0.107	0.032	**0.029**
Elegans	707	**76**	**75**	0.124	**0.015**	**0.015**

Table 3. Analysis of community detection strength of balanced and label propagation, and modularity optimization. We report *VOI* between the natural communities and those identified by the algorithms (results are averages over 1000 runs).

Network	Number	*VOI*			
		LPA	*BPA*	BPA_L	*MO*
Karate	2	0.239	0.145	**0.142**	0.218
Dolphins	2	0.363	**0.063**	**0.062**	0.257
Football	12	**0.155**	0.169	0.168	0.323

only among distinct structures (e.g., for *elegans* network, average pairwise *VOI* equals 0.1558, 0.0430 and 0.0424 for *LPA*, *BPA* and BPA_L algorithms respectively).

We conclude that balanced propagation is significantly more robust than label propagation, and can be, despite its randomized nature, considered as fairly stable. Note also that balanced propagation with logistic model (i.e., BPA_L algorithm) performs slightly better than the basic algorithm with a linear model (i.e., *BPA* algorithm).

Three of the networks in Table 1, namely *karate*, *dolphins* and *football*, have known natural partitions into communities (that result from earlier studies). To analyze also the community detection strength of balanced propagation, we measure the *VOI* between the natural partitions and those identified by different algorithms. The results appear in Table 3, when we also report the results for a classical modularity optimization algorithm (*MO*) proposed by Clauset et al. [4] (for reference).

Note that, in the case of *karate* and *dolphins* networks, balanced propagation performs significantly better than label propagation (and modularity optimization), when in the case of *football* network, the obtained *VOI* is roughly the same. Thus, despite relatively similar performance on synthetic benchmark networks (Sect. 4.1), balanced propagation more accurately identifies the true communities within these real-world networks than label propagation (and also modularity optimization).

For a better comprehension, the *fraction of correctly classified* [1] nodes for BPA_L algorithm equals 72%, 96% and 81% for *karate*, *dolphins* and *football* networks respectively (on average).

In Table 4 we also report average conductance $\overline{\Phi}$ and modularity Q of the revealed community structures for all networks in Table 1 (mainly to enable comparison with earlier work). Balanced propagation also performs better

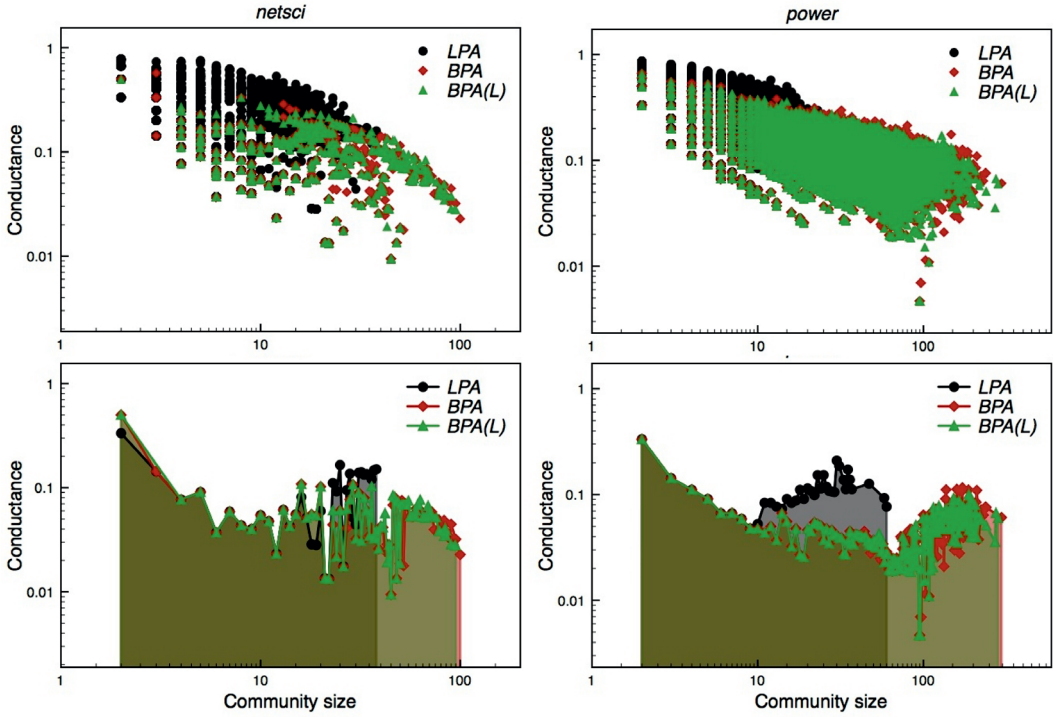


Fig. 3. (Color online) Comparison of balanced and label propagation on *netsci* and *power* networks. We report the scatter plots showing individual communities, and the minimum values (i.e., lower hulls) at different scales (top, bottom respectively). Results were obtained over 100 runs.

in terms of conductance. Still, results should be taken with caution as *BPA* and BPA_L algorithms commonly return larger communities than *LPA* algorithm, which implies lower average conductance (see below). On the other hand, according to modularity, performance depends on the size of the network. We argue that this is an artifact of an intrinsic scale incorporated into the measure of modularity (i.e., *resolution limit* [33,35]), thus, lower values of modularity obtained by balanced propagation on smaller networks should not be attributed to weaker community structure (see Tab. 3).

Again, a general pattern can be observed between both balanced propagation algorithms.

Next, we further analyze the larger two networks in Table 1, namely, *netsci* and *power*. We apply each algorithm 100 times and analyze the conductance of obtained communities at different scales. The results are reported in the form of *network community profile* (*NCP*) [37] plots, and are shown in Figure 3. *NCP* plots measure the quality of the best community (due to conductance) as a function of its size (Fig. 3, below). Social and information, and also technological, networks commonly reveal rather characteristic structure of *NCP* plots, with initial decreasing and subsequent increasing trend (for more see [37]).

Observe that balanced propagation identifies communities on a much wider scale, including also larger communities. The structure of *NCP* plots thus better coincides with the analysis of Leskovec et al. [37], where a natural (i.e., best) community size was estimated to a round 100 nodes. In other words, basic label propagation finds best communities at much smaller scale than balanced propagation (i.e., at a round 10 nodes), when the conductance is also significantly higher on average (Tab. 4). Note also that label propagation reveals a number of communities with very high conductance (i.e., (black) circles in the uppermost part of Fig. 3, top), which can be directly related to the issues of the algorithm discussed in Section 2.

Table 4. Analysis of community detection significance of balanced and label propagation. We report the average conductance $\overline{\Phi}$ and modularity Q of communities identified by different algorithms (results are averages over 1000 runs).

Net.	$\overline{\Phi}$			Q		
	LPA	*BPA*	BPA_L	*LPA*	*BPA*	BPA_L
Kara.	0.285	0.254	**0.242**	**0.355**	0.296	0.301
Dolph.	0.345	0.082	**0.078**	**0.485**	0.377	0.380
Books	0.272	**0.063**	**0.062**	**0.505**	0.460	0.460
Foot.	0.328	**0.295**	**0.296**	0.593	**0.602**	**0.602**
Jazz	0.210	**0.141**	**0.142**	**0.340**	0.285	0.285
Eleg.	0.354	0.120	**0.117**	**0.117**	0.036	0.037
Netsci	0.063	**0.006**	**0.007**	0.879	**0.945**	**0.944**
Power	0.431	**0.129**	**0.129**	0.595	**0.888**	**0.887**

We conclude that, at least for the networks analyzed, balanced propagation is indeed more stable than basic label propagation, when the quality of the identified community structure is also improved in most cases.

Last, we also briefly analyze the scalability of the proposed balanced propagation. In Table 5 we report the average number of iterations[2] made by the algorithms over

Table 5. Analysis of complexity of balanced and label propagation. We report the average number of iterations made by the algorithms over 1000 runs (see text).

Network	Iterations		
	LPA	*BPA*	BPA_L
Karate	**3.8**	12.6	12.8
Dolphins	**4.9**	21.5	22.3
Books	**4.9**	31.0	28.8
Football	**3.7**	23.4	22.7
Jazz	**4.8**	25.9	25.0
Elegans	**7.1**	16.1	16.1

1000 runs. As discussed in Section 3, we do not directly address the issues with overlapping communities. Therefore, nodes, having strong connections with different communities, can prevent basic balanced propagation from converging. The results in Table 5 thus include only the runs where the algorithms converged in a fixed (maximal) number of iterations (this includes at least 90% of runs in each case). For the same reason, *netsci* and *power* networks were not included in the analysis.

The complexity of label propagation is quite lower compared to balanced propagation. Still, all algorithms reveal communities in a relatively small number of iterations and can be easily scaled to larger networks (exhibit near linear time complexity $\mathcal{O}(|E|)$). It should also be noted that extremely fast convergence of label propagation can be somewhat related to random node updates (Sect. 2). Random update order can be seen as increasing propagation strength from certain nodes (Sect. 3), which limits the dynamics of the algorithm, and instantly leads it towards some stable, probably suboptimal (i.e., random), partition. The convergence of the algorithm is thus indeed fast, still, the identified community structure is extremely unstable and often suboptimal (as also observed by previous work [7,13,16,19]).

4.3 European road network

Road networks are not considered to convey a clear community structure, consisting of densely connected modules (due to sparsity of such networks). However, the network can still contain groups of nodes that are well isolated from others (i.e., connected through only few edges) and community detection algorithms can be employed to reveal such partition of the network. Communities should in this case largely relate to the properties of the road transport within the region, and also coincide with the geographical characteristics of the area.

We have constructed a network of all roads included in the *international E-road network* (Fig. 4). Nodes thus correspond to European cities and edges represent direct (class A, B) road connections among them. We limit the analysis to the main component of the network that consists of 1039 nodes and 1355 edges (a complete network has 1177 nodes and 1469 edges). Note that the network is neither *scale-free* [43] (i.e., maximum degree equals 10, when the degree distribution is, e.g., log-normal) nor *small-world* [29] (i.e., average distance among nodes is $l = 18.40$ and the clustering coefficient [29] equals $C = 0.02$).

Due to long average distances among different parts of the network, road networks are particularly hard to partition with standard community detection algorithms. Furthermore, as the network has almost tree-like structure, it is often hard to decide where to split long paths of nodes. Indeed, if we apply the basic label propagation (i.e., *LPA* algorithm) we obtain 343 communities with $Q = 0.5617$ and $\overline{\Phi} = 0.4424$ (on average over 1000 runs). Hence, communities consist of only 3.03 nodes on average, thus, they can only hardly be considered as meaningful.

On the other hand, balanced propagation (i.e., *BPA* algorithm) partitions the network into 35 communities with $Q = 0.8374$ and $\overline{\Phi} = 0.1224$ (on average over 1000 runs). In Figure 4 we show the community structure that obtained minimum average conductance $\overline{\Phi}$. Note how the largest communities quite accurately coincide with different (geographical) regions of Europe. In particular, from left to right (top to bottom), communities represent cities of Iberian Peninsula (e.g., Madrid), eastern Central Europe (e.g., Berlin), western Central Europe (e.g., Paris), Apennine Peninsula (e.g., Rome), eastern Russia, western Russia and Finland (e.g., Moscow), northern East Europe (e.g., Bratislava), southern East Europe (e.g., Bucharest), Balkan Peninsula (e.g., Skopje), Scandinavian Peninsula (e.g., Stockholm), etc. It is ought to be mentioned that, although community structures revealed by the algorithm through different runs indeed differ, in most cases, largest communities correspond to the same regions as discussed above. The latter thus further confirms the robustness of the balanced propagation.

5 Conclusions

The article addresses one of the main issues of label propagation algorithm for community detection – the stability of the identified community structure. We introduce balanced propagation that controls (i.e., stabilizes) the dynamics of basic label propagation through utilization of node balancers. The resulting approach is significantly more robust than its label propagation counterpart, when its community detection strength is even improved. Thus, balanced propagation retains high scalability and algorithmic simplicity of label propagation, but improves on its stability and performance. The proposition has been validated on synthetic networks with planted partition, and on several real-world networks with community structure. Moreover, the proposed algorithm was further applied to an entire European road network, where it accurately partitions the network with respect to (geographical) regions.

Due to its simplicity, balanced propagation can be easily incorporated into arbitrary (label) propagation algorithm, not limited to the field of community detection. Moreover, the work provides further comprehension of the propagation on networks, with different applications.

[2] Each iteration has linear time complexity $\mathcal{O}(|E|)$.

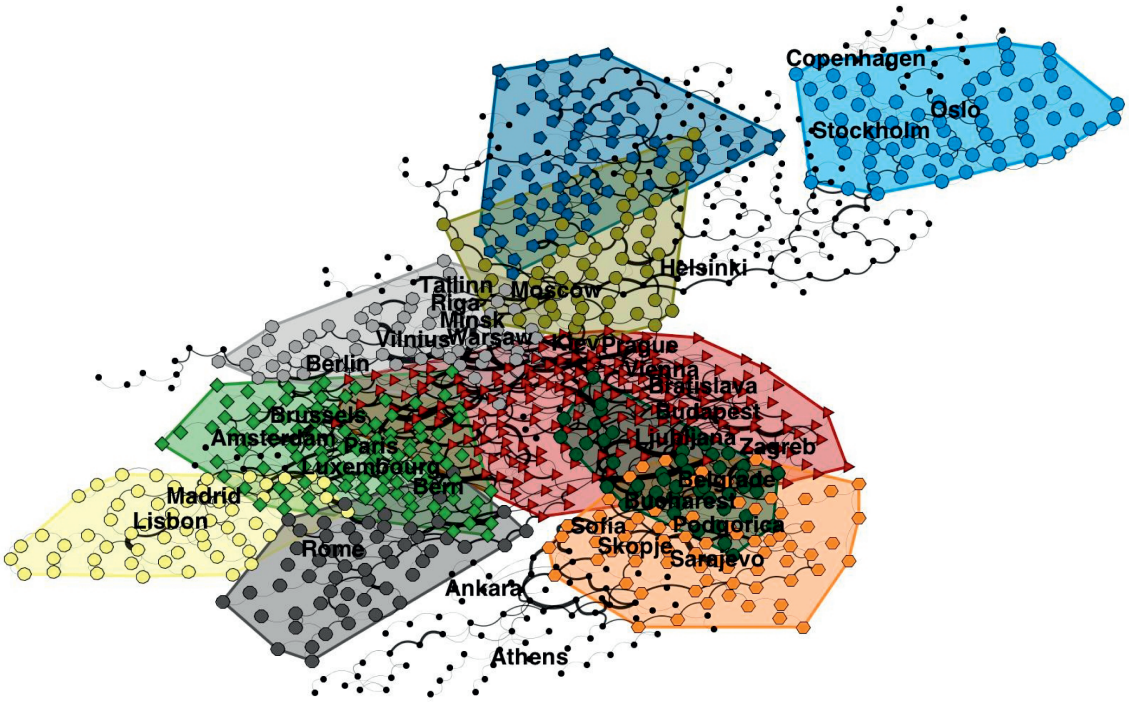


Fig. 4. (Color online) Community structure of the main component of European road network revealed with balanced propagation (i.e., *BPA* algorithm). Node symbols (colors) correspond to different communities, when edge widths represent significant inter-community edges. Due to clarity, only the largest 10 communities of total 24 are shown ($Q = 0.8344$ and $\overline{\Phi} = 0.0796$). Note how communities quite accurately coincide with different (geographical) regions of Europe.

The work has been supported by the Slovene Research Agency *ARRS* within the research program P2-0359.

References

1. M. Girvan, M.E.J. Newman, P. Natl. Acad. Sci. USA **99**, 7821 (2002)
2. G. Palla, I. Derényi, I. Farkas, T. Vicsek, Nature **435**, 814 (2005)
3. A. Arenas, A. Díaz-Guilera, C.J. Pérez-Vicente, Phys. Rev. Lett. **96**, 114102 (2006)
4. A. Clauset, M.E.J. Newman, C. Moore, Phys. Rev. E **70**, 066111 (2004)
5. F. Wu, B.A. Huberman, Eur. Phys. J. B **38**, 331 (2004)
6. S. Son, H. Jeong, J.D. Noh, Eur. Phys. J. B **50**, 431 (2006)
7. U.N. Raghavan, R. Albert, S. Kumara, Phys. Rev. E **76**, 036106 (2007)
8. G. Agarwal, D. Kempe, Eur. Phys. J. B **66**, 409 (2008)
9. V.D. Blondel, J. Guillaume, R. Lambiotte, E. Lefebvre, J. Stat. Mech. P10008 (2008)
10. M. Rosvall, C.T. Bergstrom, P. Natl. Acad. Sci. USA **105**, 1118 (2008)
11. J. Liu, Eur. Phys. J. B **77**, 547 (2010)
12. P. Ronhovde, Z. Nussinov, Phys. Rev. E **81**, 046114 (2010)
13. L. Subelj, M. Bajec, Phys. Rev. E **83**, 036103 (2011)
14. S. Fortunato, Phys. Rep. **486**, 75 (2010)
15. Y. Hu, H. Chen, P. Zhang, M. Li, Z. Di, Y. Fan, Phys. Rev. E **78**, 026121 (2008)
16. G. Tibély, J. Kertész, Physica A **387**, 4982 (2008)
17. M.J. Barber, J.W. Clark, Phys. Rev. E **80**, 026129 (2009)
18. I.X.Y. Leung, P. Hui, P. Liò, J. Crowcroft, Phys. Rev. E **79**, 066107 (2009)
19. X. Liu, T. Murata, Physica A **389**, 1493 (2009)
20. X. Liu, T. Murata, Community detection in large-scale bipartite networks, in *Proceedings of the International Conference on Web Intelligence and Intelligent Agent Technology* (2009), Vol. 1, pp. 50–57
21. S. Pang, C. Chen, T. Wei, A realtime clique detection algorithm: Time-based incremental label propagation, in *Proceedings of the International Conference on Intelligent Information Technology Application* (2009), Vol. 3, pp. 459–462
22. C. Pang, F. Shao, R. Sun, S. Li, Detecting community structure in networks by propagating labels of nodes, in *Proceedings of the International Symposium on Neural Networks* (2009), pp. 839–846
23. S. Gregory, New J. Phys. **12**, 103018 (2010)
24. X. Liu, T. Murata, Evaluating community structure in bipartite networks, in *Proceedings of the IEEE International Conference on Social Computing* (2010), pp. 576–581
25. L. Subelj, M. Bajec, Unfolding network communities by combining defensive and offensive label propagation, in *Proceedings of the ECML PKDD Workshop on the Analysis of Complex Networks* (2010), pp. 87–104
26. Q. Ye, B. Wu, Y. Gao, B. Wang, Detecting communities in massive networks based on local community attractive force optimization, in *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining* (2010), pp. 291–295
27. L. Freeman, Sociometry **40**, 35 (1977)
28. L.C. Freeman, Soc. Networks **1**, 215 (1979)
29. D.J. Watts, S.H. Strogatz, Nature **393**, 440 (1998)

30. F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, D. Parisi, P. Natl. Acad. Sci. USA **101**, 2658 (2004)
31. A. Strehl, J. Ghosh, J. Mach. Learn. Res. **3**, 583 (2002)
32. M.E.J. Newman, M. Girvan, Phys. Rev. E **69**, 026113 (2004)
33. S. Fortunato, M. Barthelemy, P. Natl. Acad. Sci. USA **104**, 36 (2007)
34. J. Kumpula, J. Saramäki, K. Kaski, J. Kertész, Eur. Phys. J. B **56**, 5 (2007)
35. B.H. Good, Y.A. de Montjoye, A. Clauset, Phys. Rev. E **81**, 046106 (2010)
36. B. Bollobás, *Modern graph theory* (Springer, 1998)
37. J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Internet Mathematics **6**, 29 (2009)
38. D.J.C. MacKay, *Information theory, inference, and learning algorithms* (Cambridge University Press, 2003)
39. L. Danon, A. Díaz-Guilera, J. Duch, A. Arenas, J. Stat. Mech. P09008 (2005)
40. M. Meila, J. Multivar. Anal. **98**, 873 (2007)
41. B. Karrer, E. Levina, M.E.J. Newman, Phys. Rev. E **77**, 046119 (2008)
42. A. Lancichinetti, S. Fortunato, F. Radicchi, Phys. Rev. E **78**, 046110 (2008)
43. A.L. Barabási, R. Albert, Science **286**, 509 (1999)
44. M.E.J. Newman, Eur. Phys. J. B **38**, 321 (2004)
45. A. Lancichinetti, S. Fortunato, Phys. Rev. E **80**, 056117 (2009)
46. P. Erdős, A. Rényi, Publ. Math. Debrecen **6**, 290 (1959)
47. W.W. Zachary, J. Anthropol. Res. **33**, 452 (1977)
48. D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, S.M. Dawson, Behav. Ecol. Sociobiol. **54**, 396 (2003)
49. V. Krebs, *A network of co-purcheased books about U.S. politics* (2008), `http://www.orgnet.com/`
50. P. Gleiser, L. Danon, Adv. Compl. Syst. **6**, 565 (2003)
51. H. Jeong, B. Tombor, R. Albert, Z.N. Oltvai, A. Barabási, Nature **407**, 651 (2000)
52. M.E.J. Newman, Phys. Rev. E **74**, 036104 (2006)

4

Odkrivanje skupin vozlišč z napredno izmenjavo oznak

V pričujočem poglavju je vloženo delo z naslovom "Odkrivanje skupin vozlišč z napredno izmenjavo oznak" [99] (angl. "Unfolding communities in large complex networks: Combining defensive and offensive label propagation for core extraction"). Delo predstavi različne metode in tehnike odkrivanja karakterističnih skupin vozlišč v omrežjih na osnovi izmenjave oznak, ki temeljijo na različnih strategijah preferenčnih vozlišč ter hierarhičnemu preiskovanju. Le-te močno izboljšajo natančnost osnovnega pristopa, pri tem pa je asimptotična časovna zahtevnost blizu idealne (t.j. linearne). Predlagane izboljšave pristopa so preizkušene na številnih sintetičnih in realnih omrežij z znano zgradbo ter primerjane z najboljšimi pristopi v literaturi.

Predstavljeno delo je v razširjeni obliki objavljeno v [99, 126].[3]

[3]Privzeta programska koda ter omrežja so dostopna na `http://lovro.lpt.fri.uni-lj.si/`.

PHYSICAL REVIEW E **83**, 036103 (2011)

Unfolding communities in large complex networks: Combining defensive and offensive label propagation for core extraction

Lovro Šubelj[*] and Marko Bajec[†]
University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia
(Received 1 June 2010; revised manuscript received 6 November 2010; published 8 March 2011)

Label propagation has proven to be a fast method for detecting communities in large complex networks. Recent developments have also improved the accuracy of the approach; however, a general algorithm is still an open issue. We present an advanced label propagation algorithm that combines two unique strategies of community formation, namely, defensive preservation and offensive expansion of communities. The two strategies are combined in a hierarchical manner to recursively extract the core of the network and to identify whisker communities. The algorithm was evaluated on two classes of benchmark networks with planted partition and on 23 real-world networks ranging from networks with tens of nodes to networks with several tens of millions of edges. It is shown to be comparable to the current state-of-the-art community detection algorithms and superior to all previous label propagation algorithms, with comparable time complexity. In particular, analysis on real-world networks has proven that the algorithm has almost linear complexity, $O(m^{1.19})$, and scales even better than the basic label propagation algorithm (m is the number of edges in the network).

DOI: 10.1103/PhysRevE.83.036103 PACS number(s): 89.75.Fb, 89.75.Hc, 87.23.Ge, 89.20.Hh

I. INTRODUCTION

Large real-world networks can comprise local structural modules (*communities*) that are groups of nodes, densely connected within and only loosely connected with the rest of the network. Communities are believed to play important roles in different real-world systems (e.g., they may correspond to functional modules in metabolic networks [1]); moreover, they also provide a valuable insight into the structure and function of large complex networks [1–3]. Nevertheless, real-world networks can reveal even more complex modules than communities [4,5].

Over the last decade the research community has shown a considerable interest in detecting communities in real-world networks. Since the seminal paper of Girvan and Newman [6] a vast number of approaches have been presented in the literature—in particular, approaches optimizing modularity Q (the significance of communities due to a selected null model [7]) [8–12], graph partitioning [13,14] and spectral algorithms [9,15], statistical methods [4], algorithms based on dynamic processes [16–20], overlapping, hierarchical, and multiresolution methods [1,6,20], and others [21] (for an excellent survey see [22]).

The size of large real-world networks has forced the research community to develop scalable approaches that can be applied to networks with several millions of nodes and billions of edges. A promising effort was made by Raghavan *et al.* [18], who employed a simple label propagation to find significant communities in large real-world networks. Tibély and Kertész [23] have shown that label propagation is in fact equivalent to a large zero-temperature kinetic Potts model, while Barber and Clark [11] have further refined the approach into a modularity optimization algorithm. Just recently, Liu and Murata [12] have combined the modularity optimization version of the algorithm with a multistep greedy agglomeration [24] and derived an extremely accurate community detection algorithm.

Leung *et al.* [19] have investigated label propagation on large web networks, mainly focusing on scalability issues, and have shown that the performance can be significantly improved with label hop attenuation and by applying node preference (i.e., node propagation strength). We carry forward their work in developing two unique strategies of community formation, namely, defensive preservation of communities, where preference is given to the nodes in the core of each community, and offensive expansion of communities, where preference is given to the border nodes of each community. Cores and borders are estimated using random walks, formulating the diffusion over the network.

Furthermore, we propose an advanced label propagation algorithm—the diffusion and propagation algorithm—that combines the two strategies in a hierarchical manner: The algorithm first extracts the *core* of the network and identifies *whisker* communities [26] (Appendix A), and then recurses on the network core (Fig. 1). The performance of the algorithm has been analyzed on two classes of benchmark networks with planted partition and on 23 real-world networks ranging from networks with tens of nodes to networks with several tens of millions of edges. The algorithm is shown to be comparable to the current state-of-the-art community detection algorithms and superior to all previous label propagation algorithms, with comparable time complexity. In particular, the algorithm exhibits almost linear time complexity (in the number of edges of the network).

The rest of the paper is structured as follows. Section II gives a formal introduction to label propagation and reviews subsequent advances, relevant for this research. Section III presents the diffusion and propagation algorithm and discusses the main rationale behind it. Empirical evaluation with discussion is done in Secs. IV, and V is our conclusion.

[*]lovro.subelj@fri.uni-lj.si
[†]marko.bajec@fri.uni-lj.si

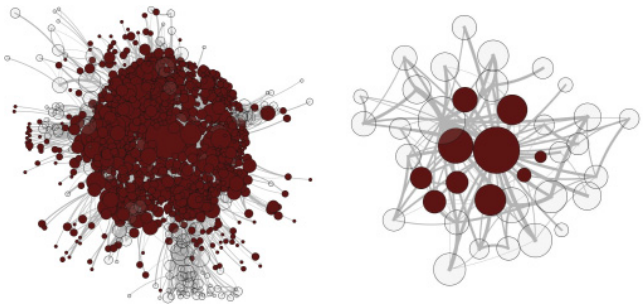


FIG. 1. (Color online) Results of diffusion and propagation algorithm applied to the network of autonomous systems of the Internet [25]. The figure shows two community networks, where the largest nodes correspond to densely connected modules of almost 10^4 nodes in the original network. Network cores, extracted by the algorithm, are colored red (dark gray) and whisker communities are represented with transparent nodes. The results show that the algorithm can detect communities on various levels of resolution—average community sizes are 16.38 and 588.79 nodes (with Q equal to 0.475 and 0.582, respectively).

II. LABEL PROPAGATION AND ADVANCES

Let the network be represented by an undirected graph $G(N,E)$, with N being the set of nodes of the graph and E being the set of edges. Furthermore, let c_n be a community (label) of node n, $n \in N$, and $\mathcal{N}(n)$ the set of its neighbors.

The basic label propagation algorithm (LPA) [18] exploits the following simple procedure. At first, each node is labeled with a unique label, $c_n = l_n$. Then, at each iteration, the node is assigned the label shared by most of its neighbors (i.e., the maximal label),

$$c_n = \underset{l}{\operatorname{argmax}} |\mathcal{N}^l(n)|, \tag{1}$$

where $\mathcal{N}^l(n)$ is the set of neighbors of n that share label l (in the case of ties, one maximal label is chosen at random). Due to the existence of multiple edges within the communities, relative to the number of edges between the communities, nodes in a community will adopt the same label after a few iterations. The algorithm converges when none of the labels changes anymore (i.e., equilibrium is reached) and nodes sharing the same label are classified into the same community.

The main advantage of label propagation is its nearly linear time complexity—the algorithm commonly converges in less then 10 iterations (on networks of moderate size). Raghavan *et al*. [18] observed that after 5 iterations 95% of the nodes already obtained their "right" label. Their observation can be further generalized: The number of nodes that change their label on the first 4 iterations roughly follow the sequence 90%, 30%, 10%, and 5%. However, because of the algorithm's simplicity, the accuracy of identified communities is often not state of the art (Sec. IV).

Leung *et al*. [19] have noticed that the algorithm, applied to large web networks, often produces a single large community, occupying more than a half of the nodes of the network. Thus, they have proposed a label hop attenuation technique, to prevent the label from spreading too far from its origin. Each label l_n has associated an additional score s_n (initially set to 1) that decreases after each propagation [Eq. (1)]. Hence,

$$s_n = \left(\max_{i \in \mathcal{N}^{c_n}(n)} s_i \right) - \delta, \tag{2}$$

with δ being the attenuation ratio. When s_n reaches 0, the label can no longer propagate onward [Eq. (3)], which successfully eliminates the formation of a single major community [19].

Leung *et al*. [19] have also shown that hop attenuation has to be coupled with node preference f_n (i.e., node propagation strength) in order to achieve superior performance. The label propagation updating rule [Eq. (1)] is thus reformulated into

$$c_n = \underset{l}{\operatorname{argmax}} \sum_{i \in \mathcal{N}^l(n)} f_i^{\alpha} s_i w_{ni}, \tag{3}$$

where w_{ni} is the edge weight (equal to 1 for unweighted graphs) and α is a parameter of the algorithm. They have experimented with preference equal to the degree of the node, $f_i = k_i$ and $\alpha = 0.1$; however, no general approach was reported.

Label hop attenuation in Eq. (2) can be rewritten into an equivalent form that allows altering δ during the course of the algorithm [19]. One keeps the label distance from the origin d_n (initially set to 0) that is updated after each propagation. Hence,

$$d_n = \left(\min_{i \in \mathcal{N}^{c_n}(n)} d_i \right) + 1, \tag{4}$$

when the score s_n is

$$s_n = 1 - \delta d_n. \tag{5}$$

Raghavan *et al*. [18] have already shown that the updating rule of label propagation [Eq. (1)], or its refinements [Eq. (3)], might prevent the algorithm from converging. Imagine a bipartite network with two sets of nodes, i.e., red and blue nodes. Let, at some iteration of the algorithm, all red nodes share label l_r and all blue nodes share label l_b. Because of the bipartite structure of the network, at the next iteration, all red and blue nodes will adopt the label l_b and l_r, respectively. Furthermore, at the next iteration all nodes will recover their original labels, preventing the algorithm from converging.

The problem can be avoided with asynchronous updating [18]. Nodes are no longer updated all together, but sequentially, in random order. Thus, when a node's label is updated, (possibly) already updated labels of its neighbors are considered (in contrast to synchronous updating, which considers only labels from the previous iteration). It should be noted that asynchronous updating can even increase the performance of the algorithm [19].

Furthermore, when a node has equally strong connections with two or more communities, its label will, in general, constantly change [18,19]. The problem is particularly apparent in author collaboration (co-authorship) networks, where a single author often collaborates with different research communities. On the collaboration network of network scientists [9] the basic label propagation algorithm fails to converge, as up to 10% of the nodes would change their label even after 10 000 iterations; the results suggest that the network contains at least 20% of such nodes, i.e., over 300 scientists collaborating with different research communities [28].

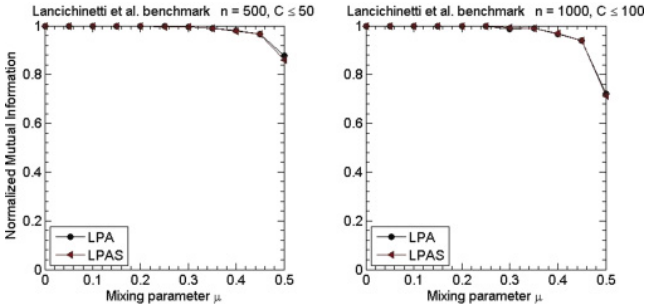


FIG. 2. Comparison of node access strategies for label propagation on two sets of benchmark networks with planted partition [27] (the results are averages over 100 realizations). Network sizes equal 500 and 1000 nodes, and communities comprise up to 50 and 100 nodes, respectively. LPA denotes the basic label propagation algorithm and LPAS denotes LPA without (subsequent) reshuffling of nodes.

Leung *et al.* [19] suggested including the concerned label itself in the maximal label consideration (and not merely neighbors' labels); however, we use a slightly modified version [18]. When there are multiple maximal labels among neighbors and one of them equals the concerned label, the node retains its label. The main difference here is that the modified version considers the concerned label only when there exist multiple maximal labels among neighbors. On the discussed collaboration network, such an algorithm converges in around 4 iterations.

Never-converging nodes can also be regarded as a clear signature of overlapping communities [1], where nodes can belong to multiple communities. Extension of label propagation to detect overlapping communities was recently proposed by Gregory [29] (and previously discussed in [18,19]). However, due to simplicity, we investigate only basic (no-overlap) versions of the label propagation algorithm.

Another important issue of label propagation is the stability of identified community structure [18], especially in large networks. For more detailed discussion see [12,18,23].

Label propagation with asynchronous updating accesses the nodes in a random order. Nodes are then shuffled after each iteration, mainly to address the problems discussed above. Although this subsequent reshuffling does not increase the algorithm's complexity, it does indeed increase its computational time. Nevertheless, the results in Fig. 2 show that LPA without subsequent reshuffling of nodes (LPAS) only slightly decreases the performance of the basic LPA. Thus, all the approaches presented in the following section use asynchronous updating with a single (initial) shuffling of nodes.

III. DIFFUSION AND PROPAGATION ALGORITHM

The section presents the diffusion and propagation algorithm that combines several approaches, also introduced in this section. We thus give here a brief review of these.

First, we further analyze label hop attenuation for LPA (Sec. II) and propose different dynamic hop attenuation strategies in Sec. III A. Next, we consider various approaches for node propagation preference (Sec. II). By estimating node preference by means of the diffusion over the network, we derive two algorithms that result in two unique strategies of community formation, namely, *defensive preservation* and *offensive expansion* of communities. The algorithms are denoted *defensive* and *offensive* diffusion and attenuation LPA (DDALPA and ODALPA) and are presented in Sec. III B.

The DALPA algorithms are combined into the basic diffusion and propagation algorithm (BDPA), preserving the advantages of both defensive and offensive approaches (Sec. III C). BDPA already achieves superior results on networks of moderate size (Sec. IV); for use with larger networks, the algorithm is further enhanced with core extraction and whiskers identification. The improved algorithm is denoted the (general) diffusion and propagation algorithm (DPA) and is presented in Sec. III C.

A. Dynamic hop attenuation

Hop attenuation has proven to be a reliable technique for preventing the emergence of a major community occupying most of the nodes of the network [19]. It is, however, not evident what the value of attenuation ratio δ should be [Eq. (2)]. Leung *et al.* [19] have experimented with values around 0.10 and obtained good results; still, their experimental setting was rather limited. Furthermore, our preliminary empirical analysis suggests that there is no (simple) universal value for δ applicable for *all* different types of networks (the results are omitted here).

Leung *et al.* [19] have also observed that large values of δ may prevent the natural growth of communities and have proposed a dynamic strategy that decreases δ from 0.50 toward 0. In the early iterations of the algorithm, large values of δ prevent a single label from rapidly occupying a large set of nodes and ensure the emergence of a number of strong community cores. The value of δ is then decreased, to gradually relax the restriction and to allow formation of the actual communities depicted in the network topology. The results on real-world networks show that such a strategy has very good performance on larger networks (Sec. IV); still, the results can be further improved. The empirical evaluation in Sec. IV also proves that the strategy is too aggressive for smaller networks, where it is commonly outperformed even by basic LPA.

We propose different dynamic hop attenuation strategies, based on the hypothesis that hop attenuation should only be employed when a community, or a set of communities, is rapidly occupying a large portion of the network. Otherwise, the restriction should be (almost) completely relaxed, to allow label propagation to reach the equilibrium unrestrained. Thus, the approach would retain the dynamics of label propagation and still prevent the emergence of a major community.

We have considered several strategies for detecting the emergence of a large community or a set of large communities. Because of limited space, we limit the discussion to two. After each iteration, the value of δ (initially set to 0) is updated according to the following rule:

nodes: δ is set to the proportion of nodes that changed their label;

communities: δ is set to the proportion of communities (i.e., labels) that disappeared.

Both strategies successfully address the problem of major community formation; however, a detailed comparison is

omitted here. The algorithms proposed here all use the *nodes* strategy, because of its much finer granularity, as opposed to the *communities* approach—after 4 iterations the number of communities is, in general, already 20 times smaller than the number of nodes (Sec. II); thus, the estimate of δ is rather rough for the *communities* strategy. For the empirical evaluation see Sec. IV.

B. Defensive and offensive propagation

Leung *et al.* [19] have proved that using node preference, to increase the propagation strength (i.e., label spread) from certain nodes, can improve the performance of basic LPA. We conducted several experiments by using variations of different measures of node centrality for node propagation preference (i.e., degree and eigenvector centrality [30,31] and node clustering coefficient [32]). The results are omitted here; however, they clearly indicate that none of these static measures applies for *all* different types of networks (i.e., general networks).

We have also observed that good performance can be obtained by putting higher preference to the core of each community (i.e., to its most central nodes). For instance, on the Zachary's karate club network [33], where three high-degree nodes reside in the core of the two (natural) communities, degree and eigenvector centralities are superior. However, on Girvan and Newman [6] benchmark networks, where all the nodes have equal degree (on average), the measures are rendered useless and are outperformed by the node clustering coefficient. On the Lancichinetti *et al.* [27] benchmark networks, the best performance is, interestingly, obtained by inverted degree or eigenvector centrality. The measures seem to counter each node's degree (low-degree nodes have high propagation strength, and vice versa); thus, the propagation utilizes merely the connectedness among nodes, disregarding its strength.

Based (also) on the above observations, we have developed two algorithms that estimate node preference by means of the diffusion over the network. During the course of the algorithms, the diffusion is formulated using a random walker within each of the (current) communities of the network. The rationale here is twofold: (1) to estimate the (label) propagation within each of the (current) communities [34], and (2) to derive an estimation of the core and border of each (current) community (with the core being the most central nodes of the community and the border being its edge nodes).

Let p_n be the probability that a random walker, utilized on the community labeled with c_n, visits node n. p_n can be computed as

$$p_n = \sum_{i \in \mathcal{N}^{c_n}(n)} p_i / k_i^{c_n}, \tag{6}$$

where the sum goes over all the neighbors of n within the community c_n, and $k_i^{c_n}$ is the intracommunity degree of node i. The employed formulation is similar to algorithms such as PageRank [35] and HITS [36], and also to the basic eigenvector centrality measure.

Finally, we present the two algorithms mentioned above, namely, defensive and offensive diffusion and attenuation LPA (DDALPA and ODALPA). The defensive algorithm applies preference (i.e., propagation strength) to the core of each community, i.e., $f_n^\alpha = p_n$, and the updating rule in (3) is rewritten as

$$c_n = \operatorname*{argmax}_l \sum_{i \in \mathcal{N}^l(n)} p_i s_i w_{ni}. \tag{7}$$

On the other hand, the offensive version applies preference to the border of each community, i.e., $f_n^\alpha = 1 - p_n$, and the updating rule becomes

$$c_n = \operatorname*{argmax}_l \sum_{i \in \mathcal{N}^l(n)} (1 - p_i)\, s_i w_{ni}. \tag{8}$$

As opposed to the algorithm of Leung *et al.* [19], the main novelty here is in considering (current) communities, found by the algorithm, to estimate the (current) state of the label propagation process and then to adequately alter the dynamics of the process.

To better estimate the border of each community, the offensive algorithm uses degrees k_i (instead of intracommunity degrees $k_i^{c_n}$) for the estimation of diffusion values p_n [see Eq. (6)]. The modification results in higher values of $1 - p_n$ for nodes with large intercommunity degrees (i.e., nodes that reside in the borders of communities) and thus provides more adequate formulation of the node propagation strength for the offensive version (the results are omitted here).

When a node's label changes, the values p_n should be reestimated for each node in the concerned node's previous or current community. However, this would likely render the algorithm inapplicable on larger networks. Thus, we only update the value p_n [according to Eq. (6)] when the node n changes its label (initially all p_n are set to $1/|N|$). Although the approach is only a rough approximation of an exact version, preliminary empirical experiments reveal no significant gain by using the exact values for p_n.

Defensive and offensive label propagation algorithms result in two unique strategies of community formation, namely, defensive preservation and offensive expansion of communities. The defensive algorithm quickly establishes a larger number of strong community cores [in the sense of Eq. (7)] and is able to defensibly preserve them during the course of the algorithm. This results in an immense ability of detecting communities, even when they are only weakly defined in the network topology. On the other hand, the offensive approach produces a range of communities of various sizes, as commonly observed in the real-world networks [3,18]. Laying the pressure on the border of each community expands those that are strongly defined in the network topology. This constitutes a more natural (offensive) struggle among the communities and results in a great accuracy of the communities revealed.

Comparison of the algorithms on two real-world networks is depicted in Fig. 3. The examples show that defensive propagation prefers networks with rather homogeneous distribution of the sizes of the communities, and that offensive propagation favors networks with more heterogeneous (e.g., power law) distribution. It should, however, be noted that both approaches can achieve superior performance on both of the networks. Still, on average, the defensive approach performs better on the social network *football* [6], while offensive outperforms defensive on the metabolic network *elegans* [37].

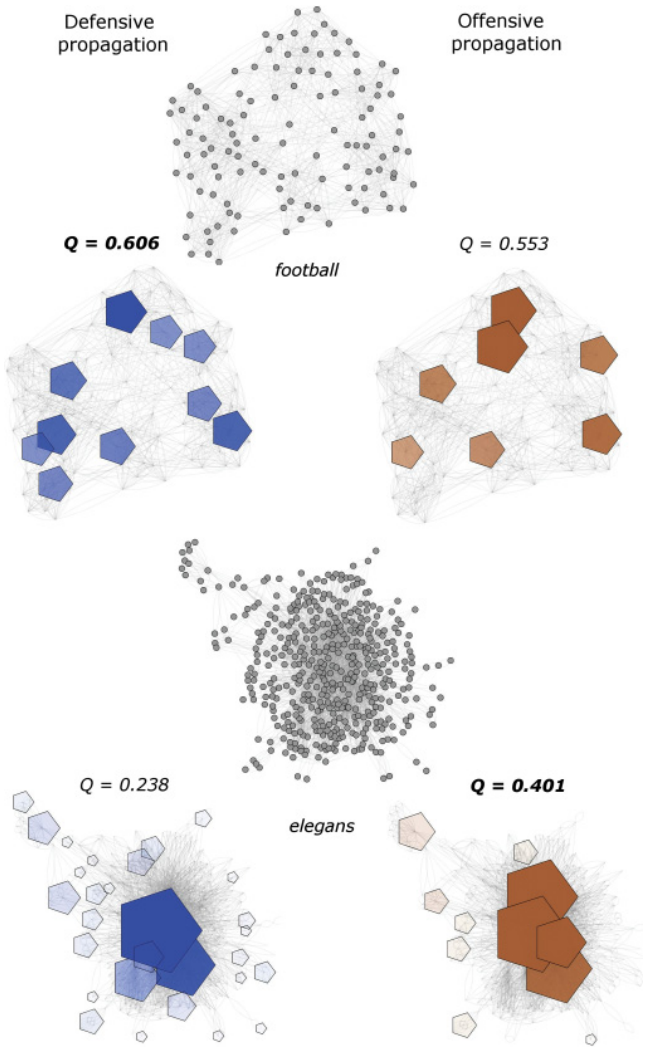


FIG. 3. (Color online) Comparison of defensive and offensive label propagation on two real-world networks, i.e., a social network of American football matches at a US college [6] and a metabolic network of the nematode *Caenorhabditis elegans* [37]. The revealed communities are shown with pentagonal nodes, and the sizes and intensities of colors (shadings) of the nodes are proportional to the sizes of communities. The networks comprise two relatively different community structures, considering the distribution of sizes of the communities. This is rather homogeneous in the case of *football* and (presumably) a power law in the case of *elegans*.

For an empirical analysis and further discussion of the algorithms, see Sec. IV; and for the pseudocode of the algorithms and discussion on some of the implementation issues, see Appendix B.

C. Diffusion and propagation algorithm

Defensive and offensive label propagation (Sec. III B) convey two unique strategies of community formation. An obvious improvement would be to combine the strategies, thus retaining the strong detection ability of the defensive approach and high accuracy of the offensive strategy. However, simply using the algorithms one after another does not attain the desired properties. The reason is that any label propagation algorithm, being run until convergence, finds local optimum (i.e., local equilibrium) that is hard to escape from.

Raghavan *et al.* [18] have already discussed the idea (however, in a different context) that label propagation could be improved if one had *a priori* knowledge about community cores. Core nodes could then be labeled with the same label, leaving all the other nodes labeled with a unique label. During the course of the algorithm, the (uniquely labeled) nodes would tend to adopt the label of their nearest attractor (i.e., the community core) and thus join its community. This would improve the algorithm's stability [18] and also the accuracy of the identified communities (Sec. IV).

The defensive and offensive label propagation algorithms are combined in the following manner. First, the defensive strategy is applied, to produce initial estimates of the communities and to accurately detect their cores. All border nodes of each community are then relabeled (labeled with a unique label), so that approximately one-half of the nodes retain their original label. Last, the offensive strategy is applied, which refines the community cores and accurately detects also their borders. Such combined strategy preserves advantages of both defensive and offensive label propagation algorithms and is denoted the basic diffusion and propagation algorithm (BDPA). Schematic representation of the algorithm is depicted in Fig. 4 (steps 3 and 4).

The core (and border) of each community is estimated by means of diffusion p_n (Sec. III B). As core nodes possess more intracommunity edges then border nodes, this results in higher values of p_n for core nodes. Thus, within the algorithm, the node n is relabeled due to the following rule:

$$c_n = \begin{cases} c_n & \text{for } p_n > m_{c_n}, \\ l_n & \text{for } p_n \leqslant m_{c_n} \end{cases}, \tag{9}$$

where m_{c_n} is the median of values p_n for nodes in community c_n, and l_n is a unique label. Thus, the core nodes retain their original labels, while all border nodes are relabeled. Note that all nodes with p_n equal to the median are also relabeled, to adequately treat smaller communities, where most of the nodes share the same value of p_n.

Empirical evaluation shows that BDPA significantly outperforms basic LPA and also the algorithm of Leung *et al.* [19] on smaller networks. However, when networks become larger, the hop attenuation strategy of Leung *et al.* [19] produces much larger communities, with higher values of modularity (on average).

Different authors have proposed approaches that detect communities in a hierarchical manner (e.g., [10]). The algorithm is first applied to the original network and initial communities are obtained. One then constructs the community network, where nodes represent communities and edges are added between them, while their nodes are connected in the original network. The algorithm is then recursively applied to the community network and the process repeats. At the end, the best communities found by the algorithm are reported (according to some measure).

The idea was also proposed in the context of label propagation [19]; however, the authors did not report any empirical results. We have analyzed the behavior of

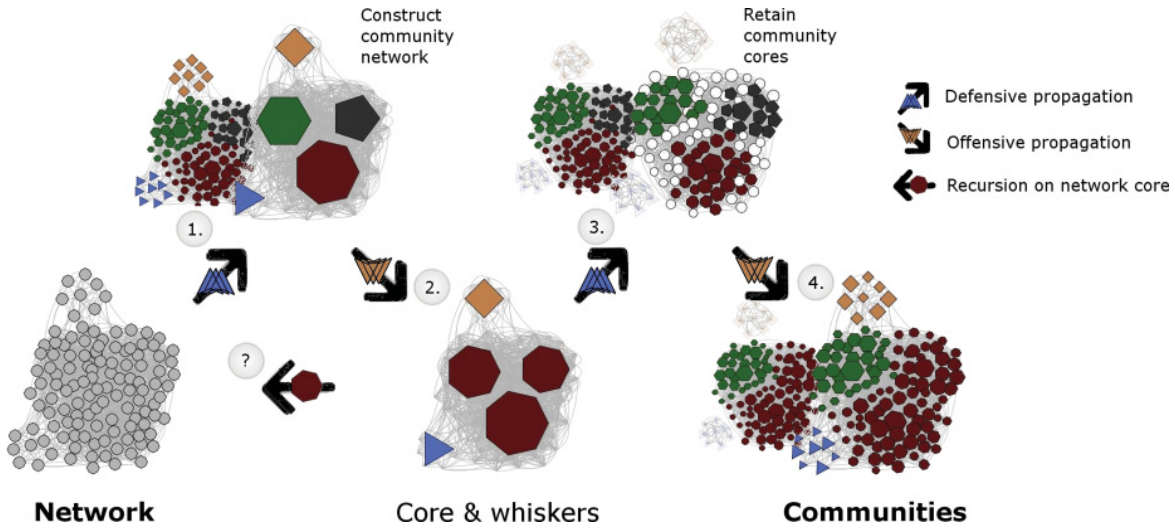


FIG. 4. (Color online) Diagram of (general) diffusion and propagation algorithm (DPA; the figure is merely a schematic representation of the algorithm and does not correspond to the actual result of the given network). The algorithm combines defensive and offensive label propagation in a hierarchical manner (steps 1 and 2) to extract the core of the network (red heptagon communities) and to identify whisker communities (blue triangle and orange square communities). Whiskers are retained as identified communities, while the algorithm is recursively applied to the core of the (community) network. The recursion continues until all of the nodes of the (current) network are classified into the same community (i.e., offensive propagation in step 2 flood-fills), when the basic diffusion and propagation algorithm (BDPA) is applied (steps 3 and 4). For more detailed discussion on the algorithms, see text.

hierarchical label propagation on real-world networks and also on benchmark networks with planted partition. The analysis has shown that on the second iteration (when the algorithm is first run on the community network), the label propagation (already) produces one major community or even *flood-fills* (all nodes are classified into the same community).

Although the analysis revealed undesirable behavior, we have observed that the major community commonly coincides with the core of the network, while other communities correspond to whisker communities. Leskovec *et al.* [3] have extensively analyzed large social and information networks and observed that (these) networks reveal clear core-periphery structure—most of the nodes are in the central core of the network, which does not have a clear community structure, whereas the best communities reside in the periphery (i.e., whiskers), which is only weakly connected with the core. For further discussion see Appendix A.

Based on the above observations, we propose the following algorithm denoted the (general) diffusion and propagation algorithm (DPA); a schematic representation of the algorithm is depicted in Fig. 4. First, defensive label propagation is applied to the original network (step 1), which produces a larger number of smaller communities that are used to construct the corresponding community network. Second, offensive label propagation is used on the constructed community network (step 2), to extract the core of the network (i.e., its major community) and to identify whisker communities (i.e., all other communities). The above procedure is then recursively applied only to the core of the (community) network, while the whisker communities are retained as identified communities. The recursion continues until the offensive propagation in step 2 flood-fills (i.e., the extracted core contains all of the nodes of the network analyzed), when the basic BDPA is applied (steps 3 and 4).

Empirical analysis on real-world networks shows that DPA outperforms all other label propagation algorithms (with comparable time complexity) and is comparable to current state-of-the-art community detection algorithms. Furthermore, the algorithm exhibits almost linear complexity (in the number of edges of the network) and scales even better than the basic LPA. It should also be noted that the application of the algorithm is not limited to networks that exhibit core-periphery structure.

For a thorough empirical analysis and further discussion on both presented algorithms, see Sec. IV; and for the pseudocode of the algorithms and discussion on some of the implementation issues, see Appendix B.

IV. EVALUATION AND DISCUSSION

The section presents the results of the empirical evaluation of the proposed algorithms.

The algorithms were first compared on two classes of benchmark networks with planted partition, namely, Girvan and Newman [6] and Lancichinetti *et al.* [27] benchmark networks. For the latter, we also varied the size of the networks (1000 and 5000 nodes) and the size of the communities (from 10 to 50 and from 20 to 100 nodes). The results are assessed in terms of normalized mutual information (NMI) [52] and are shown in Fig. 5.

Analysis clearly shows the difference between defensive and offensive propagation, especially on larger networks [Figs. 5(d) and 5(e)]. The offensive propagation (ODALPA) performs slightly better than the basic LPA and can still relatively accurately detect communities, while LPA already performs rather poorly [Fig. 5(d)]. On the other hand, the

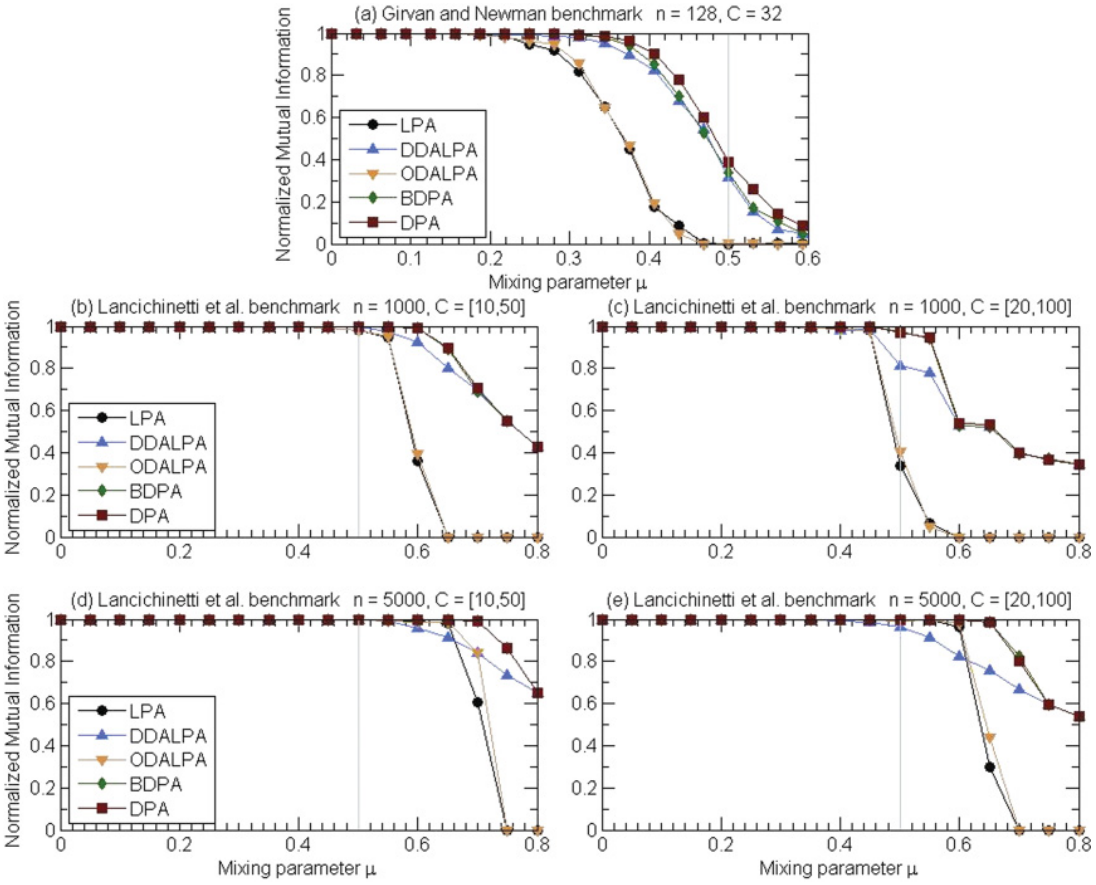


FIG. 5. (Color online) Comparison of the proposed algorithms on two classes of benchmark networks with planted partition, namely, Girvan and Newman [6] networks and four sets of Lancichinetti *et al*. [27] networks (the results are averages over 100 realizations). Network sizes equal 128, 1000, and 5000 nodes, and communities comprise up to 100 nodes. Straight (gray) lines at $\mu = 0.5$ denote the point beyond which the communities are no longer defined in the strong sense [13].

defensive propagation (DDALPA) does not detect communities as accurately as the other two approaches [Figs. 5(d) and 5(e)]; however, the algorithm still reveals the communities even when they are only weakly defined (and the other two approaches clearly fail). In other words, the defensive algorithm has high *recall*, whereas the offensive approach achieves high *precision*.

Furthermore, BDPA (and DPA) outperforms all three aforementioned algorithms. Note that the performance does not simply equal the upper hull of those for DDALPA and ODALPA. The analysis also shows that core extraction (i.e., DPA) does not improve the results on networks with thousands of nodes or less; the slight improvement on Girvan and Newman [6] benchmark networks results only from hierarchical investigation, and not core extraction. Nevertheless, as shown below, the results can be significantly improved on larger networks.

Lancichinetti and Fortunato [53] have conducted a thorough empirical analysis of more then 10 state-of-the-art community detection algorithms. To enable the comparison, the benchmark networks in Fig. 5 were selected so that they exactly coincide with those used in [53]. By comparing the results, we can conclude that DPA does indeed perform at least as good as the best algorithms analyzed in [53], namely, the hierarchical modularity optimization of Blondel *et al*. [10], the model selection approach of Rosvall and Bergstrom [16], the spectral algorithm proposed by Donetti and Munoz [15], and the multiresolution spin model of Ronhovde and Nussinov [20]. Moreover, on larger networks [Figs. 5(d) and 5(e)], DPA obtains even better results than all of the algorithms analyzed in [53]—for $\mu = 0.8$, none of the analyzed algorithms can obtain NMI above $\approx$0.35, while the values for DPA are 0.651 and 0.541, respectively.

DPA (and BDPA) was further analyzed on 23 real-world networks (Table I), ranging from networks with tens of nodes to networks with several tens of millions of edges [54]. To conduct a general analysis, we have considered a wide range of different types of real-world networks, in particular, social, communication, citation, collaboration, web, Internet, biological, and other networks (all networks were treated as unweighted and undirected). Because of the large number of networks considered, detailed description is omitted here.

The DPA algorithm was compared with all other proposed label propagation algorithms (to our knowledge), and with a greedy modularity optimization approach (Table I). The algorithms are as follows: LPA denotes basic label propagation [18], and LPAD denotes LPA with decreasing

TABLE I. Peak (maximal) modularities Q for various label propagation algorithms and a greedy optimization of modularity. The modularity for DPA for *elegans* was obtained with $\delta_{\max} = 1$ and for *asi* with $\delta_{\max} = 0$ (Appendix B); otherwise the values are 0.420 and 0.588, respectively. Values in italics correspond to the approaches that have significant time complexity compared to DPA.

Network	Description	*Nodes*	*Edges*	GMO	LPA	LPAD	LPAQ	LPAM	BDPA	DPA	No. CE[c]	T[c]
karate	Zachary's karate club [33]	34	78	0.381	0.416	0.402	0.399	**0.420**	**0.419**	**0.420**	0.02	
dolphins	Lusseau's bottlenose dolphins [38]	62	159		**0.529**	0.526	0.516	**0.529**	**0.528**	**0.529**	0.59	
books	Co-purchased political books [39]	105	441		**0.526**	0.519	0.522	**0.527**	**0.527**	**0.527**	0.46	
football	American football league [6]	115	616	0.556	**0.606**	**0.606**	0.604	**0.605**	**0.606**	**0.606**	0.37	
elegans	Metabolic network *C. elegans* [37]	453	2025	0.412	0.421	0.413	0.409	*0.452*	0.424	**0.427**[b]	0.17	
jazz	Jazz musicians [40]	198	2742	0.439	0.443	0.443	**0.445**	**0.445**	**0.444**	**0.444**	0.00	
netsci	Network scientists [9]	1589	2742		0.902	0.947			0.907	**0.960**	1.00	
yeast	Yeast protein interactions [41]	2114	4480		0.694	0.799			0.725	**0.824**	1.04	
emails	Emails within a university [42]	1133	5451	0.503	0.557	0.560	0.537	*0.582*	0.555	**0.562**	0.01	
power	Western US power grid [32]	4941	6594		0.612	0.804			0.668	**0.908**	1.14	
blogs	Weblogs on politics [43]	1490	16718		**0.426**	**0.426**			**0.426**	**0.426**	1.00	
pgp	*PGP* web of trust [44]	10680	24340	0.849	0.754	0.844	0.726	*0.884*	0.782	**0.869**	1.08	
asi	Autonomous syst. of Internet [25]	22963	48436		0.511	0.591			0.528	**0.600**[b]	1.02	0 s
codmat[3]	*Cond. Matt.* archive 2003[a] [45]	27519	116181	0.661	0.616	0.683	0.582	*0.755*	0.634	**0.735**	1.00	1.5 s
codmat[5]	*Cond. Matt.* archive 2005[a] [45]	36458	171736		0.586	0.643			0.608	**0.683**	1.00	
kdd[3]	*KDD-Cup* 2003 dataset [46]	27770	352285		0.624	**0.630**			0.619	0.617	1.00	3 s
nec	*nec* web overlay map [47]	75885	357317		0.693	0.738			0.703	**0.767**	1.03	
epinions	*Epinions* web of trust [48]	75879	508837		0.382	0.362			0.399	**0.402**	1.00	4.5 s
amazon[3]	*Amazon* co-purchasing 2003 [49]	262111	1.2M		0.682	0.749			0.701	**0.857**	1.01	20 s
ndedu	Webpages in nd.edu domain [50]	325729	1.5M		0.840	0.890			0.863	**0.903**	1.14	
google	Web graph of *Google* [3]	875713	4.3M		0.805	0.923			0.822	**0.968**	1.01	2.5 m
nber	*NBER* patents citations [51]	3.8M	16.5M		0.573	0.624			0.583	**0.759**	1.20	
live	*Live Journal* friendships [3]	4.8M	69.0M		0.538	0.539			0.557	**0.693**	1.00	44 m

[a]Reduced to the largest component of the original network.
[b]Obtained with slightly modified version of DPA (see caption).
[c]Average number of core extractions and computational times for DPA.

hop attenuation and node preference equal to the degree of the node [19] (Sec. II). The modularity optimization version of LPA is denoted LPAQ [11], and its refinement with multistep greedy merging LPAM [12]. Furthermore, GMO denotes greedy modularity optimization, proposed by Clauset *et al.* [8].

For each algorithm, we report peak (maximal) modularities obtained on the networks analyzed. Modularities for LPA, LPAD, BDPA, and DPA were obtained by running the algorithms from 2 to 100 000 times on each network (depending on the size of the network). On the other hand, peak modularities for LPAQ and LPAM (and also GMO) were reported by Liu and Murata [12].

The results show that DPA outperforms all other label propagation algorithms, except LPAM on networks of medium size (i.e., *elegans*, *emails*, *pgp*, and *codmat*[3]). However, further analysis reveals that on these networks, LPAM already has considerable time complexity compared to DPA. It should also be noted that modularities obtained by LPAM on three of these networks correspond to the highest modularity values ever reported in the literature. Similarly, peak modularities obtained by DPA (and some others) on smaller networks also equal the highest modularities ever published (to our knowledge, the modularity for *football* even slightly exceeds the highest value ever reported, i.e., 0.606, as opposed to 0.605). In summary, DPA obtains significantly higher values of modularity than other comparable label propagation approaches, especially on larger networks (with millions of nodes and edges).

As already discussed in Sec. III C, BDPA achieves superior results on smaller networks, better than LPA, LPAQ, and LPAD (and GMO). However, the algorithm is not appropriate for

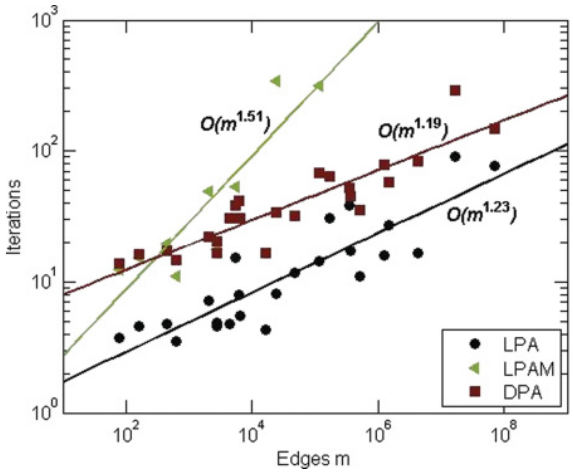


FIG. 6. (Color online) Time complexity of different label propagation algorithms estimated on real-world networks from Table I (results are averages over 100 iterations). From top to bottom, straight lines correspond to $0.83m^{0.51}$, $5.15m^{0.19}$, and $1.03m^{0.23}$, while the text denotes the overall time complexity of the algorithms (LPAM, DPA, and LPA, respectively). On a network with a billion edges, the (projected) number of iterations for DPA and LPA would equal 265 and 113, respectively.

TABLE II. Mean pairwise NMI of distinct community structures identified by different label propagation algorithms in 10 000 iterations (on selected set of networks from Table I).

Network	Nodes	Edges	LPA	BDPA	DPA
karate	34	78	0.574	0.578	**0.660**
dolphins	62	159	0.714	0.762	**0.774**
books	105	441	0.737	0.803	**0.805**
football	115	616	0.878	0.896	**0.897**
elegans	453	2025	0.610	0.615	**0.618**
jazz	198	2742	0.602	0.748	**0.808**

larger networks, where hierarchical core extraction prevails (i.e., DPA).

We have also analyzed the number of core extractions (Sec. III C) made by DPA on these networks (Table I). Core extraction does not gain on networks with less than thousands of nodes or edges, where the average number is commonly close to 0. However, when networks become larger, a (single) core extraction produces a significant gain in modularity (on these networks). Interestingly, even on a network with several millions of nodes and several tens of millions of edges (i.e., *live*), the number of extractions is still 1 (on average).

Next, we have thoroughly compared the time complexity of a simple LPA and DPA (and also LPAM [12]). On each iteration of the algorithms, each edge of the network is visited (at most) twice. Thus the time complexity of a single iteration equals $O(m)$, with m being the number of edges. The complexity for DPA is even lower, after the core has been extracted; however, due to simplicity, we consider each iteration to have complexity $O(m)$.

Iterative algorithms (such as label propagation) are commonly assessed only on smaller networks, where the number of iterations can be bounded by a small constant. In this context, both LPA and DPA exhibit nearly linear complexity, $O(m)$. However, on networks with thousands or millions of nodes and edges, this "constant" indeed increases—even for simple LPA, which is known for its speed, the number of iterations notably increases on larger networks. We have thus analyzed the total number of iterations made by the algorithms on real-world networks (Table I). The results are shown in Fig. 6 (the number of edges m is chosen to represent the size of the network). Note that the number of iterations for DPA corresponds to the sum of the iterations made by all of the algorithms run within (i.e., DDALPA, ODALPA, and BDPA).

As discussed earlier, DPA (and LPA) scales much better than LPAM—the average number of iterations on a network with tens of millions of edges is 147 and 78 for DPA and LPA, respectively, while LPAM already exceeds 300 iterations on networks with tens of thousands of edges. Furthermore, results also show that DPA scales even better than simple LPA [i.e., $O(m^{1.19})$, as opposed to $O(m^{1.23})$]; however, it is outperformed by LPA because of a larger constant. Nevertheless, the analysis shows promising results for future analyses of large complex networks.

In the context of analyzing large networks, it should be mentioned that by far the fastest convergence is obtained by using the defensive propagation algorithm DDALPA (Sec. III B). On the largest of the networks (i.e., *live*), the algorithm converges in only 25 iterations (3 times faster than LPA); still, the modularity of the revealed community structure is only 0.470.

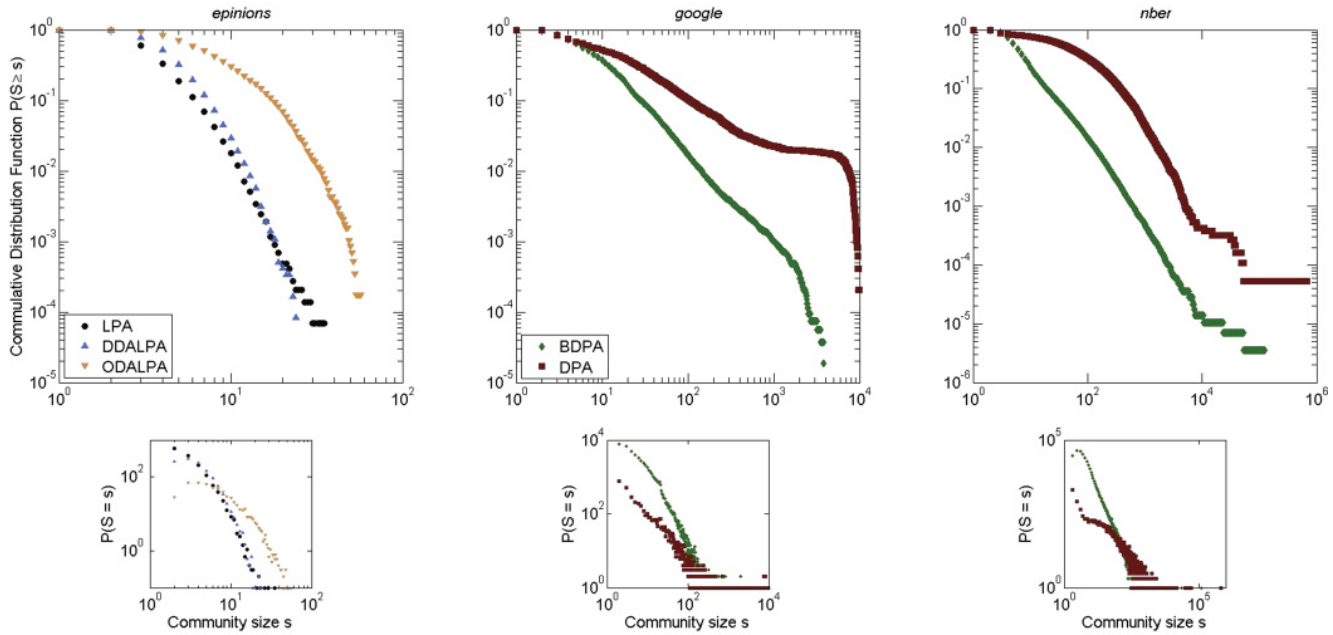


FIG. 7. (Color online) (Cumulative) distributions of the community sizes for three real-world networks from Table I (for the *epinions* network, the results were averaged over 10 runs). Note some particularly large communities revealed by DPA in the case of the *google* and *nber* networks (with around 10^4 and 10^6 nodes, respectively). Interestingly, these coincide with the low-conductance [55] communities reported in [3].

Last, we have also studied the stability of DPA (and BDPA), and compared it with simple LPA. The latter is known to find a large number of distinct community structures in each network [12,18,23], while Tibély and Kertész [23] have argued that these structures are relatively different among themselves. Indeed, on the *zachary* network LPA revealed 628 different community structures (in 10 000 iterations), while this number equals 159 and 124 for BDPA and DPA, respectively. However, as the number of distinct communities can be misleading, we have rather directly compared the identified community structures.

In Table II we show mean pairwise NMI of (distinct) community structures that were identified by the algorithms on a selected set of real-world networks. DPA (and BDPA) appears to be more stable than LPA; moreover, the identified community structures are relatively similar for all of the algorithms considered (in most networks analyzed). Interestingly, the results also seem to correlate well with the modularities shown in Table I—the clearer the community structure of the network, the more stable the algorithms appear. Nevertheless, as indicated by various previous authors [18,23], the number of different community structures can be very high, especially in larger networks (e.g., 1116 and 1330 for DPA applied to the *football* and *jazz* networks, respectively).

(Cumulative) distributions of sizes of communities, obtained with the proposed algorithms on three real-world networks, are shown in Fig. 7.

V. CONCLUSION

This paper proposes an advanced label propagation community detection algorithm that combines two unique strategies of community formation. The algorithm analyzes the network in a hierarchical manner that recursively extracts the core of the network and identifies whisker communities. The algorithm employs only local measures for community detection and does not require the number of communities to be specified beforehand. The proposition was rigorously analyzed on benchmark networks with planted partition and on a wide range of real-world networks, with up to several millions of nodes and tens of millions of edges. The performance of the algorithm is comparable to the current state-of-the-art community detection algorithms; moreover, the algorithm exhibits almost linear time complexity (in the number of edges of the network) and scales even better than the basic label propagation algorithm. The proposal thus gives prominent grounds for future analysis of large complex networks.

This work also provides further understanding on the dynamics of label propagation, in particular, on how different propagation strategies can alter the dynamics of the process and reveal community structures with unique properties.

ACKNOWLEDGMENTS

The authors wish to thank (anonymous) reviewers for comments and criticisms that helped improve the paper. This work has been supported by the Slovene Research Agency ARRS within Research Program No. P2-0359.

APPENDIX A: CORE-PERIPHERY STRUCTURE

Leskovec *et al.* [3] have conducted an extensive analysis of large social and information—and some other—networks. They have observed that these networks can be clearly divided into the central core and remaining periphery (i.e., the core-periphery structure). The periphery is constituted of many small, well-defined communities (in terms of conductance [55]) that are only weakly connected to the rest of the network. When they are connected by a single edge, they are called whiskers (or 1-whiskers). On the other hand, the core of the network consists of larger communities that are well connected, and thus only loosely defined in the sense of communities. Their analysis has thus shown that the best communities (according to conductance) reside in the periphery of these networks (i.e., whiskers) and have a characteristic size of around 100 nodes. For further discussion, see [3,56].

APPENDIX B: ALGORITHMS

In this section we give the pseudocode of all the algorithms proposed in this paper (Figs. 8–10) and discuss some of the implementation issues.

Because of the nature of label propagation, it may be that when the algorithm converges, two (disconnected) communities share the same label. This happens when a node propagates its label in two direction but is itself relabeled in the later stages of the algorithm. Nevertheless, disconnected communities can be detected at the end using a simple breath-first search.

Each run of BDPA or DPA (Figs. 9 and 10) unfolds several sets of communities and the best are returned at the end

Input: Graph $G(N, E)$ with weights W
Output: Communities C (i.e. node labels)
1: $\delta \leftarrow 0$
2: **for** $n \in N$ **do**
3: $c_n \leftarrow l_n$ {Unique label.}
4: $d_n \leftarrow 0$
5: $p_n \leftarrow 1/|N|$
6: **end for**
7: *shuffle*(N)
8: **while** *not converged* **do**
9: **for** $n \in N$ **do**
10: $c_n \leftarrow \text{argmax}_l \sum_{i \in \mathcal{N}^l(n)} p_i(1 - \delta d_i) w_{ni}$
11: **if** c_n *has changed* **then**
12: $d_n \leftarrow (\min_{i \in \mathcal{N}^{c_n}(n)} d_i) + 1$
13: $p_n \leftarrow \sum_{i \in \mathcal{N}^{c_n}(n)} p_i / k_i^{c_n}$
14: **end if**
15: **end for**
16: $\delta \leftarrow$ *proportion of labels changed*
17: **if** $\delta \geq \delta_{max}$ **then**
18: $\{\delta_{max}$ is fixed to $\frac{1}{2}.\}$
19: $\delta \leftarrow 0$
20: **end if**
21: **end while**
22: **return** C

FIG. 8. Defensive label propagation algorithm with (dynamic) hop attenuation (DDALPA). In the offensive version (ODALPA), the node preference p_i is replaced by $1 - p_i$ (line 10) and the degree $k_i^{c_n}$ is replaced by k_i (line 13).

Input: Graph $G(N, E)$ with weights W
Output: Communities C (i.e. node labels)
$C \leftarrow DDALPA(G, W)$
for $c \in C$ **do**
{Retain community cores.}
$m_c \leftarrow median(\{p_n | \ n \in N \wedge c_n = c\})$
for $n \in N \wedge c_n = c \wedge p_n \leq m_c$ **do**
$c_n \leftarrow l_n$ {Unique label.}
$d_n \leftarrow 0$
$p_n \leftarrow 0$ {Maximal preference.}
end for
end for
$C \leftarrow ODALPA(G, W)$
return C {Returns best communities.}

FIG. 9. Basic diffusion and propagation algorithm (BDPA).

(according to some measure of goodness of communities). For the analysis in Sec. IV, the algorithms reported community structures that obtained the highest modularity (computed on the original network). Thus, the results might be attributed to modularity's resolution limit problem [57] or other limitations [58]; still, this is not a direct artifact of the algorithms.

Input: Graph $G(N, E)$ with weights W
Output: Communities C (i.e. node labels)
$C \leftarrow DDALPA(G, W)$
$C_C \leftarrow ODALPA(G_C, W_C)$
if C_C *contains one community* **then**
$C \leftarrow BDPA(G, W)$
else
{Recursion on core c in C_C.}
$C \leftarrow (C_C - \{c\}) \cup DPA(G_C(c), W_C(c))$
end if
return C {Returns best communities.}

FIG. 10. Diffusion and propagation algorithm (DPA).

An additional note should be made for the offensive propagation algorithm ODALPA (Fig. 8). When used on networks with several thousands of nodes or less, diffusion values p_n should only be updated (line 13) after the first iteration; otherwise the algorithm might not converge. The reason is that during the first iteration, communities are still rather small (because of the size of the network) and thus all of the nodes lie in the border of the communities. Hence, updating the diffusion values results in applying propagation preference to all of the nodes.

[1] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, Nature (London) **435**, 814 (2005).
[2] E. Ravasz and A. L. Barabási, Phys. Rev. E **67**, 026112 (2003).
[3] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, Internet Mathematics **6**, 29 (2009).
[4] M. E. J. Newman and E. A. Leicht, Proc. Nat. Acad. Sci. USA **104**, 9564 (2007).
[5] S. Pinkert, J. Schultz, and J. Reichardt, PLoS Comput. Biol. **6**, e1000659 (2010).
[6] M. Girvan and M. E. J. Newman, Proc. Nat. Acad. Sci. USA **99**, 7821 (2002).
[7] M. E. J. Newman and M. Girvan, Phys. Rev. E **69**, 026113 (2004).
[8] A. Clauset, M. E. J. Newman, and C. Moore, Phys. Rev. E **70**, 066111 (2004).
[9] M. E. J. Newman, Phys. Rev. E **74**, 036104 (2006).
[10] V. D. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre, J. Stat. Mech. (2008) P10008.
[11] M. J. Barber and J. W. Clark, Phys. Rev. E **80**, 026129 (2009).
[12] X. Liu and T. Murata, Physica A **389**, 1493 (2009).
[13] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, Proc. Natl. Acad. Sci. USA **101**, 2658 (2004).
[14] C. Pang, F. Shao, R. Sun, and S. Li, in *Proceedings of the International Symposium on Neural Networks*, edited by W. Yu, H. He, and N. Zhang (Springer, Wuhan, 2009), pp. 839–846.
[15] L. Donetti and M. A. Muñoz, J. Stat. Mech. (2004) P10012.
[16] M. Rosvall and C. T. Bergstrom, Proc. Natl. Acad. Sci. USA **105**, 1118 (2008).
[17] A. Firat, S. Chatterjee, and M. Yilmaz, Comput. Stat. Data Anal. **51**, 6285 (2007).
[18] U. N. Raghavan, R. Albert, and S. Kumara, Phys. Rev. E **76**, 036106 (2007).
[19] I. X. Y. Leung, P. Hui, P. Liò, and J. Crowcroft, Phys. Rev. E **79**, 066107 (2009).
[20] P. Ronhovde and Z. Nussinov, Phys. Rev. E **81**, 046114 (2010).
[21] X. Liu and T. Murata, in *Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, Vol. 1 (IEEE, Washington, DC, 2009), pp. 50–57.
[22] S. Fortunato, Phys. Rep. **486**, 75 (2010).
[23] G. Tibély and J. Kertész, Physica A **387**, 4982 (2008).
[24] P. Schuetz and A. Caflisch, Phys. Rev. E **77**, 046112 (2008).
[25] M. E. J. Newman, [http://www-personal.umich.edu/~mejn/netdata/].
[26] Throughout the article we refer to core and whiskers of the network as being the result of the algorithm, although this might not necessarily coincide with the analysis of Leskovec *et al.* [9].
[27] A. Lancichinetti, S. Fortunato, and F. Radicchi, Phys. Rev. E **78**, 046110 (2008).
[28] The conclusion naturally depends on the definition of research communities, which are, in this case, communities revealed by the algorithm.
[29] S. Gregory, e-print arXiv:0910.5516.
[30] L. Freeman, Sociometry **40**, 35 (1977).
[31] L. C. Freeman, Soc. Networks **1**, 215 (1979).
[32] D. J. Watts and S. H. Strogatz, Nature (London) **393**, 440 (1998).
[33] W. W. Zachary, J. Anthropol. Res. **33**, 452 (1977).
[34] The estimation could also be done by using the label propagation itself; however, we believe that using the floating-point counterpart of the approach would produce more accurate results in practice.

[35] S. Brin and L. Page, Comput. Networks ISDN **30**, 107 (1998).
[36] J. M. Kleinberg, J. ACM **46**, 604 (1999).
[37] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A. Barabási, Nature (London) **407**, 651 (2000).
[38] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, Behav. Ecol. Sociobiol. **54**, 396 (2003).
[39] V. Krebs, [http://www.orgnet.com/].
[40] P. Gleiser and L. Danon, Adv. Complex Syst. **6**, 565 (2003).
[41] H. Jeong, S. P. Mason, A. Barabási, and Z. N. Oltvai, Nature (London) **411**, 41 (2001).
[42] R. Guimerà, L. Danon, A. Díaz-Guilera, F. Giralt, and A. Arenas, Phys. Rev. E **68**, 065103 (2003).
[43] L. A. Adamic and N. Glance, in *Proceedings of the International Workshop on Link Discovery*, edited by J. Adibi, M. Grobelnik, D. Mladenic, and P. Pantel (ACM, New York, 2005), pp. 36–43.
[44] M. Boguná, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas, Phys. Rev. E **70**, 056122 (2004).
[45] M. E. J. Newman, Proc. Natl. Acad. Sci. USA **98**, 404 (2001).
[46] See [http://www.sigkdd.org/kddcup/].
[47] M. Hoerdt, M. Jaeger, A. James, D. Magoni, J. Maillard, D. Malka, and P. Merindol, [http://www.labri.fr/perso/magoni/nec/].
[48] M. Richardson, R. Agrawal, and P. Domingos, in *Proceedings of the International Semantic Web Conference*, edited by D. Fensel, K. P. Sycara, and J. Mylopoulos, Vol. 2 (Springer, Berlin, 2003), pp. 351–368.
[49] J. Leskovec, L. A. Adamic, and B. A. Huberman, ACM Trans. Web **1**, (2007).
[50] R. Albert, H. Jeong, and A. Barabási, Nature (London) **401**, 130 (1999).
[51] B. H. Hall, A. B. Jaffe, and M. Tratjenberg, *The NBER Patent Citation Data File: Lessons, Insights and Methodological Tools*, Tech. Rep. (National Bureau of Economic Research, 2001).
[52] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas, J. Stat. Mech. (**2005**) P09008.
[53] A. Lancichinetti and S. Fortunato, Phys. Rev. E **80**, 056117 (2009).
[54] The analysis on networks with hundreds of millions or even billions of edges was bounded due to limited memory resources.
[55] B. Bollobás, *Modern Graph Theory* (Springer-Verlag, Berlin, 1998).
[56] J. Leskovec, K. J. Lang, and M. W. Mahoney, in *Proceedings of the ACM International Conference on World Wide Web*, edited by M. Rappa, P. Jones, J. Freire, and S. Chakrabarti (ACM, New York, 2010), pp. 631–640.
[57] S. Fortunato and M. Barthelemy, Proc. Natl. Acad. Sci. USA **104**, 36 (2007).
[58] B. H. Good, Y. A. de Montjoye, and A. Clauset, Phys. Rev. E **81**, 046106 (2010).

5

Odkrivanje skupin vozlišč s posplošeno izmenjavo oznak

V pričujočem poglavju je vloženo delo z naslovom "Odkrivanje skupin vozlišč s posplošeno izmenjavo oznak" [72] (angl. "Ubiquitousness of link-density and link-pattern communities in real-world networks"). Delo predstavi različne metode in tehnike odkrivanja karakterističnih skupin vozlišč v omrežjih na osnovi izmenjave oznak, ki temeljijo na analogiji med različnimi skupinami vozlišč ter njihovimi lastnostmi. Le-te omogočajo odkrivanje splošnejših skupin kot osnovni pristop (t.j. skupnosti in modulov), pri tem pa ne zahtevajo predhodnega znanja o omrežju (npr. število ali vrsto skupin). Predlagane posplošitve pristopa so preizkušene na sintetičnih in realnih omrežjih z znano zgradbo ter primerjane s pristopi v literaturi.

Predstavljeno delo je v razširjeni obliki objavljeno v [72, 73, 139].[4]

[4]Privzeta omrežja so dostopna na http://lovro.lpt.fri.uni-lj.si/.

Eur. Phys. J. B (2012) 85: 32
DOI: 10.1140/epjb/e2011-20448-7

THE EUROPEAN PHYSICAL JOURNAL B

Regular Article

Ubiquitousness of link-density and link-pattern communities in real-world networks

L. Šubelj[a] and M. Bajec

University of Ljubljana, Faculty of Computer and Information Science, 1000 Ljubljana, Slovenia

Received 9 June 2011 / Received in final form 24 October 2011
Published online 18 January 2012 – © EDP Sciences, Società Italiana di Fisica, Springer-Verlag 2012

Abstract. Community structure appears to be an intrinsic property of many complex real-world networks. However, recent work shows that real-world networks reveal even more sophisticated modules than classical cohesive (link-density) communities. In particular, networks can also be naturally partitioned according to similar patterns of connectedness among the nodes, revealing link-pattern communities. We here propose a propagation based algorithm that can extract both link-density and link-pattern communities, without any prior knowledge of the true structure. The algorithm was first validated on different classes of synthetic benchmark networks with community structure, and also on random networks. We have further applied the algorithm to different social, information, technological and biological networks, where it indeed reveals meaningful (composites of) link-density and link-pattern communities. The results thus seem to imply that, similarly as link-density counterparts, link-pattern communities appear ubiquitous in nature and design.

1 Introduction

Complex real-world networks commonly reveal local cohesive modules of nodes denoted (*link-density*) *communities* [1]. These are most frequently observed as densely connected clusters of nodes that are only loosely connected between. Communities possibly play crucial roles in different real-world systems [2,3]; furthermore, community structure also has a strong impact on dynamic processes taking place on networks [4,5]. Thus, communities provide an insight into not only structural organization but also functional behavior of various real-world systems [3,6–8].

Consequently, analysis of community structure is currently considered one of the most prominent areas of network science [9–11], while it has also been the focus of recent efforts in a wide variety of other fields. Besides providing many significant theoretical grounds [8], a substantial number of different community detection algorithms has also been proposed in the literature [12–20] (for reviews see [10,11,21]). However, most of the past research was focused primarily on classical communities characterized by higher density of edges [22]. In contrast to the latter, some recent work demonstrates that real-world networks reveal even more sophisticated communities [23–26] that are indistinguishable under classical frameworks.

Networks can also be naturally partitioned according to similar patterns of connectedness among the nodes, revealing *link-pattern communities* [23,24]. (The term was formulated by Long et al. [24]). Loosely speaking, link-pattern communities correspond to clusters of nodes that are similarly connected with the rest of the network (i.e., share common neighborhoods). Note that link-density communities are in fact a special case of link-pattern communities (with some fundamental differences discussed later on). Thus, some of the research on the former also apply for the latter [13,14,28–30]. However, contrary to the flourish of the literature on classical communities in the last decade, a relatively small number of authors have considered more general link-pattern counterparts [23–26,31–37] (in the same sense as in this paper[1]). Although this could be attributed to a number of factors like increased complexity or lack of adequate generative models and algorithms, more importantly, existence of meaningful link-pattern communities has not been properly verified under the same framework in various different types of real-world networks that are commonly analyzed in the literature (still, some networks have been considered in the past). In this paper we try to address this issue. (Note that similar stance was also made by Newman and Leicht [23]).

We extend balanced propagation [38] with defensive preservation of communities [20] into a general approach that can extract arbitrary network modules ranging from link-density to link-pattern communities. To the best of our knowledge, this is the only such algorithm that does not require some prior knowledge of the true structure (e.g., the number of communities), or does not optimize

[a] e-mail: lovro.subelj@fri.uni-lj.si

[1] Link-pattern communities are known as blockmodels [27] in social networks literature. These were rigorously analyzed in the past, however, the main focus and employed formulation differs from ours.

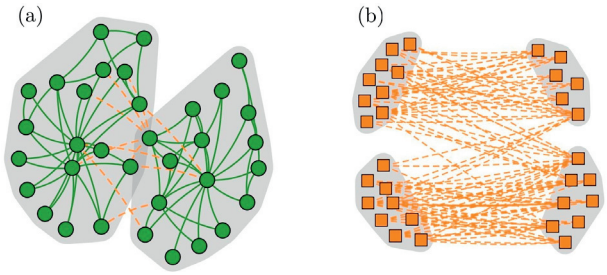


Fig. 1. (Color online) Link-density and link-pattern communities (i.e., shaded regions) in (a) *karate* and (b) *women* social networks, respectively (Tab. 2). The former represents social interactions among the members of karate club observed by Zachary [39], while the latter shows social events (right-hand side) visited by women (left-hand side) in Natchez, Mississippi [40]. Link-density communities correspond to cohesive modules of nodes, whereas link-pattern communities represent common patterns of connectedness among the nodes.

some heuristic selected beforehand. We have validated the proposed algorithm on two classes of synthetic benchmark networks with community structure, and also on random networks. The algorithm was further applied to different social, information, technological and biological networks, where it reveals meaningful composites of link-density and link-pattern communities that are well supported by the network topology. The results thus seem to imply that, similarly as link-density counterparts, link-pattern communities appear ubiquitous in nature and technology.

The rest of the paper is structured as follows. In Section 2 we discuss the relation between link-density and link-pattern communities in greater detail, and propose a propagation based algorithm for their detection. Results on synthetic and real-world networks are presented and formally discussed in Section 3, while in Section 4 we summarize our main observations and discuss some prominent directions for future research.

2 Link-density and link-pattern communities

Although classical link-density communities can be considered under the same framework as link-pattern communities, there exist some significant differences between the two (Fig. 1). Most obviously, link-pattern communities do not correspond to cohesive modules of nodes, whereas, such communities commonly do not even feature connectedness. Connectedness is considered a fundamental structural property of link-density communities, and thus a common ingredient of different objective functions and community detection algorithms [10].

While link-density communities are often related to the notions of assortative mixing by degree and homophily [1,41] (at least in social networks), link-pattern communities might in fact represent the origin of more commonly observed disassortative degree mixing [42,43]. As the latter has been analyzed to much lesser extent than the former [23], direct dependence has not yet been verified in real-world networks. Nevertheless, disassortative mixing refers merely to the phenomena that nodes mainly connect to dissimilar nodes, and thus outside of their respective (link-pattern) community. However, how such communities relate between each other, and with the rest of the network, remains unexplained. (Although network assortativity was most commonly analyzed in the context of node degree [41,42], we refer to the notion in general).

Note also that, as nodes of some link-pattern community are commonly not directly connected, they exhibit somewhat higher mutual independence than nodes within some link-density community. On the contrary, nodes from neighboring link-pattern communities are somewhat more dependent than in the case of classical communities.

Due to all above, we strictly distinguish between link-density and link-pattern communities within the proposed algorithm. However, it ought to be mentioned that this is rather different from other authors, who have typically considered all communities under link-pattern regime [23–26,31–33]. Nevertheless, the latter could be attributed to the fact that other approaches are mainly based on previous work in social sciences, statistics or artificial intelligence, where such setting might be more adequate.

In Section 2.1 we first introduce a balanced propagation based algorithm for classical community detection; while the algorithm is extended for general community detection in Section 2.2.

2.1 Classical community detection

Let the network be represented by an undirected and unweighted multi-graph $G(N, E)$, with N being the set of nodes of the graph and E being the set of edges. Furthermore, let c_n be the community (label) of node n, $n \in N$, and $\mathcal{N}(n)$ the set of its neighbors.

Algorithms presented below are in fact based on a label propagation proposed by Raghavan et al. [16]. The label propagation algorithm (LPA) [16] extracts (link-density) communities by exploiting the following simple procedure. At first, each node is labeled with a unique label, $c_n = l_n$. Then, at each iteration, each node adopts the label shared by most of its neighbors. Hence,

$$c_n = \underset{l}{\operatorname{argmax}} |\mathcal{N}^l(n)|, \tag{1}$$

where $\mathcal{N}^l(n)$ is the set of neighbors of n that share label l (ties are broken uniformly at random). To prevent oscillations of labels, node n retains its current label when it is among most frequent in $\mathcal{N}(n)$ [16]. Due to existence of many intra-community edges, relative to the number of inter-community edges, nodes in a (link-density) community form a consensus on some particular label after a few iterations. Thus, when an equilibrium is reached, disconnected groups of nodes sharing the same label are classified into the same community.

Due to extremely fast structural inference of label propagation, the algorithm exhibits near linear time complexity [16,20] (in the number of edges) and can easily

scale to networks with millions (or even billions) of nodes and edges [20,44]. Also, due to its algorithmic simplicity, it is currently among more commonly adopted algorithms in the literature. Still, label propagation can be further improved in various ways [20,38,45–48].

In the following we present two advances of the basic approach that improve on its robustness and community detection strength. Both result in a simple incorporation of node preferences p_n [46] into the updating rule of label propagation as

$$c_n = \underset{l}{\operatorname{argmax}} \sum_{m \in \mathcal{N}^l(n)} p_m, \tag{2}$$

(see Eq. (6)). Node preferences adjust the propagation strength of each respective node, and can thus direct the propagation process towards more desirable partitions [20,46]. Note that preferences p_n can be set to an arbitrary node statistic (e.g., degree [46]).

To address issues with oscillations of labels in some networks (e.g., bipartite networks), nodes' labels are updated in a random order [16] (independently among iterations). Although this solves the aforementioned problem, the introduction of randomness severely hampers the robustness of the algorithm, and consequently also the stability of the identified community structure. Different authors have noted that label propagation reveals a large number of different community structures even in smaller networks, while these structures are also relatively different among themselves [20,49].

We have previously shown that updating nodes in some particular order results in higher propagation strength for nodes that are updated at the beginning, and lower propagation strength for nodes that are updated towards the end [38]. The order of node updates thus governs the algorithm in a similar manner than corresponding node propagation preferences. Based on the latter, we have proposed a balanced propagation algorithm [38] that utilizes node preferences to counteract (i.e., balance) the randomness introduced by random update orders. In particular, we introduce the notion of node balancers that are set to the reverse order in which the nodes are assessed. Thus, lower and higher propagation strength is assigned to nodes considered first and last, respectively.

Let nodes N be ordered in some random way, and let i_n denote the normalized position of node n in this order, $i_n \in (0, 1]$. Hence,

$$i_n = \frac{\text{index of node } n}{|N|}. \tag{3}$$

Node balancers b_n can then be modeled with a simple linear function as $b_n = i_n$. However, using a logistic curve allows for some further control over the algorithm. Thus,

$$b_n = \frac{1}{1 + \exp(-\beta(i_n - \alpha))}, \tag{4}$$

where α and β are parameters of the algorithm. Intuitively, we fix α to 0.5, while stability parameter β is set to 0.25 according to some preliminary experiments [38] (see below). Note that balancers b_n are re-estimated before each iteration, and are incorporated into the algorithm as node propagation preferences (see Eq. (6)).

Setting the stability parameter β to 0 yields the basic label propagation approach, while increasing β significantly improves the robustness of the algorithm. However, computational complexity thus also increases. Hence, balanced propagation improves the stability of the identified community structure for the sake of higher complexity, while the trade-off is in fact governed by the parameter β. Note that community detection strength of the refined algorithm is also improved in most cases. For a more detailed discussion see [38].

To even further improve the performance of the algorithm we also adopt defensive preservation of communities [20]. The strategy increases the propagation strength from the core of each currently forming community, which results in an immense ability of detecting communities, even when they are only weakly depicted in the network's topology. Laying the pressure from the borders also prevents a single community from occupying a large portion of the network, which else occurs in, e.g., information networks [46]. Thus, the strategy defensively preserves communities and forces the algorithm to more gradually reveal the final structure. For further discussion see [20,44].

In the algorithm, community cores are estimated by means of the diffusion over the network. The latter is modeled by employing a random walker within each community. Let d_n be the probability that a random walker utilized on community c_n visits node n. Then,

$$d_n = \sum_{m \in \mathcal{N}^{c_n}(n)} \frac{d_m}{k_m^{c_n}}, \tag{5}$$

where $k_m^{c_n}$ is the intra-community degree of node m. Besides deriving the estimates of cores and borders, the main objection here is to mimic label propagation within each community, to estimate the current state of the propagation process, and then to adequately alter its dynamics (see Eq. (6)). Note that values d_n are re-estimated according to equation (5) when the corresponding node updates its label (initially, all d_n are set to $1/|N|$).

Similarly as above, diffusion values d_n are incorporated into the algorithm as node propagation preferences. Thus, the updating rule for balanced propagation algorithm with defensive preservation of communities is

$$c_n = \underset{l}{\operatorname{argmax}} \sum_{m \in \mathcal{N}^l(n)} b_m d_m. \tag{6}$$

The above is taken as a basis for a general community detection algorithm presented in the following section. Note that the formulation can be extended to weighted networks in a straightforward fashion.

2.2 General community detection

Label propagation algorithm (and its advances) cannot be directly adopted for detection of link-pattern communities, as the bare nature of label propagation demands

cohesive (connected) clusters of nodes (Sect. 2). However, link-pattern communities can still be seen as cohesive modules when one considers second order neighborhoods (i.e., nodes at distance 2). Thus, instead of propagating labels between the neighboring nodes, the labels are rather propagated through node's neighbors (i.e., between nodes at distance 2). For instance, when a group of nodes exhibits similar pattern of connectedness with other nodes, propagating labels through these latter nodes would indeed reveal the respective link-pattern community (similarly as for classical label propagation).

Considering the above, balanced propagation based algorithm presented in Section 2.1 can be extended for link-pattern communities in a rather *ad hoc* fashion. Let δ_l be a community dependent parameter, $\delta_l \in [0, 1]$, such that $\delta_l \approx 1$ and $\delta_l \approx 0$ for link-density and link-pattern communities, respectively. Thus, when δ_l varies from 1 to 0, communities range from classical link-density communities to proper link-pattern communities. Balanced propagation in equation (6) can then be simply advanced into a general community detection algorithm as

$$c_n = \underset{l}{\operatorname{argmax}} \left(\delta_l \sum_{m \in \mathcal{N}^l(n)} b_m d_m + (1 - \delta_l) \sum_{m \in \mathcal{N}^l(s) | s \in \mathcal{N}(n)} b_m \frac{d'_m}{k_s} \right), \tag{7}$$

where similarly as in equation (5), diffusion values d'_n are estimated using random walks. Hence,

$$d'_n = \sum_{m \in \mathcal{N}^{c_n}(s) | s \in \mathcal{N}(n)} \frac{d'_m}{\sum_{s \in \mathcal{N}(m)} k_s^{c_n}}. \tag{8}$$

(Denominators in Eqs. (7), (8) provide that the sums are proportional to the degree of the node k_n). Else, the proposed algorithm is identical as before, and is denoted general propagation algorithm (GPA). Note that setting all δ_l to 1 yields the classical community detection algorithm in equation (6).

Due to simplicity, in GPA all δ_l are fixed to 0.5. Still, the algorithm can detect either link-density or link-pattern communities, or different mixtures of both, when they are clearly depicted in the network's topology (Sect. 3). However, the algorithm can also detect communities that are of neither link-density nor link-pattern type.

As our main intention is to unfold meaningful composites of mainly link-density and link-pattern communities, we also propose a variant of the algorithm denoted $\mathrm{GPA_C}$. The latter algorithm re-estimates the values δ_l on each iteration, in order to reveal clearer community structure. In particular, we measure the quality of each community using the conductance Φ [50], to determine whether the community better conforms with link-density or link-pattern regime. (The conductance measures the goodness of a link-density community, or equivalently, the quality of the corresponding network cut). As good link-density communities exhibit low values of conductance, and good link-pattern communities exhibit high values, after each iteration of the algorithm (though omitted on first) we set δ_l according to

$$\delta_l = 1 - \Phi(l) = \frac{1}{k^l} \sum_{n \in N^l} k_n^l, \tag{9}$$

where k^l is the strength of community l, $k^l = \sum_{n \in N^l} k_n$ (initially all δ_l are set to 0.5). As the strategy adjusts values of δ_l with respect to each individual community, the algorithm more accurately reveals different composites of link-density and link-pattern communities (Sect. 3).

For networks with clear assortative or disassortative mixing, values δ_l can in fact be more accurately estimated on the level of the entire network (Sect. 3). Hence,

$$\delta_l = \sum_l \frac{|N^l|}{|N|}(1 - \Phi(l)), \tag{10}$$

while the resulting algorithm is denoted $\mathrm{GPA_N}$.

All proposed algorithms have complexity near $O(k|E|)$, where k is the average degree in the network.

3 Results and discussion

In the following sections we analyze the proposed algorithms on different synthetic and real-world networks (Sects. 3.1 and 3.2, respectively).

General propagation algorithms (i.e., GPA, $\mathrm{GPA_C}$ and $\mathrm{GPA_N}$) are compared against two other approaches. As a representative of classical community detection algorithms, we employ basic label propagation (i.e., LPA). Next, we also adopt the mixture model with expectation-maximization [51] proposed by Newman and Leicht [23] (denoted $\mathrm{MM_{EM}}$). Their algorithm can detect arbitrary network modules and is currently among state-of-the-art approaches for detection of link-pattern communities [23,25]. Still, it demands the number of communities to be known beforehand. Note that the exact number of communities (currently) cannot be adequately estimated in large real-world networks [52]. Due to simplicity, we limit the number of iterations to 100 for all the algorithms.

The results are assessed in terms of normalized mutual information (NMI) [9], which has become a de facto standard in community detection literature. Let $\mathcal{C}$ be a partition revealed by the algorithm and let $\mathcal{P}$ be the true partition of the network (corresponding random variables are C and P, respectively). NMI of $\mathcal{C}$ and $\mathcal{P}$ is then

$$\mathrm{NMI} = \frac{2I(C, P)}{H(C) + H(P)}, \tag{11}$$

where $I(C, P)$ is the mutual information of the partitions, i.e., $I(C, P) = H(C) - H(C|P)$, and $H(C)$, $H(P)$ and $H(C|P)$ are standard and conditional entropies. NMI of identical partitions equals 1, and is 0 for independent ones.

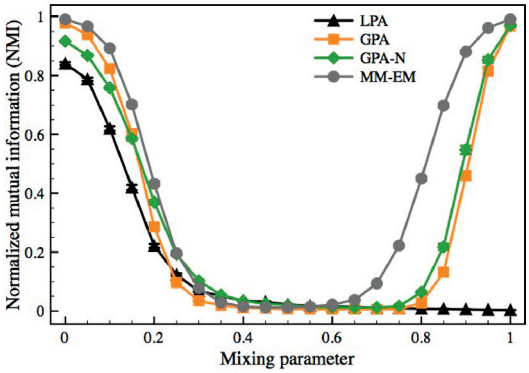


Fig. 2. (Color online) Mean NMI over 1000 realizations of synthetic networks with two communities. Error bars showing standard error of the mean are smaller than the symbol sizes.

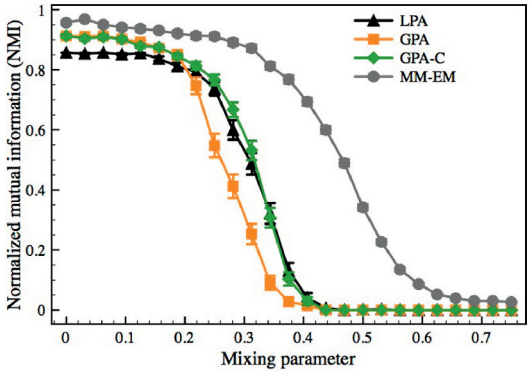


Fig. 3. (Color online) Mean NMI over 100 realizations of synthetic networks with four communities. Error bars show standard error of the mean.

3.1 Synthetic networks

The algorithms were first applied to synthetic benchmark networks with two communities of 32 nodes. Average degree is fixed to 6, while the community structure is controlled by a mixing parameter μ, $\mu \in [0, 1]$. When μ equals 0, all edges are (randomly) placed between the nodes of the same community, and when μ equals 1, all edges are (randomly) placed between the nodes of different communities. Thus, when μ varies from 0 to 1, community structure ranges between link-density and link-pattern regime (i.e., assortative and disassortative mixing). Note that network structure is completely random for $\mu = 0.5$.

The results appear in Figure 2. As anticipated, classical community detection algorithm LPA is unable to distinguish between a network with disassortative mixing and a completely random network (i.e., $\mu \approx 1$ and $\mu \approx 0.5$, respectively). Moreover, LPA also has the worst performance for all community regimes. On the other hand, mixture model $\mathrm{MM_{EM}}$ performs significantly better than other algorithms, especially in the case of link-pattern communities (i.e., $\mu > 0.5$). However, we argue that this is largely due to the fact that the algorithm is given the true number of communities in advance.

Observe that general propagation algorithms GPA and $\mathrm{GPA_N}$ can indeed detect both link-density and link-pattern communities. However, the algorithm with a network-wise re-estimation of δ_l performs slightly better, except when the structure results in clear link-density communities (i.e., $\mu \leq 0.1$). Still, the analysis on real-world networks in Section 3.2 confirms that $\mathrm{GPA_N}$ more accurately reveals different types of communities (including link-density).

We further apply the algorithms to a class of benchmark networks also adopted in [25]. The latter is in fact a generalization of the benchmark proposed by [1] Girvan and Newman for classical community detection. More precisely, networks comprise four communities of 32 nodes, thus, two communities correspond to classical link-density modules, while the other two form a bipartite structure of link-pattern communities. The networks are thus neither assortative nor disassortative (but locally assortative or disassortative). Average degree is fixed to 16, while the community structure is again controlled by a mixing parameter μ, $\mu \in [0, 1]$. Lower values correspond to clearer community structure – when $\mu = 0.5$, one half of the edges is set according to the designed structure, while the other are placed at random (on average).

The results in Figure 3 also report the performance of LPA, although a classical community detection algorithm is obviously not suited for these networks. However, one can thus observe that, when community structure is rather clearly defined (i.e., $\mu < 0.25$), only a small improvement can be achieved with a general community detection algorithm (on these networks). Therefore, to more accurately estimate the performance of GPA and $\mathrm{GPA_C}$, we increase the value of parameter β to 4 (Sect. 2.1). This further stabilizes the community structure identified by the algorithms, however, the computational time thus increases.

Mixture model $\mathrm{MM_{EM}}$ performs significantly better than other algorithms, which could be attributed to a known number of communities as above. Otherwise, general propagation algorithms GPA and $\mathrm{GPA_C}$ both detect link-density and link-pattern communities within these networks, however, only until communities are clearly depicted in the networks' topologies (i.e., $\mu < 0.25$). When μ further increases, the algorithm with a cluster-wise re-estimation of δ_l still manages to reveal (link-density) communities to some extent, whereas, GPA already fails.

Considering also the results reported in [25], image graph approach of Pinkert et al. [25] performs even slightly better than $\mathrm{MM_{EM}}$, while the model selection of Rosvall and Bergstrom [14] is a bit worse than $\mathrm{GPA_C}$. Thus, we conclude that general propagation algorithms can indeed reveal link-density and link-pattern communities under the same framework, still, the accuracy on these networks is worse with respect to some other state-of-the-art approaches. However, all these approaches demand the number of communities to be given apriori, thus, the algorithms are actually not fully comparable. Moreover, analysis on real-world networks in Section 3.2 reveals that, when the number of communities increases, the above advantage is in fact rendered useless.

Table 1. Mean NMI over 10 000 and 1000 runs for *karate*, *women* and *football*, *corporate* networks, respectively.

Network	Communities	LPA	GPA	GPA_N	GPA_C	MM_{EM}
Karate	2	*0.6501*	0.6992	0.7625	0.7547	**0.7806**
Football	12	*0.8908*	0.8464	**0.8570**	0.8493	0.8069
Women	4	–	0.7663	0.7680	0.7675	**0.8337**
Corporate	8 (9)	–	0.6680	**0.6735**	0.6651	0.5995

Note that the above benchmark networks represent a relatively poor description of real-world network structure (see Sect. 2). However, construction of networks with both assortative and disassortative mixing is not straightforward, as one inevitably has to define how link-pattern communities connect with the rest of the network. Still, generalization of hierarchical network model [6,53] appears as the most prominent formulation of different community regimes. Here, probabilities assigned to nodes of a predefined hierarchy of communities dictate the connections between the nodes in the network. High probabilities at the bottom level of the hierarchy yield classical cohesive modules, whereas link-pattern communities are characterized by higher probabilities at one level above.

To further validate the proposition, we have also applied the propagation algorithms to a random graph à la Erdös-Rényi [54] that (presumably) has no community structure. The number of nodes is fixed to 256, while we vary the average degree k between 2 and 64. When k exceeds a certain threshold, all algorithms reveal only trivial communities (i.e., connected components of the network). The transition occurs at $k \approx 8$, $k \approx 10$ and $k \approx 12$ for LPA, GPA and GPA_C, and GPA_N, respectively. Hence, community structures revealed by general propagation algorithms are beyond simple random configurations, while the algorithms are also not attributed to resolution limit [55] issues (i.e., existence of an intrinsic scale, below which communities are no longer recognized).

3.2 Real-world networks

The proposed algorithms were further applied to ten real-world networks with community structure (Tab. 2). All these networks are commonly analyzed in the community detection literature and include different social, technological, information and biological networks (detailed description is omitted). Due to simplicity, all networks are treated as unweighted and undirected. Furthermore, *corporate*, *jung* and *javax* networks are reduced to largest connected components and treated as simple graphs.

We first consider four well known social networks, namely, *karate*, *football*, *women* and *corporate* networks. The former two represent classical benchmarks for link-density community detection, as they reveal clear assortative mixing (Fig. 1a). On the other hand, the latter two are in fact bipartite networks, thus, the respective network communities can be considered of pure link-pattern type (Fig. 1b). However, the networks are not properly disassortative, due to different types of nodes.

Table 2. Real-world networks with community structure.

Network	Description	Nodes	Edges
Karate	Zachary's karate club [39]	34	78
Football	Amer. football league [1]	115	616
Women	Davis's south. women [40]	18, 14	89
Corporate	Scottish corporates [56]	131, 86	348
Jung	JUNG graph library [57]	305	710
Javax	Java library (`javax`) [57]	705	3313
Amazon	Amazon web graph[1]	2879	5037
Protein	S. cerevisiae proteins [3]	2445	6265
Gnutella	Gnutella peer-to-peer [58]	62 586	147 892
Condmatt	Cond. Matt. archive [59]	36 458	171 736

[1]Stanford WebBase project, `http://diglib.stanford.edu:8091/~testbed/doc2/WebBase/`.

All these networks have known sociological partitions into communities that result from earlier studies, while partition of *corporate* network is limited to only 86 corporate nodes. Comparison between community structures extracted by different algorithms and known network structures can be seen in Table 1. The number of communities in MM_{EM} algorithm is set to the true value for all networks except *corporate*, where we set it to nine (Tab. 1).

Although the mixture model MM_{EM} performs better than general propagation algorithms on synthetic benchmark networks (Sect. 3.1), the latter appears to be dependent on the number of communities. When the number of communities, and thus the size of the network, is relatively small (i.e., *karate* and *women* networks), the MM_{EM} most accurately reveals the true network structure. However, when the number of communities increases (i.e., *football* and *corporate* networks), all propagation algorithms significantly outperform MM_{EM}. The latter can be related to previously discussed weakness of MM_{EM}.

Note that somewhat lower performance of propagation algorithms on *karate* and *women* networks is actually due to the fact that the algorithms reveal three communities in these networks, which does not coincide with the sociological partitioning of the nodes. In particular, the algorithms extract a small module from the larger community in *karate* network (Fig. 1a), and merge the two communities representing events in *women* network (Fig. 1b). However, similarly as in the case of sociological communities, both these structures are well supported by the networks' topologies and thus commonly reported by community detection algorithms in the literature. Considering

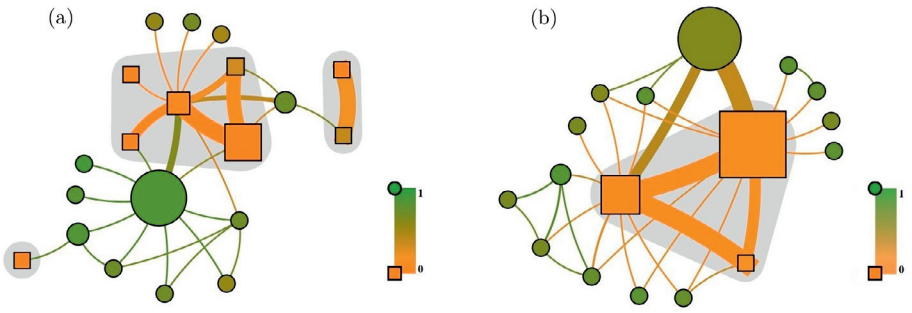


Fig. 4. (Color online) Community structures of (a) *jung* and (b) *javax* technological networks revealed with GPA_C. Node sizes are proportional to the community sizes, while the symbols (colors) correspond to the values of δ_l (Eq. (9)).

the partition of *women* network with three communities, GPA, GPA_N and GPA_C reveal structures with NMI equal to 0.8769, 0.8809 and 0.8799, respectively, while MM_{EM} obtains only 0.8027 (on average).

General propagation algorithms with re-estimation of δ_l, i.e., GPA_C and GPA_N, mostly outperform the basic GPA. As the algorithms adopt to either assortative or disassortative mixing regime in each network, they manage to extract the true communities more accurately. Observe also that network-wise re-estimation is somewhat more adequate for these networks than a cluster-wise version, due to a clear mixing regime. However, for networks with both types of mixing, GPA_C should obviously be employed.

We conclude that general propagation can reveal link-density and link-pattern communities in real-world networks. Thus, exactly the same algorithm is suitable for classical community detection in unipartite networks and link-pattern community detection in multi-partite networks. With respect to high values of NMI in Table 1 (except for *corporate* network), the proposed algorithms can also be considered as relatively accurate.

As the above social networks are particularly homogeneous, they reveal either assortative or disassortative mixing. However, social networks could indeed comprise both regimes, still, such networks would have to be heterogeneous by nature (i.e., convey different types of relations between individuals). In fact, heterogeneity seems to be a necessary condition for a network to reveal different composites of link-density and link-pattern communities. In the following we analyze four of the remaining networks in Table 2 that are all heterogeneous by nature.

Our main intention in the following is to reveal meaningful composites of not only link-density but also link-pattern communities, and thus imply that such structures could appear ubiquitous in various complex networks. Therefore, we apply GPA_C to each network 10 times, and report the structure with the highest fraction of nodes within link-pattern communities. It should be noted that community structures of these networks should not be considered identified, as networks possibly reveal a large number of different structures that are all significant and well supported by their topologies [53] (e.g., communities exist on different scales). Note that multiple structures could also imply that no clear one exists (e.g., overlapping communities [3]). However, general propagation algorithms find no communities in random networks (Sect. 3.1), thus, all revealed structures are at least beyond random.

First, we analyze two technological networks, namely, *jung* and *javax* networks (Tab. 2). These are class dependency networks, where nodes correspond to software classes and edges represent different types of dependencies among them (e.g., inheritance, parameters, variables etc.). The networks are thus obviously heterogeneous and should comprise different types of communities [57].

Revealed community structures are shown in Figure 4. Observe that networks convey both clear link-density and link-pattern communities, whereas, the latter are further combined in rather complex configurations (i.e., shaded regions in Fig. 4). In particular, besides simple bipartite structures and isolated link-pattern communities, networks also reveal connected clusters of multiple link-pattern communities. Note that, although link-pattern communities are mainly connected between themselves, they can also be strongly connected with else cohesive modules of nodes. Moreover, both link-density and link-patter communities can reside in either network interior or periphery.

We next analyze the main communities in greater detail (Tab. 3). The core, i.e., major link-density community, of *jung* network (Fig. 4a) consists of only visualization classes, while these are else almost inexistent in other communities. As one could anticipate, the community is highly cohesive and independent from the rest of the network. Two link-pattern communities on the right-hand side contain utility classes for GraphML format; while the upper community mainly contains different parsers, the lower mostly consists of meta-data classes, used by the former. Thus, the number of inter-community edges is obviously high. Central configuration of five link-pattern communities also contains well defined modules with particularly clear functional roles. More precisely, communities contain basic graph classes, interfaces for various algorithms, their implementations, different layout classes and filters, respectively. The strength of connections among the communities further supports this functional differentiation (e.g., implementations of different algorithms are strongly dependent on various interfaces and graph classes).

Similarly clear communities are also revealed in *javax* network (Fig. 4b). The core of the network consists of

Table 3. Analysis of community structures revealed in technological networks (Fig. 4). 'core' denotes the largest link-density community, while 'k-configuration'-s represent shaded regions in Figure 4 (k is the number of link-pattern communities).

Network	Community l	$\|N^l\|$	δ_l	Description
Jung	Core	65	0.86	[`jung.visualization.`] `*(Server|Viewer|Pane|Model|Context)` (9); `control.*` (4); `control.*Control` (5); `layout.*` (7); `picking.*State` (3); `picking.*Support` (6); `renderers.*Renderer` (13); `renderers.*Support` (3); etc.
	5-conf. (upper left)	3	0.00	[`jung.algorithms.filters.`] `*Filter` (3).
	5-conf. (upper right)	21	0.33	[`jung.graph.`] `*(Graph|Multigraph|Tree)` (18); etc.
	5-conf. (central)	28	0.07	[`jung.`] `algorithms.generators.*Generator` (2); `algorithms.importance.*` (4); `algorithms.layout.*Layout*` (3); `algorithms.scoring.*Scorer` (2); `algorithms.shortestpath.*` (2); `graph.*(Graph|Tree|Forest)` (4); etc. (interfaces)
	5-conf. (lower left)	13	0.00	[`jung.algorithms.`] `layout.*Layout*` (7); `layout3d.*Layout` (3); etc.
	5-conf. (lower right)	44	0.03	[`jung.`] `algorithms.cluster.*Clusterer*` (4); `algorithms.generators.random.*Generator` (5); `algorithms.importance.*Betweenness*` (3); `algorithms.metrics.*` (3); `algorithms.scoring.**` (5); `algorithms.shortestpath.*` (5); `graph.util.*` (7); etc. (implementations)
	2-conf. (upper)	13	0.03	[`jung.io.graphml.`] `parser.*Parser` (10); etc.
	2-conf. (lower)	13	0.38	[`jung.io.graphml.`] `*Metadata` (8); etc.
	1-conf. (central)	2	0.00	[`jung.visualization.control.`] `*Plugin` (2).
Javax	Core	179	0.64	[`javax.swing.`] `plaf.*UI` (24); `plaf.basic.Basic*UI` (42); `plaf.metal.Metal*UI` (22); `plaf.multi.Multi*UI` (30); `plaf.synth.Synth*UI` (40); etc.
	3-conf. (upper)	193	0.15	[`javax.`] `accessibility.Accessible*` (10); `swing.J*` (41); `swing.**(Border|Borders|Box|Button|Dialog|Divider|Editor|Factory|Filter|Icon|Kit|LookAndFeel|Listener|Model|Pane|Panel|Popup|Renderer|UIResource|View)` (92); etc.
	3-conf. (left)	113	0.11	[`javax.`] `accessibility.Accessible*` (6); `swing.*` (34); `swing.event.*Event` (8); `swing.event.*Listener` (13); `swing.plaf.*UI` (6); etc.
	3-conf. (lower)	44	0.19	[`javax.swing.`] `text.*View` (15); `text.html.*View` (16); etc.

look-and-feel classes for different GUI components. Note that the majority of classes differ only in a small part of their name, which indicates the respective GUI component and look-and-feel implementation. In contrast to before, the community is not highly cohesive, as these classes are extensively used by, e.g., various GUI components. The latter in fact appear within the largest link-pattern community, which is thus strongly dependent on the former. Note also that the latter link-pattern community consists of almost all GUI components of Java, although they reside in various packages and their names (i.e., functions) differ substantially. For more details on community structures of both technological networks see Table 3.

Despite mostly qualitative analysis, general propagation algorithms indeed reveal significant community structures within these technological networks, while the communities can also be related to particularly clear functional roles. Obviously, the latter could not be detected under the classical framework of merely cohesive modules. Note also that the proposed algorithms do not only partition the underlying software systems, as in the case of classical community detection, but also reveal important dependencies among different subsystems that would otherwise remain concealed. It ought to be mentioned that we have previously conjectured the existence of other modules besides classical communities in software networks [57].

Next, we analyze the community structure of *amazon* information network that represents a small sample of Amazon web graph (Tab. 2). The revealed network structure can be seen in Figure 5. Due to the size of the network and the nature of the domain, an exact analysis of extracted communities could not be conducted. Still, in the following, we discuss the main properties and highlight some interesting observations.

A large number of nodes is classified into dense core of the network (1381 nodes), however, the algorithm also reveals five well defined communities in the periphery (with 300 nodes on average). Thus, as one could anticipate, the extracted partition rather accurately coincides with the core-periphery structure [7] that is commonly found in information networks [7,60]. For reference, value of δ_l for the core equals 0.86, and is 0 for the only link-pattern community. Communities in periphery exhibit 0.86 on average.

We have analyzed the link-pattern community in greater detail and observed that the majority of its nodes

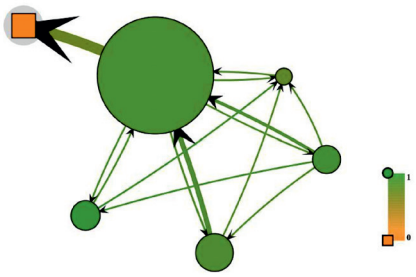


Fig. 5. (Color online) Community structure of *amazon* information network revealed with GPA_C. Edge directions were not considered by the algorithm.

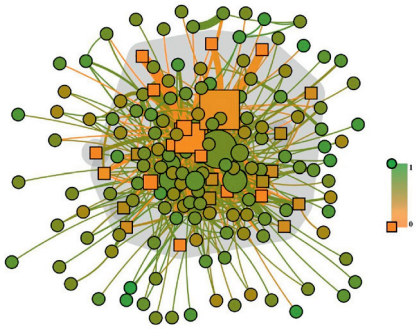


Fig. 6. (Color online) Community structure of *protein* biological network revealed with GPA_C.

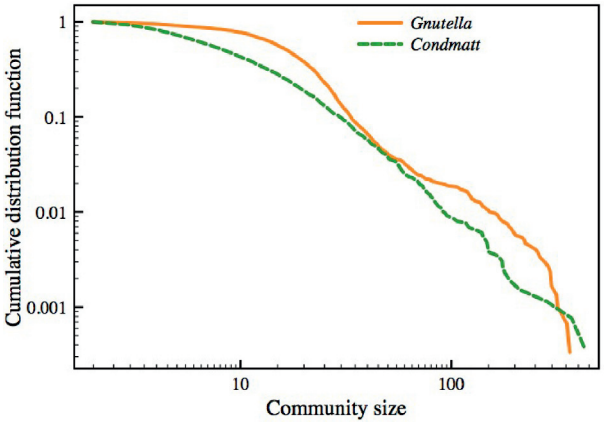


Fig. 7. (Color online) Cumulative size distributions of community structures revealed with GPA_C in *gnutella* information and *condmatt* social networks.

correspond to web pages on musical instruments[2] sold on Amazon. In particular, 231 of 288 nodes represent web pages on various instruments, while each page corresponds to a different brand (e.g., Yamaha). What makes the community particularly significant is the fact that only one of other 2591 nodes in the network also represents a web page on musical instruments (the latter is the node connected to all nodes in the respective community). Hence, the algorithm manages to extract a meaningful link-pattern community from the core of the network, while the community is not only exhaustive but also rather clear.

Observe that link-density communities generally more strongly connect towards the core of the network, whereas, in the case of link-pattern community, the connection is significantly stronger in the direction from the core. As the network was treated as undirected, the latter cannot be considered as an artifact of the algorithm. The revealed pattern could in this context imply that nodes in link-pattern communities provide important content (i.e., authority nodes [61]), while hub nodes [61] reside mainly in link-destiny communities. Again, the occurrence of different types of communities can be related to a form of network heterogeneity (i.e., edge directions).

For a complete analysis, we also apply the algorithm to an example of a biological network (that is also heterogeneous by definition). In particular, we analyze *protein* network that represents protein-protein interactions of yeast Saccharomyces cerevisiae (Tab. 2). The revealed community structure appears in Figure 6, while detailed description of communities is omitted. Observe that the algorithm reveals a large number of clear link-density and link-pattern communities of various sizes (171 communities of 2 to 127 nodes), while both exist in the interior and the periphery of the network. Different types of communities are combined in complex configurations (shaded regions in Fig. 6), which, as in the examples above, suggests that link-pattern communities, similarly as link-density counterparts, are ubiquitous in real-world networks.

Last, we also analyze community structures revealed with GPA_C in two remaining networks in Table 2. More precisely, we consider *gnutella* information network of peer-to-peer communications within Gnutella file sharing, and *condmatt* social network representing scientific author collaborations extracted from Condensed Matter archive. While the former can be characterized by a unique dissassortative behavior, the latter is in fact a prominent example of assortative mixing, and thus classical community structure. Indeed, more than 92 percent of the nodes in *gnutella* network are classified into 2670 link-pattern communities, while, on the contrary, almost 85 percent of the nodes in *condmatt* network reside in 2100 classical link-density modules (δ_l equals 0.30 and 0.64 on average, respectively). Figure 7 shows also cumulative community size distributions for both networks. Although distribution for *condmatt* network appears to be power-law for the most part, as commonly observed in classical community detection [3,22], the latter does not hold for *gnutella* network. In particular, communities most distinctively exists on two scales with tens and hundreds of nodes, which provides some evidence that link-pattern communities might reflect in disassortative mixing by degree (Sect. 2).

4 Conclusions

The paper proposes a balanced propagation based algorithm for detection of arbitrary network modules, ranging from classical cohesive (link-density) communities to

[2] This can be determined by the occurrence of '11091801' within the URL of the respective web page.

more general link-pattern communities. The proposed algorithm was first validated on synthetic benchmark networks with community structure, and also on random networks. It was then further applied to different social, technological, information and biological networks, where it indeed reveals significant (composites of) link-density and link-pattern communities. In the case of larger real-world networks, the proposed algorithm more accurately detects the true communities than a state-of-the-art algorithm, while, in contrast to other approaches proposed in the literature, it does not require some prior knowledge of the true network structure. The latter is in fact crucial for the analysis of large real-world networks [52].

Heterogeneity appears to be a necessary condition for the network to reveal both link-density and link-pattern communities. However, although often not apparent at first sight, most real-world networks are in fact heterogeneous by nature. Qualitative results on real-world networks further imply that link-pattern communities, similarly as link-density counterparts, appear ubiquitous in nature and technology. Moreover, link-pattern communities are also commonly combined with classical modules into complex configurations, thus, different types of communities should not be analyzed independently. A generative model or measure for a general community structure of real-world networks would be of great benefit. It ought to be mentioned that the existence of link-pattern communities in real-world networks has implications in numerous other fields of network science (e.g., dynamic processes).

The analysis in the paper does not directly imply which common properties of real-world networks one can expect under link-density or link-pattern regime. However, further work demonstrates that most significant link-pattern communities are revealed in regions with low values of clustering coefficients [62,63], while just the opposite holds for classical modules. Furthermore, link-pattern communities may be the origin of degree disassortativity observed in various real-world networks [42,43], while they also commonly contradict the small-world phenomena [62]. Hence, different network properties seem to be governed by the same underlying principle [8], which represents a prominent direction for future research.

This work has been supported by the Slovene Research Agency ARRS within Research Program No. P2-0359.

References

1. M. Girvan, M.E.J. Newman, Proc. Natl. Acad. Sci. USA **99**, 7821 (2002)
2. P. Gleiser, L. Danon, Adv. Complex Syst. **6**, 565 (2003)
3. G. Palla, I. Derényi, I. Farkas, T. Vicsek, Nature **435**, 814 (2005)
4. A. Arenas, A. Díaz-Guilera, C.J. Pérez-Vicente, Phys. Rev. Lett. **96**, 114102 (2006)
5. V. Zlatic, A. Gabrielli, G. Caldarelli, Phys. Rev. E **82**, 066109 (2010)
6. E. Ravasz, A. Barabási, Phys. Rev. E **67**, 026112 (2003)
7. J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Internet Mathematics **6**, 29 (2009)
8. D.V. Foster, J.G. Foster, P. Grassberger, M. Paczuski, Phys. Rev. E **84**, 066117 (2011)
9. L. Danon, A. Díaz-Guilera, J. Duch, A. Arenas, J. Stat. Mech.: Theory E., P09008 (2005)
10. S. Fortunato, Phys. Rep. **486**, 75 (2010)
11. M.A. Porter, J. Onnela, P.J. Mucha, Notices of the American Mathematical Society **56**, 1082 (2009)
12. F. Wu, B.A. Huberman, Eur. Phys. J. B **38**, 331 (2004)
13. R. Guimerà, L.A.N. Amaral, Nature **433**, 895 (2005)
14. M. Rosvall, C.T. Bergstrom, Proc. Natl. Acad. Sci. USA **104**, 7327 (2007)
15. S. Son, H. Jeong, J.D. Noh, Eur. Phys. J. B **50**, 431 (2006)
16. U.N. Raghavan, R. Albert, S. Kumara, Phys. Rev. E **76**, 036106 (2007)
17. G. Agarwal, D. Kempe, Eur. Phys. J. B **66**, 409 (2008)
18. T.S. Evans, R. Lambiotte, Eur. Phys. J. B **77**, 265 (2010)
19. J. Liu, Eur. Phys. J. B **77**, 547 (2010)
20. L. Šubelj, M. Bajec, Phys. Rev. E **83**, 036103 (2011)
21. M.E.J. Newman, Eur. Phys. J. B **38**, 321 (2004)
22. F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, D. Parisi, Proc. Natl. Acad. Sci. USA **101**, 2658 (2004)
23. M.E.J. Newman, E.A. Leicht, Proc. Natl. Acad. Sci. USA **104**, 9564 (2007)
24. B. Long, X. Xu, Z. Zhang, P. Yu, Community learning by graph approximation, in *Proceedings of the IEEE International Conference on Data Mining* (2007), pp. 232−241
25. S. Pinkert, J. Schultz, J. Reichardt, PLoS Comput. Biol. **6**, e1000659 (2010)
26. Y. Park, C. Moore, J.S. Bader, PLoS One **5**, e8118 (2010)
27. H.C. White, S.A. Boorman, R.L. Breiger, Am. J. Sociol. **81**, 730 (1976)
28. Y. Zhao, E. Levina, J. Zhu, Proc. Natl. Acad. Sci. USA **108**, 7321 (2011)
29. P. Latouche, E. Birmelé, C. Ambroise, Ann. Appl. Stat. **5**, 309 (2011)
30. A. Decelle, F. Krzakala, C. Moore, L. Zdeborová, Phys. Rev. Lett. **107**, 065701 (2011)
31. E.M. Airoldi, D.M. Blei, S.E. Fienberg, E.P. Xing, J. Mach. Learn. Res. **9**, 1981 (2008)
32. N. Barbieri, M. Guarascio, G. Manco, A block mixture model for pattern discovery in preference data, in *Proceedings of the IEEE International Conference on Data Mining Workshops* (2010), pp. 1100–1107
33. C. Lin, J. Koh, A.L.P. Chen, A better strategy of discovering link-pattern based communities by classical clustering methods, in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining* (2010), pp. 56–67
34. M. Shafiei, H. Chipman, Mixed-membership stochastic block-models for transactional networks, in *Proceedings of the IEEE International Conference on Data Mining* (2010), pp. 1019–1024
35. K. Henderson, T. Eliassi-Rad, S. Papadimitriou, C. Faloutsos, HCDF: A hybrid community discovery framework, in *Proceedings of the SIAM International Conference on Data Mining* (2010), pp. 754–765
36. K. Cooper, M. Barahona, e-print `arXiv:10122726v1` [physics.soc-ph] (2010)
37. A. Decelle, F. Krzakala, C. Moore, L. Zdeborová, Phys. Rev. E **84**, 066106 (2011)
38. L. Šubelj, M. Bajec, Eur. Phys. J. B **81**, 353 (2011)

39. W.W. Zachary, J. Anthropol. Res. **33**, 452 (1977)
40. A. Davis, B.B. Gardner, M.R. Gardner, *Deep South* (Chicago University Press, 1941)
41. M.E.J. Newman, M. Girvan, Phys. Rev. E **67**, 026126 (2003)
42. M.E.J. Newman, Phys. Rev. Lett. **89**, 208701 (2002)
43. M.E.J. Newman, J. Park, Phys. Rev. E **68**, 036122 (2003)
44. L. Šubelj, M. Bajec, Unfolding network communities by combining defensive and offensive label propagation, in *Proceedings of the ECML PKDD Workshop on the Analysis of Complex Networks* (2010), pp. 87–104
45. M.J. Barber, J.W. Clark, Phys. Rev. E **80**, 026129 (2009)
46. I.X.Y. Leung, P. Hui, P. Liò, J. Crowcroft, Phys. Rev. E **79**, 066107 (2009)
47. S. Gregory, New J. Phys. **12**, 103018 (2010)
48. P. Boldi, M. Rosa, M. Santini, S. Vigna, Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks, in *Proceedings of the International World Wide Web Conference* (2011), pp. 587–596
49. G. Tibély, J. Kertész, Physica A **387**, 4982 (2008)
50. B. Bollobás, *Modern graph theory* (Springer, 1998)
51. A.P. Dempster, N.M. Laird, D.B. Rubin, J. R. Stat. Soc. **39**, 1 (1977)
52. B. Karrer, M.E.J. Newman, Phys. Rev. E **83**, 016107 (2011)
53. A. Clauset, C. Moore, M.E.J. Newman, Nature **453**, 98 (2008)
54. P. Erdős, A. Rényi, Publ. Math. Debrecen **6**, 290 (1959)
55. S. Fortunato, M. Barthelemy, Proc. Natl. Acad. Sci. USA **104**, 36 (2007)
56. J. Scott, M. Hughes, *The anatomy of Scottish capital: Scottish companies and Scottish capital, 1900-1979* (Croom Helm, 1980)
57. L. Šubelj, M. Bajec, Physica A **390**, 2968 (2011)
58. J. Leskovec, J. Kleinberg, C. Faloutsos, ACM Trans. Knowl. Discov. Data **1**, (2007)
59. M.E.J. Newman, Proc. Natl. Acad. Sci. USA **98**, 404 (2001)
60. J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Statistical properties of community structure in large social and information networks, in *Proceedings of the International World Wide Web Conference* (2008), pp. 695–704
61. J.M. Kleinberg, J. ACM **46**, 604 (1999)
62. D.J. Watts, S.H. Strogatz, Nature **393**, 440 (1998)
63. S.N. Soffer, A. Vázquez, Phys. Rev. E **71**, 057101 (2005)

6

Analiza in uporaba skupin v programskih omrežjih

V pričujočem poglavju je vloženo delo z naslovom "Analiza in uporaba skupin v programskih omrežjih" [117] (angl. "Community structure of complex software systems: Analysis and applications"). Delo predstavi rezultate analize karakterističnih skupin vozlišč v omrežjih odvisnosti med razredi objektno-usmerjenega programja. Rezultati kažejo, da programska omrežja vsebujejo izrazite skupine vozlišč, dočim le-te (delno) sovpadajo s paketi pripadajočih programskih knjižnic. Na podlagi dognanj analize so predlagani tudi različni primeri uporabe v programskem inženirstvu.

Predstavljeno delo je v razširjeni obliki objavljeno v [72, 73, 117, 120, 139].[5]

[5]Privzeta programska omrežja so dostopna na `http://lovro.lpt.fri.uni-lj.si/`.

Physica A 390 (2011) 2968–2975

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Community structure of complex software systems: Analysis and applications

Lovro Šubelj *, Marko Bajec

Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 10 August 2010
Received in revised form 1 November 2010
Available online 12 April 2011

Keywords:
Community structure
Complex networks
Software systems

ABSTRACT

Due to notable discoveries in the fast evolving field of complex networks, recent research in software engineering has also focused on representing software systems with networks. Previous work has observed that these networks follow scale-free degree distributions and reveal small-world phenomena, while we here explore another property commonly found in different complex networks, i.e. community structure. We adopt class dependency networks, where nodes represent software classes and edges represent dependencies among them, and show that these networks reveal a significant community structure, characterized by similar properties as observed in other complex networks. However, although intuitive and anticipated by different phenomena, identified communities do not exactly correspond to software packages. We empirically confirm our observations on several networks constructed from *Java* and various third party libraries, and propose different applications of community detection to software engineering.

1. Introduction

Analysis of complex real-world networks has led to some significant discoveries in recent years. The research community has revealed several common properties of different real-world networks [1–3], including various social, biological, Internet, software and other networks. These properties provide an important insight into the function and structure of general complex networks [4,5]. Moreover, they allow for better comprehension of the underlying real-world systems and thus give prominent grounds for future research in a wide variety of different fields.

In the field of software engineering, network analysis has just recently been adopted to acquire better comprehension of the complex software systems [6–10]. Nowadays, software represents one of the most diverse and sophisticated human made systems; however, only little is known about the actual structure and quantitative properties of (large) software systems. Cai and Yin [9] have denoted this dilemma as the *software law problem*, which represents an effort towards identifying and formulating physics-like laws, obeyed by (most) software systems, that could later be applied in practice. However, the main objective of the software law problem is to investigate what software looks like.

In the context of employing complex networks analysis, the research community has already made several discoveries over the past years. In particular, different authors have observed that networks, constructed from various software systems, follow *scale-free* [2] (i.e. power-law) degree distributions and reveal *small-world* [1] phenomena [11,6]. We proceed this work by exploring another property commonly found in other real-world networks, i.e. *community structure* [3]. The term denotes the occurrence of local structural modules (*communities*) that are groups of nodes densely connected within and only loosely connected with the rest of the network. Communities play crucial roles in many real-world systems [12,5], however, the

* Corresponding author.
E-mail address: lovro.subelj@fri.uni-lj.si (L. Šubelj).

0378-4371/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.physa.2011.03.036

community structure of complex software system networks has not yet been thoroughly investigated. For a comprehensive survey on network community structure see Refs. [13,14].

The main contributions of our work are as follows. We adopt *class dependency networks*, where nodes represent software classes and edges represent dependencies among them, and show that these networks reveal a significant community structure, with similar properties as observed for other complex networks. We also note that a network, representing a core software library, exhibits less significant community structure. Furthermore, we prove that, although intuitive and anticipated by different phenomena, revealed communities do not (completely) correspond to software packages. Thus, we demonstrate how community detection can be employed to obtain highly modular software packages that still relate to the original packages.

The rest of the article is structured as follows. First, in Section 2, we briefly present relevant related work and emphasize the novelty of our research. Next, Section 3 introduces employed class dependency networks. In Section 4 we present an empirical evaluation of the community structure of class dependency networks, and propose possible applications to software engineering. Finally, in Section 5, we give final conclusions and identify areas of further research.

2. Related work

Although software systems have already been investigated over the last 30 years [15], the research community has only recently begun to employ network analysis to gain better comprehension of complex software [6–8,16,9,10,17]. As mentioned above, different authors have observed that networks, constructed from software systems, follow scale-free degree distributions [11,6,18,19] and exhibit small-world properties [6,20,21]. Software networks thus reveal common behavior, similar to that observed in other complex networks [4,3]. Furthermore, authors have also identified several different phenomena (e.g. software optimization and tinkering) that might govern such complex behavior [11,22,23,21,24], when the analysis of *clustering* [1] has also revealed a hierarchical structure in software networks [6].

On the other hand, the community structure of software networks has not yet been thoroughly investigated. Several authors have already discussed the notion of communities in the context of software systems [6,22,25,21,26,16], however, no general empirical analysis and formal discussion was ever conducted (to the best of our knowledge). Still, authors have observed different phenomena that could promote the emergence of community structure in software networks [25,16], and have discussed possible applications within software engineering and other sciences [6,16]. Otherwise, several models (processes) of network evolution have already been proposed to explain the emergence of local structural modules in different networks. These include increase of stability [27], network fluctuations [28], goal variation [29], opinion formation [30], constraint optimization [31] and others [32]. However, the origin of community structure in software networks remains unclear.

In a wider context of software network analysis, authors have discovered reoccurring *motifs* (i.e. small subgraphs) in software networks [8], similar to those found in genetic and neural networks. In Ref. [7] a network model is presented, which explains observed asymmetries in degree distributions of software networks (see, e.g., Ref. [6]). Furthermore, different random-walk based measures have been proposed to measure key (i.e. most influential) software classes and packages [33,10]. The researchers have also investigated connectedness, robustness and patterns within software networks [6,26]. Just recently software systems were also treated as evolving complex networks [9]. For a more general discussion on software networks see Ref. [34].

3. Class dependency networks

Previous research on the analysis of software systems has employed a variety of different types of software networks (i.e. graphs). In particular, *package*, *class* and *method collaboration graphs* [6,18], *subrutine call graphs* [6], *software architecture* [26] and *software mirror graphs* [9], *software architecture maps* [11], *inter-package dependency networks* [20] and many others [6,21,36]. The networks primarily divide whether they are constructed from source code, byte code or software execution traces, and due to the level of software architecture they represent. However, as discussed in Section 2, most of these networks share some common characteristics.

For the purpose of this research we introduce *class dependency networks* (Fig. 1). Here an object-oriented software is represented by an undirected multi-graph $G(N, E)$, where N is the set of nodes and E is the set of edges. Graph G is constructed from software source code in the following manner. Each software class c is represented by a node $n_c \in N$, when edge $\{n_{c_1}, n_{c_2}\} \in E$ represents a *dependency* between classes c_1 and c_2. Dependencies are of four types, namely, *inheritance* (class c_2 inherits or implements class c_1), *field* (c_2 contains a field of type c_1), *parameter* (c_2 contains a method that takes type c_1 as a parameter) and *return* (c_1 contains a method that returns type c_2).

Note that class dependency networks are constructed merely from the header information of the classes, and their methods and fields. As this information is commonly determined by a group of developers, prior to the actual software development, it is less influenced by the subjective nature of each particular developer. Hence, the networks more adequately represent the (intended) structure of some particular software, still, some relevant information might thus be discarded.

An example of class dependency network is shown in Fig. 1. The network reveals rather strong community structure, furthermore, the communities also roughly coincide with the actual software packages. However, as will be shown in

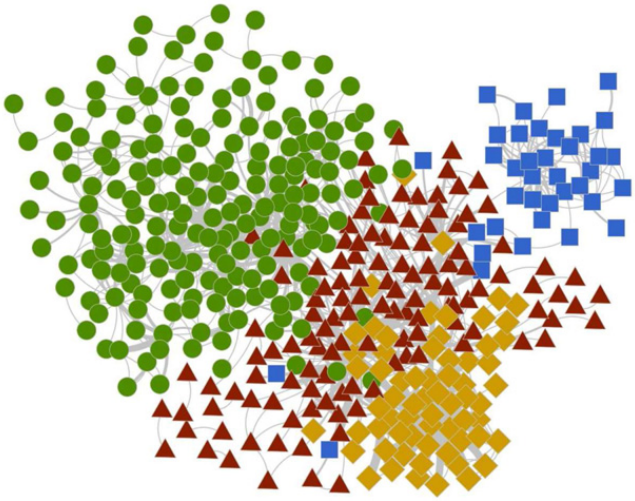


Fig. 1. Class dependency network for *JUNG* graph and network framework [35]. Node symbols (colors) indicate four high-level packages of the framework — `visualization` (green circles), `algorithms` (red triangles), `graph` (orange diamonds) and `io` (blue squares). The network reveals a rather clear community structure that roughly coincides with the software packages.

Table 1
Class dependency networks for different software systems ($|P|$ is the number of packages).

Network	Description	$\|N\|$	$\|E\|$	$\|P\|$
junit	*JUnit* 4.8.1 (testing framework). [37]	128	470	22
jmail	*JavaMail* 1.4.3 (mail and messaging framework). [38]	220	893	14
flamingo	*Flamingo* 4.1 (GUI component suite). [39]	251	846	16
jung	*JUNG* 2.0.1 (graph and network framework). [35]	422	1730	39
colt	*Colt* 1.2.0 (scientific computing library). [40]	520	3691	19
org	*Java* 1.6.0 (*org* namespace). [41]	716	7895	47
javax	*Java* 1.6.0 (*javax* namespace). [41]	2581	22370	110
java	*Java* 1.6.0 (*java* namespace). [41]	2378	34858	54

Section 4, modularity of the natural communities, depicted in the network's topology, is much larger than that of the packages, determined by the developers.

4. Empirical analysis and applications

In the proceeding sections we present and discuss results of the empirical evaluation of community structure of class dependency networks (Section 4.1), address the relation between communities and software packages (Section 4.2) and propose possible applications of community detection to software engineering (Section 4.3).

The empirical evaluation is done using 8 class dependency networks constructed[1] from *Java* and several third party libraries (Table 1). The networks range from those with hundreds of nodes to those with several tens of thousands of edges (all isolated nodes have been discarded). Due to generality, networks were selected thus they represent a relatively diverse set of software systems.

To reveal the community structure of each network we employ three community detection algorithms. In particular, a divisive algorithm based on *edge betweenness* [3], a greedy agglomerative optimization of *modularity* (see below) [42,43] and a fast partitional algorithm based on *label propagation* [44]. The algorithms are denoted *EB*, *MO* and *LP* respectively, whereas, the detailed description is omitted. It should be noted that our objective is not to compare the algorithms, but rather to compare the revealed communities, and thus address their stability.

The community structure, identified by the algorithms, is assessed using *modularity* Q [45] that measures the significance of communities due to a selected *null model*. Let l_i be the community (label) of node $n_i \in N$ and let A_{ij} denote the number of edges incident to nodes $n_i, n_j \in N$. Furthermore, let P_{ij} be the expected number of incident edges for n_i, n_j in the null model. The modularity then reads

$$Q = \frac{1}{2m} \sum_{n_i, n_j \in N} (A_{ij} - P_{ij})\, \delta(l_i, l_j), \tag{1}$$

[1] Networks were constructed by parsing *JAR* archives provided by the developers. However, due to various issues, some of the software classes could have been discarded.

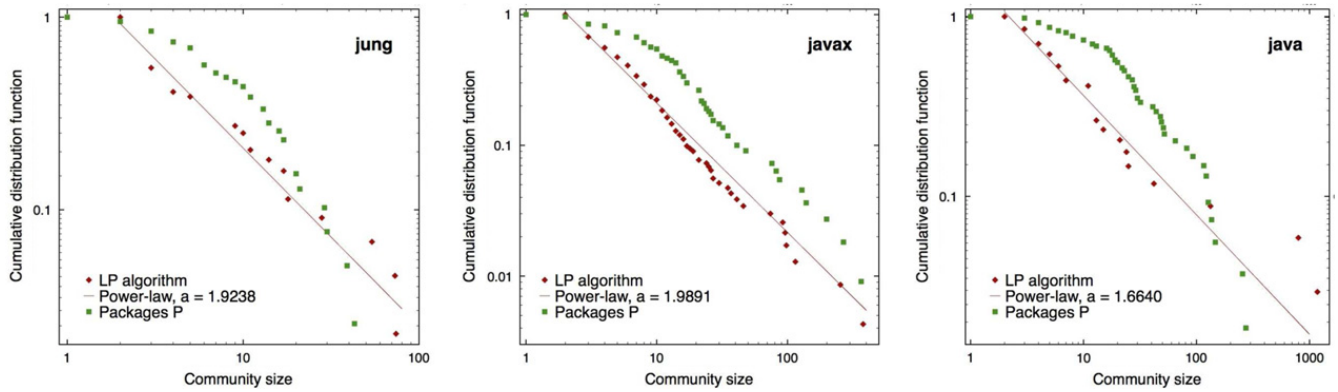


Fig. 2. Cumulative distribution functions of community sizes for *jung*, *javax* and *java* networks. The distributions revealed by *LP* algorithm (roughly) follow power-laws with the exponents α shown (i.e. $P(s) \sim s^{-\alpha}$, where s is the community size); however, the distributions of package sizes are not characterized by power-laws (e.g. log-normal distributions).

where m is the number of edges, $m = |E|$, and δ is the Kronecker delta. The modularity thus measures the fraction of the difference between the number of intra-community edges and the expected number of edges in the null model ($Q \in [-1, 1]$). Higher values represent stronger community structure. Commonly a random graph with the same degree distribution as the original is selected for the null model. Hence, $P_{ij} = \frac{k_i k_j}{2m}$, where k_i is the degree of node $n_i \in N$.

Despite being widely adopted for the analysis of network community structure in the past, the measure of modularity has a number of known weaknesses. Fortunato and Barthélemy [46] have shown that modularity is myopic to clearly defined communities (e.g. cliques) that are smaller than a certain threshold (i.e. *resolution limit* [46]). The analysis of Leskovec et al. [47] also confirms that modularity favors larger communities. Furthermore, Good et al. [48] have analyzed community structures with high values of modularity. Their results show that there is a great degeneracy among these structures, without a clear peak around an optimal solution (i.e. *degeneracy problem* [48]). Moreover, there is commonly an exponential number of community structures with high modularity, whereas, they can exhibit great structural differences.

Considering the above discussion, it is not surprising that modularity maximization algorithms perform poorly compared to other algorithms [49]. However, modularity is still commonly used for the evaluation of network community structure, mainly due to the lack of other measures.

The identified community structure is also compared to the actual software packages. Let $\mathcal{L}$ be the partition (i.e. communities) revealed by some algorithm and let $\mathcal{P}$ be the partition that represents software packages (corresponding random variables are L and P respectively). We compare the partitions by computing their *normalized mutual information NMI* [50] ($NMI \in [0, 1]$). Hence,

$$NMI = \frac{2I(L, P)}{H(L) + H(P)}, \tag{2}$$

where $I(L, P)$ is the *mutual information* of the partitions, $I(L, P) = H(L) - H(L|P)$, and $H(L)$, $H(P)$ and $H(L|P)$ are standard and conditional entropies. *NMI* of identical partitions equals 1, and is 0 for independent partitions.

4.1. Community structure of class dependency networks

Mean modularities obtained with three community detection algorithms for the selected set of class dependency networks (Section 4) can be seen in Table 2. For all networks except *java*, the algorithms have managed to reveal community structures with particularly high values of modularity, i.e. between 0.55 and 0.75 on average, where values above 0.30 are commonly regarded as an indication of (significant) community structure [44,51–53]. The networks thus reveal a much stronger community structure than expected in a random network with the same degree distribution. Note also that all of the algorithms obtain high modularities for all of the networks considered. This indicates rather stable communities, strongly depicted in the networks' topologies.

In the case of *java* network, observe that the average degree is considerably larger than for other networks (Table 1). Hence, the network is extremely dense and the communities are thus only loosely defined. Consequently, the algorithms fail to attain any significant community structure; however, as the network represents the core of *Java* programming language, it is expected to convey a less modular structure.

In Fig. 2 we show the (cumulative) distributions of community sizes obtained with the *LP* algorithm for *jung*, *javax* and *java* networks. Interestingly, the distributions (roughly) follow power-laws with the exponents α around 2 (i.e. $P(s) \sim s^{-\alpha}$, where s is the community size). Scale-free distribution of community sizes is a common property, observed in various other complex networks [42,5]. Furthermore, the values of α also coincide with values found for other networks, where authors commonly report α between 1 and 3 [43,42,54,5].

Table 2
Mean modularities Q obtained for class dependency networks of different software systems. Values were computed from 100 iterations (10 iterations for *EB* algorithm), where missing values could not be recovered due to limited time resources. Modularities of the natural community structures, depicted in the networks' topologies (i.e. extracted by the algorithms), are much larger than those of the actual software packages.

Network	*EB*	*MO*	*LP*	P^+	P
junit	0.5587	0.5759	0.5542	0.1140	0.0893
jmail	0.5607	0.5972	0.5401	0.2350	0.2086
flamingo	0.6466	0.6823	0.6485	0.2870	0.2511
jung	0.7210	0.7324	0.6874	0.3279	0.3212
colt	–	0.6025	0.5599	−0.0158	−0.0332
org	–	0.5599	0.5254	0.1847	0.1830
javax	–	0.7667	0.7422	0.3119	0.2907
java	–	0.4664	0.4132	0.2269	0.2206

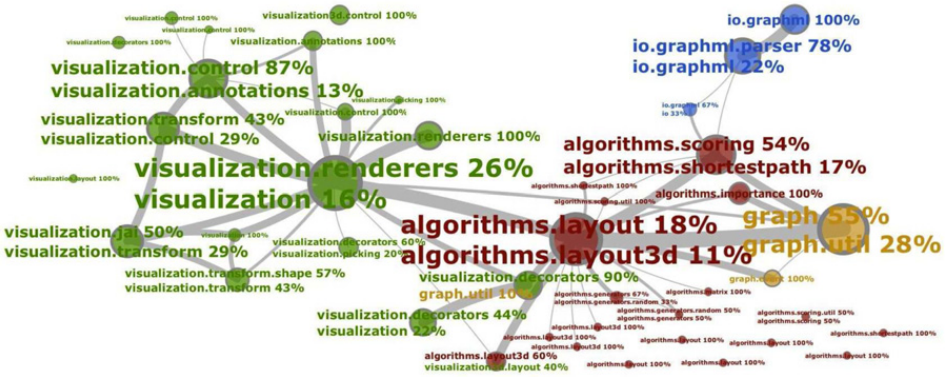


Fig. 3. Community network for *jung* class dependency network (Fig. 1) revealed by the *LP* algorithm (modularity equals $Q = 0.7062$). For each community we show the distribution of classes over software packages (weakly represented packages are not shown), where colors indicate four high-level packages of the framework (see Fig. 1). Communities clearly distinguish between high-level packages, but they do not completely coincide with the actual (bottom-most) packages.

We conclude that class dependency networks contain significant community structure that also reveals similar properties as observed in other complex networks. Thus, besides scale-free degree distributions and the small-world effect, software networks reflect another common network property, i.e. community structure.

To further address the validity of our results, we briefly discuss different phenomena that could promote the emergence of community structure in software networks. Li et al. [16] and Jenkins and Kirk [26] have discussed the influence of internal *cohesion*, i.e. functional strength of the components, and external *coupling*, i.e. inter-dependencies among components, on the structure of software systems (and networks). Highly modular software should clearly demonstrate the *minimum coupling-maximum cohesion* principle [55], which would naturally promote the emergence of strong structural modules within software networks. The modularity of software networks thus reflects the modularity of underlying software systems.

Furthermore, Baxter et al. [25] have emphasized that object-oriented software is commonly developed according to *Lego hypothesis* [56]. The hypothesis states that software is constructed out of a larger number of smaller components that are relatively independent of each other. This phenomena should clearly reflect in software networks, where components should emerge as network communities.

In summary, software networks enclose a strong natural tendency to form community structure. In the case of class dependency networks, communities should, due to the above discussion and by intuition, correspond to software packages (Fig. 3). This aspect is thoroughly explored and discussed in the proceeding section.

4.2. Relation of network communities to software packages

The analysis of the relation between network communities and software packages reveals that packages are considerably different to communities. We first note that packages do not feature *connectedness* in class dependency networks (exact results are omitted). The latter is regarded as a basic property of communities and states that communities should correspond to connected sets of nodes. As a consequence, software packages can comprise of disconnected sets of nodes, which is an indicator of relatively poor modular structure.

Let P represent the actual (bottom-most) software packages and let P^+ represent packages that feature connectedness (i.e. disconnected packages are treated as several different packages). Table 2 shows modularities of software packages for the analyzed class dependency networks. The values are considerably lower than the modularities of the natural community structures, revealed in the networks' topologies (i.e. extracted by the algorithms). Thus, they cannot be regarded as an indication of significant modular structure. Moreover, in Fig. 2 we show the distributions of package sizes for *jung*, *javax*

Table 3
Peak (maximum) *NMI* between network communities, extracted by the algorithms, and software packages *P* for different class dependency networks. Values were computed from 100 iterations (10 iterations for the *EB* algorithm). The results indicate relatively poor correspondence between natural network communities and software packages.

Network	*EB*	*MO*	*LP*	P^+
junit	0.6605	0.5823	0.6285	0.8412
jmail	0.5300	0.5248	0.5553	0.8379
flamingo	0.5686	0.5408	0.5590	0.7882
jung	0.6011	0.6094	0.6887	0.9187
colt	–	0.4784	0.5277	0.6507
org	–	0.5301	0.5385	0.9123
javax	–	0.6365	0.6826	0.8096
java	–	0.3453	0.3063	0.8386

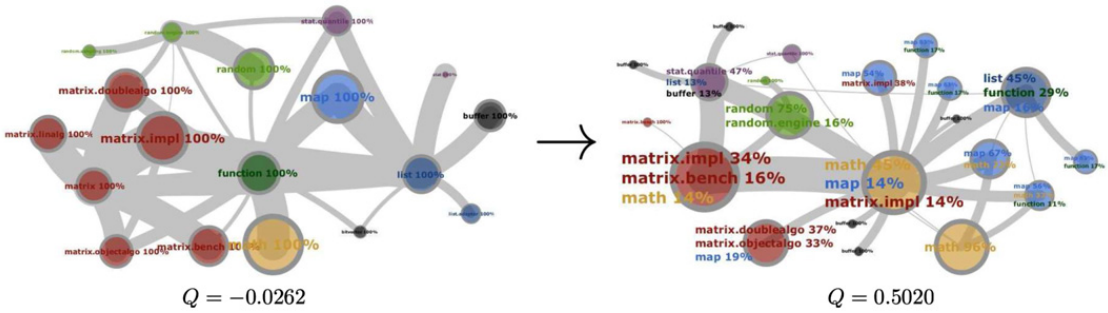


Fig. 4. Community networks for class dependency network that represents classes within `cern.colt` and `cern.jet` packages of *colt* library (reduced to the largest connected component). Networks correspond to the original software packages *P* (left) and communities, revealed with the *LP* algorithm by refining software packages *P* (right). For each community we show the distribution of classes over software packages, where colors indicate high-level packages of the library. Refined communities (i.e. packages) obtain significantly higher modularity and can still be related to the original packages.

and *java* networks. The distributions are obviously not characterized by power-laws, as observed in the case of communities (distributions are, e.g., log-normal or stretched exponential, which coincides with the observations in Ref. [25]). Last, we also (directly) compare the packages with network communities by computing *NMI* of the corresponding partitions (Table 3). The results further confirm the above observations – software packages only weakly relate to network communities and are not characterized by the same laws or properties.

We stress that the origin of the disparity between network communities and software packages is not entirely evident. The lack of connectedness of software packages, and low values of modularity, suggest that class dependency networks give a poor representation of software systems, or disregard some relevant relations among classes (from the perspective of software packages). However, different distributions of sizes clearly show that there is some additional departure between the communities and software packages, which is independent of the actual network representation (i.e. class dependencies).

Finally, we discuss a particularly low value of modularity for *colt* library packages (Table 2). As the library represents a core framework for scientific computing, where the performance is often of greater importance than extensibility, maintenance and modular structure, it is expected for the system to exhibit only poor modular structure. The modularity of software packages thus reflects the modularity of the underlying software system, which in fact motivates the applications, presented in the proceeding section.

4.3. Applications of community detection to software engineering

Due to the weak modular structure of software packages, an obvious application of community detection to software engineering is to reveal highly modular packaging of software systems (Fig. 4). The choice of class dependencies (i.e. type of the network) is in that case of course arbitrary. However, simply applying a community detection algorithm to employed networks would often prove useless, as the identified communities would only hardly be mapped to the existing software packages. The latter is vital due to the comprehension of the results. A simple solution is to start with the communities that represent original software packages, and then refine them, using some community detection algorithm. The algorithm should thus merely refine and merge the communities, where no new communities (i.e. labels) should be introduced. This preserves original software packages, their hierarchy and identifiers, which enables complete comprehension of the final results. An example can be seen in Fig. 4.

Another obvious application to software engineering is (network) *abstraction*. Community detection can be employed to obtain a clear representation of software systems on a relatively high level of abstraction. Furthermore, one can also address the *centrality* [57,58] (or other measures of influence) of the identified communities, to expose key nodes and structures

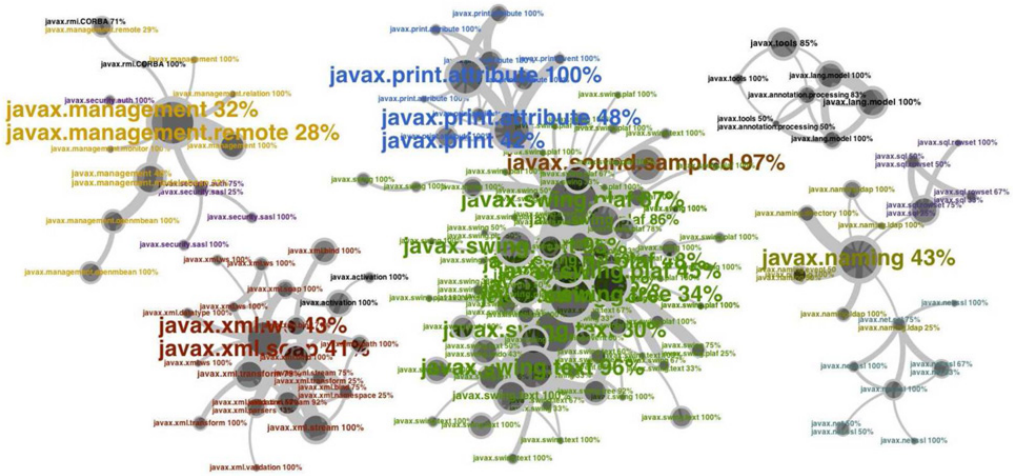


Fig. 5. Community network for the *javax* class dependency network revealed by the *LP* algorithm (only the largest five connected components are shown; modularity equals $Q = 0.7521$). For each community we show the distribution of classes over software packages, where colors indicate high-level packages of the library. The representation gives a clear insight into the structure of the *javax* namespace, and shows relations (i.e. dependencies) among different sub-packages of the system.

throughout the entire system [33,10]. A simple application of community detection to software abstraction can be seen in Fig. 5 (and Fig. 3).

The article represents seminal work in the area of applying network community detection methods and techniques in software engineering. However, further work is needed to design sophisticated applications that would be of considerable benefit in practice.

5. Conclusion

The article explores community structure of networks, constructed from complex software systems (i.e. class dependency networks). The main contribution is in showing that software networks reveal a significant community structure, characterized by similar properties as commonly observed for other complex networks. Software networks thus reveal another general network phenomena, besides scale-free degree distributions and the small-world effect, which is a prominent observation for the software-law problem. Furthermore, the results are of even greater importance, as software represents one of the most complex human made systems.

Future work will mainly focus on considering other types of class dependency networks that will include additional relations among classes. Moreover, we will introduce the notions of *positive* and *negative* relations, to more adequately model similarity and diversity among software classes. Due to the discussed limitations of modularity, the analysis should also be extended to other measures of community structure. The main objective will be to establish further understanding of (community) structure of class dependency networks, and to assess its relation to software packages. The results could thus promote various novel applications in the software engineering domain.

Acknowledgments

The work has been supported by the Slovene Research Agency *ARRS* within the research program P2-0359.

References

[1] D.J. Watts, S.H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (6684) (1998) 440–442.
[2] A. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[3] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, P. Natl. Acad. Sci. USA (2002) 7821–7826.
[4] S.H. Strogatz, Exploring complex networks, Nature 410 (2001) 268.
[5] G. Palla, I. Derényi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society, Nature 435 (2005) 814.
[6] C.R. Myers, Software systems as complex networks: Structure, function, and evolvability of software collaboration graphs, Phys. Rev. E 68 (2).
[7] S. Valverde, R. V Solé, Logarithmic growth dynamics in software networks, Europhys. Lett. 72 (5) (2005) 858–864.
[8] S. Valverde, R.V. Solé, Network motifs in computational graphs: a case study in software architecture, Phys. Rev. E 72 (2) (2005) 026107.
[9] K. Cai, B. Yin, Software execution processes as an evolving complex network, Inform. Sciences 179 (12) (2009) 1903–1928.
[10] G.A. Kohring, Complex dependencies in large software systems, Adv. Complex Syst. 12 (6) (2009) 565–581.
[11] S. Valverde, R.F. Cancho, R.V. Solé, Scale-free networks from optimal design, Europhys. Lett. 60 (4) (2002) 512.
[12] P. Gleiser, L. Danon, Community structure in jazz, Adv. Complex Syst. 6 (4) (2003) 565.
[13] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3-5) (2010) 75–174.
[14] S.E. Schaeffer, Graph clustering, Comput. Sc. Rev. 1 (1) (2007) 27–64.
[15] M.H. Halstead, Elements of Software Science, Elsevier, 1977.

[16] D. Li, Y. Han, J. Hu, Complex network thinking in software engineering, in: Proceedings of the International Conference on Computer Science and Software Engineering, 2008, pp. 264–268.
[17] L. Subelj, M. Bajec, Unfolding network communities by combining defensive and offensive label propagation, in: Proceedings of the International Workshop on the Analysis of Complex Networks, 2010, pp. 87–104.
[18] D. Hyland-Wood, D. Carrington, S. Kaplan, Scale-free nature of java software package, class and method collaboration graphs, Tech. rep., University of Maryland, 2006.
[19] G. Concas, M. Marchesi, S. Pinna, N. Serra, On the suitability of yule process to stochastically model some properties of object-oriented systems, Physica A 370 (2) (2006) 817–831.
[20] N. LaBelle, E. Wallingford, Inter-package dependency networks in open-source software, e-print arXiv:0411096v1.
[21] S. Valverde, R.V. Solé, Hierarchical small worlds in software architecture, Dynam. Cont. Dis. Ser. B 14 (2007) 1–11.
[22] R.V. Solé, R. Ferrer-Cancho, J.M. Montoya, S. Valverde, Selection, tinkering, and emergence in complex networks: crossing the land of tinkering, Complexity 8 (1) (2003) 20–33.
[23] A.A. Gorshenev, Y.M. Pis'mak, Punctuated equilibrium in software evolution, Phys. Rev. E 70 (6) (2004) 1–4.
[24] X. Zheng, D. Zeng, H. Li, F. Wang, Analyzing open-source software systems as complex networks, Physica A 387 (24) (2008) 6190–6200.
[25] G. Baxter, M. Frean, J. Noble, M. Rickerby, H. Smith, M. Visser, H. Melton, E. Tempero, Understanding the shape of java software, in: Proceedings of the ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, 2006, pp. 397–412.
[26] S. Jenkins, S. Kirk, Software architecture graphs as complex networks: a novel partitioning scheme to measure stability and evolution, Inform. Sciences 177 (12) (2007) 2587–2601.
[27] E.A. Variano, J.H. McCoy, H.Lipson, Networks, dynamics, and modularity, Phys. Rev. Lett. 92 (18) (2004) 188701.
[28] R. Guimerà, M. Sales-Pardo, L.A.N. Amaral, Modularity from fluctuations in random graphs and complex networks, Phys. Rev. E 70 (2) (2004) 025101.
[29] N. Kashtan, U. Alon, Spontaneous evolution of modularity and network motifs, P. Natl. Acad. Sci. USA 102 (39) (2005) 13773–13778.
[30] P. Holme, M.E.J. Newman, Nonequilibrium phase transition in the coevolution of networks and opinions, Phys. Rev. E 74 (5) (2006) 056108.
[31] R.K. Pan, S. Sinha, Modular networks emerge from multiconstraint optimization, Phys. Rev. E 76 (2007) 045103.
[32] X. Gu, An evolutionary model for the origin of modularity in a complex gene network, J. Exp. Zool. 312 (2) (2009) 75–82.
[33] A. Zaidman, T. Calders, S. Demeyer, J. Paredaens, Applying webmining techniques to execution traces to support the program comprehension process, in: Proceedings of the European Conference on Software Maintenance and Reengineering, 2005, pp. 134–142.
[34] S. Valverde, R.V. Solé, On the nature of design, in: Complex Engineered Systems, Springer-Verlag, New York, 2006, pp. 72–100.
[35] J. O'Madadhain, D. Fisher, S. White, P. Smyth, Y. biao Boey, Analysis and visualization of network data using JUNG, J. Stat. Softw. 10 (2) (2005) 1–35.
[36] L. Wen, D. Kirk, R.G. Dromey, Software systems as complex networks, in: Proceedings of the IEEE International Conference on Cognitive Informatics, 2007, pp. 106–115.
[37] JUnit testing framework, http://junit.org/.
[38] JavaMail mail and messaging framework, http://java.sun.com/products/javamail/.
[39] Flamingo GUI component suite, https://flamingo.dev.java.net/.
[40] Colt scientific computing library, http://acs.lbl.gov/software/colt/.
[41] Java language, http://java.sun.com/.
[42] M.E.J. Newman, Detecting community structure in networks, Eur. Phys. J. B 38 (2) (2004) 321–330.
[43] A. Clauset, M.E.J. Newman, C. Moore, Finding community structure in very large networks, Phys. Rev. E 70 (6) (2004) 066111.
[44] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (3) (2007) 036106.
[45] M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.
[46] S. Fortunato, M. Barthelemy, Resolution limit in community detection, P. Natl. Acad. Sci. USA (2007) 36–41.
[47] J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters, Internet Math 6 (1) (2009) 29–123.
[48] B.H. Good, Y. deMontjoye, A. Clauset, Performance of modularity maximization in practical contexts, Phys. Rev. E 81 (4) (2010) 046106.
[49] A. Lancichinetti, S. Fortunato, Community detection algorithms: a comparative analysis, Phys. Rev. E 80 (5) (2009) 056117.
[50] L. Danon, A. Díaz-Guilera, J. Duch, A. Arenas, Comparing community structure identification, J. Stat. Mech., P09008.
[51] V.D. Blondel, J. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech., P10008.
[52] J. Mei, S. He, G. Shi, Z. Wang, W. Li, Revealing network communities through modularity maximization by a contraction-dilation method, New J. Phys., 11 (4).
[53] X. Liu, T. Murata, Advanced modularity-specialized label propagation algorithm for detecting communities in networks, Physica A 389 (7) (2009) 1493.
[54] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, D. Parisi, Defining and identifying communities in networks, P. Natl. Acad. Sci. USA 101 (2004) 2658–2663.
[55] W.P. Stevens, G.J. Myers, L.L. Constantive, Structured design, IBM Syst. J. 38 (2) (1999) 231–256.
[56] C. Szyperski, Component Software: Beyond object-Oriented Programming, Addison-Wesley, 1998.
[57] L. Freeman, A set of measures of centrality based on betweenness, Sociometry 40 (1) (1977) 35–41.
[58] L.C. Freeman, Centrality in social networks: conceptual clarification, Soc. Networks 1 (3) (1979) 215–239.

7

Razprava

V disertaciji so v prvi vrsti predstavljene različne metode in tehnike odkrivanja karakterističnih skupin vozlišč v omrežjih na osnovi izmenjave oznak [66]. Natančneje, uravnoteženo izmenjavo oznak [138] v poglavju 3, ki izboljša robustnost osnovnega pristopa, napredno (npr. zadržano) izmenjavo oznak [99] v poglavju 4, ki izboljša natančnost pristopa ter s pomočjo slabljenja oznak prepreči nastanek velike skupine v omrežju, in posplošeno izmenjavo oznak [72] v poglavju 5, ki še razširi pristop na odkrivanje splošnejših skupin. Kot smo omenili že v poglavju 2, lahko predlagane razširitve in izboljšave med seboj tudi poljubno združujemo, dočim pa vse niso vedno nujno potrebne. Na primer, uravnotežena izmenjava oznak pogosto odpravi tudi pojav velike skupine, sploh če le-to uporabimo skupaj z zadržano izmenjavo, zatorej slabljenje oznak v tem primeru ni potrebno [72, 139]. Omenimo, da je sicer ob pomanjkanju znanja o proučevanem omrežju (posplošena) zadržana uravnotežena izmenjava verjetno najprimernejši pristop [73]. V literaturi pa so bile predlagane še številne druge izboljšave izmenjave oznak (glej razdelek 1.2), vendar so primerjave med pristopi razmeroma redke. Tako danes ni jasno kateri se najbolje obnesejo v posameznih vrstah realnih omrežjih, kar odpira različne možnosti za nadaljnje delo.

Delo na odkrivanju skupin vozlišč v omrežjih je objavljeno v [72, 73, 99, 126, 138, 139].

Doktorska disertacija predstavlja tudi pomemben doprinos k razumevanju same izmenjave med vozlišči ter nekaterih sorodnih procesov nad omrežji [150, 151] (npr. oblikovanje mnenj v spletnih družabnih omrežjih). Predlagani pristopi za odkrivanje skupin vozlišč namreč temeljijo na različnih značilnostih (t.j. obnašanju) procesa izmenjave, kar lahko s pridom prenesemo tudi na druga področja.

Pri tem napredna izmenjava oznak prikazuje, da je moč preko preferenc vozlišč (t.j. "moč" širjenja oznake) na izmenjavo vplivati tako, da le-ta doseže ravnovesno

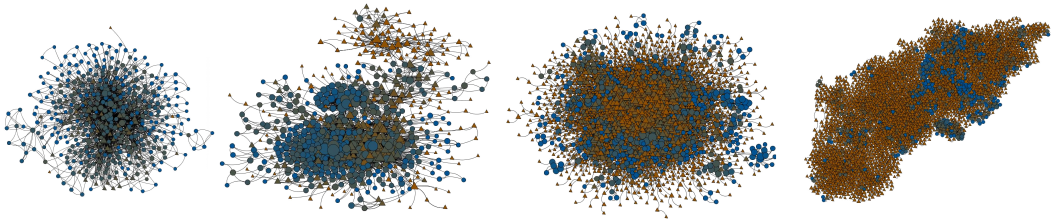
stanje z vnaprej določenimi lastnostmi (glej poglavje 4). Na drugi strani uravnotežena izmenjava oznak temelji na spoznanju, da se vrstni red izmenjave odraža podobno kot preference vozlišč (glej poglavje 3). Na podlagi tega kot preference uvedemo protiuteži vozlišč, ki uravnotežijo sicer naključni vrstni red. Zdi pa se, da je vsaj del slednjega nepotreben. Na primer, izmenjava oznak bi načeloma lahko potekala v vnaprej določenem vrstnem redu, pri čimer bi pristop postal (skoraj) deterministične in očitno robusten [138]. Vendar pa ni jasno, kakšen naj bo tedaj vrstni red. Podobno se zdi, da bi s primernim vrstnim redom izmenjave, protiuteži vozlišč lahko popolnoma opustili. Pri tem velja omeniti, da je izmenjava oznak v omrežju dejansko ekvivalentna *Pottsovemu modelu* [152] (angl. *Potts model*) v statistični mehaniki [122, 136]. Slednje fiziki proučujejo že od 50. let preteklega stoletja, različna dognanja pa bi morda lahko uporabili tudi v zgornjem primeru.

Delo na robustnem odkrivanju skupin vozlišč v omrežjih je objavljeno v [138, 139].

Poleg omenjenega nekateri izmed predlaganih pristopov za odkrivanje skupin vozlišč uporabljajo hierarhično preiskovanje omrežij [99, 126] (glej poglavje 4). Pri tem na podlagi pridobljene razvrstitve v skupine zgradimo *super-omrežje* (angl. *super-network*), kjer skupine predstavimo s *super-vozlišči* (angl. *super-vertex, -node*), ki so povezana, v kolikor obstaja povezava tudi v prvotnem omrežju. Omenjen postopek nato rekurzivno ponavljamo, na koncu pa vrnemo razvrstitev, ki ima npr. najvišjo modularnost Q [69]. Posebej uspešen je pristop, pri katerem upoštevamo tudi značilno *jedrno-obrobno zgradbo* [153] (angl. *core-periphery structure*) številnih realnih omrežij [154]. Pristop najprej razkrije gosto jedro omrežja ter jasne skupine vozlišč na obrobju, postopek pa nato rekurzivno ponovimo nad super-omrežjem, ki predstavlja le jedro prvotnega [99]. Omenjena pristopa hierarhijo skupin gradita od *spodaj navzgor* (angl. *bottom-up*). Uspešnost pa lahko še občutno izboljšamo tako, da skupine od *zgoraj navzdol* (angl. *top-down*) naprej delimo v manjše [73], razvrstitev pa sicer določimo po metodi *največjega verjetja* [155] (angl. *maximum likelihood*).

Delo na hierarhičnem odkrivanju skupin vozlišč v omrežjih je objavljeno v [73, 99, 126].

Posplošena izmenjava oznak razširi osnovni pristop na bolj splošne skupine vozlišč, t.j. skupnosti in module (glej poglavje 5). Kot smo omenili že v poglavju 2, je pri tem pomembno, da vnaprej presodimo, kje v omrežju je moč pričakovati določeno vrsto skupin. Na primer, izjemno uspešna je strategija na osnovi *popravljene nakopičenosti* [145] (angl. *degree-corrected clustering*), ki meri gostoto omrežja v okolici posameznega vozlišča, pri čimer upošteva tudi odvisnosti med stopnjami sosednjih vozlišč [146]. Skupnosti namreč navadno najdemo v delih omrežja z visoko nakopičenostjo, prav nasprotno pa velja za module [73, 139] (glej sliko 7.1). Pouda-



Slika 7.1: Metabolično omrežja ogorčice “C. elegans” [156], omrežje odvisnosti dela programskega jezika “Java” [117], beljakovinsko omrežje kvasovke “S. cerevisiae” [46] ter električno omrežje zahodnega dela Amerike [22] (od leve proti desni). Skupnosti in module vozlišč navadno najdemo v delih omrežij z visoko (modro) oziroma nizko (oranžno) nakopičenostjo [145] (glej tekst). Slika je povzeta po [73].

rimo, da je omenjen pristop vsaj primerljiv z najboljšimi drugimi v literaturi [72, 73], vendar je računanje nakopičenosti v velikih omrežjih lahko relativno zahtevno. Pri tem pa teorija in različne analize nakazujejo, da bi bilo podobno uspešnost morda moč doseči tudi s pomočjo analize odvisnosti med vozlišči ter nekaterih drugih lastnosti realnih omrežij [73, 144], kar ponuja številne možnosti za nadaljnje delo.

Delo na analizi skupin vozlišč v realnih omrežjih je objavljeno v [72, 73, 139].

Med verjetno najočitnejše primere omrežij, v katerih najdemo tako jasne skupnosti kot tudi module, pa sodijo omrežja odvisnosti kompleksnega programja [73, 117], ki jih obravnavamo v poglavju 6. Pri tem povezava med paketi pripadajočih programskih knjižnic ter različnimi skupinami vozlišč v programskih omrežjih omogoča številne primere uporabe [117, 120]. Poudarimo, da je zgradbo programja sicer moč proučevati tudi preko drugih splošnih lastnosti realnih omrežij [120]. Na primer, potenčna porazdelitev [23] vhodnih stopenj vozlišč nakazuje na veliko ponovno uporabnost programja, dočim pa v primeru izhodnih stopenj slednje predstavlja razmeroma zapleteno programsko kodo. Podobno lahko pojav malega sveta [22] v programskih omrežjih pogosto enačimo z dobro zasnovano in strukturirano kodo, ter z značilnostmi same domene. Na drugi strani pa je moč tudi lastnosti posameznih vozlišč v programskih omrežjih povezati z različnimi značilnostmi programja [116, 120]. Praktično orodje za proučevanje kompleksnega programja na osnovi analize omrežij bi tako lahko predstavljalo pomemben doprinos v programskem inženirstvu.

Delo na analizi programskih omrežij je objavljeno v [72, 73, 117, 120, 139].

8

Zaključek

Doktorska disertacija se ukvarja predvsem z odkrivanjem in proučevanjem karakterističnih skupin vozlišč v realnih omrežjih. Predstavljene so različne metode in tehnike odkrivanja skupin na osnovi izmenjave oznak med vozlišči [66], ki močno izboljšajo robustnost in natančnost osnovnega pristopa [99, 138], hkrati pa pristop razširijo še na odkrivanje splošnejših skupin vozlišč [72] (t.j. skupnosti in modulov). Predlagane razširitve in izboljšave tako skupaj predstavljajo celovit pristop, pri tem pa ne zahtevajo nikakršnega predhodnega znanja o zgradbi omrežja (npr. število ali vrsto skupin). Uspešnost na sintetičnih in realnih omrežjih je vsaj primerljiva z najboljšimi pristopi v literaturi [73, 99], asimptotična časovna zahtevnost pa je blizu idealne [99] (t.j. linearne). Slednje je v primeru velikih realnih omrežij ključno. Poleg tega so predlagani pristopi enostavni za razumevanje in implementacijo ter omogočajo vključitev poljubnega dodatnega znanja o proučevani domeni. Z omenjenimi lastnostmi se lahko ponašajo le redki drugi pristopi v literaturi [54], tako pričujoča disertacija predstavlja velik napredek na področju odkrivanja karakterističnih skupin vozlišč v omrežjih ter hkrati pomemben doprinos k znanosti.

Disertacija se osredotoča tudi na proučevanje skupin vozlišč v omrežjih objektnoumerjenega programja. To verjetno predstavlja eno od najkompleksnejših človeških stvaritev, kljub temu pa je le malo znanega o zgradbi dobrega programja [104]. Analize sicer kažejo, da programska omrežja vsebujejo izrazite skupine vozlišč, ki razmeroma dobro pojasnijo njihovo zgradbo [73, 117], hkrati pa hierarhije skupin (delno) sovpadajo z organizacijo pripadajočih programskih knjižnic [72, 117]. Pri tem prvo predstavlja pomemben doprinos k razumevanju zgradbe velikih realnih omrežij [73] (na nivoju skupin vozlišč), slednje pa omogoča številne praktične primere uporabe v programskem inženirstvu ter tudi drugod [117, 120].

Nadaljnje delo na področju disertacije bo temeljilo na izhodiščih, ki smo jih v večini že nakazali v poglavju 7. V prvi vrsti bodo med seboj primerjani in kritično ovrednoteni različni pristopi izmenjave oznak, ki so bili predstavljeni v literaturi. Trenutno je v teku analiza 25 različnih izboljšav in razširitev osnovnega pristopa, nabor pa bo v prihodnosti še razširjen na npr. prekrivajoče skupine. Poleg tega bo podrobneje proučena povezava med nekaterimi različicami osnovnega pristopa (npr. vrstni red izmenjave), kar lahko omogoči razvoj še učinkovitejših algoritmov.

Delo pa se bo nadaljevalo tudi na proučevanju povezave med karakterističnimi skupinami vozlišč ter drugimi lastnostmi omrežij. Slednje lahko pripelje do izboljšanih pristopov za odkrivanje splošnih skupin vozlišč, ter do razvoja modelov naključnih omrežij, ki bi lahko pojasnili obstoj različnih skupin v realnih omrežjih. Posebna pozornost bo namenjena programskim omrežjem, ter primerom uporabe v programskem inženirstvu, trenutno delo na področju pa je predstavljeno v [73].

— Lovro Šubelj, Ljubljana, 2013

Literatura

[1] L. Euler, "Solutio problematis ad geometriam situs pertinentis", *Commentarii Academiae Scientiarum Imperialis Petropolitanae*, št. 8, str. 128–140, 1736.

[2] M. Newman, *Networks: An Introduction*. Oxford: Oxford University Press, 2010.

[3] F. Harary, *Graph Theory*. Cambridge: Perseus, 1995.

[4] B. Bollobás, *Modern Graph Theory*. Heidelberg: Springer, 1998.

[5] J. L. Moreno, *Who Shall Survive?* Beacon: Beacon House, 1934.

[6] W. W. Zachary, "An information flow model for conflict and fission in small groups", *Journal of Anthropological Research*, št. 33, zv. 4, str. 452–473, 1977.

[7] A. Davis, B. B. Gardner & M. R. Gardner, *Deep South*. Chicago: Chicago University Press, 1941.

[8] M. Girvan & M. E. J. Newman, "Community structure in social and biological networks", *Proceedings of the National Academy of Sciences of United States of America*, št. 99, zv. 12, str. 7821–7826, 2002.

[9] X. Liu & T. Murata, "Community detection in large-scale bipartite networks", v *Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, Milano, Italy, str. 50–57, 2009.

[10] S. Milgram, "The small world problem", *Psychology Today*, št. 1, zv. 1, str. 60–67, 1967.

[11] L. Freeman, "A set of measures of centrality based on betweenness", *Sociometry*, št. 40, zv. 1, str. 35–41, 1977.

[12] M. S. Granovetter, "The strength of weak ties", *American Journal of Sociology*, št. 78, zv. 6, str. 1360–1380, 1973.

[13] S. Wasserman & K. Faust, *Social Network Analysis*. Cambridge: Cambridge University Press, 1994.

[14] J. Scott, *Social Network Analysis: A Handbook*. London: Sage, 2. izd., 2000.

[15] D. Cartwright & F. Harary, "Structural balance: A generalization of Heider's theory", *Psychological Review*, št. 63, zv. 5, str. 277–293, 1956.

[16] P. Erdős & A. Rényi, "On random graphs I", *Publicationes Mathematicae Debrecen*, št. 6, str. 290–297, 1959.

[17] R. Hobbs & M. Lombardi, *Mark Lombardi: Global Networks*. New York: Independent Curators International, 2003.

[18] D. J. d. S. Price, "Networks of scientific papers", *Science*, št. 149, str. 510–515, 1965.

[19] F. R. Pitts, "A graph theoretic approach to historical geography", *Professional Geographer*, št. 17, zv. 5, str. 15–20, 1965.

[20] J. G. White, E. Southgate, J. N. Thomson & S. Brenner, "The structure of the nervous system of the nematode Caenorhabditis elegans", *Philosophical Transactions of the Royal Society B: Biological Sciences*, št. 314, zv. 1165, str. 1–340, 1986.

[21] J. D. Pelletier, "Self-organization and scaling relationships of evolving river networks", *Journal of Geophysical Research*, št. 104, zv. 4, str. 7359–7375, 1999.

[22] D. J. Watts & S. H. Strogatz, "Collective dynamics of 'small-world' networks", *Nature*, št. 393, zv. 6684, str. 440–442, 1998.

[23] A. L. Barabási & R. Albert, "Emergence of scaling in random networks", *Science*, št. 286, zv. 5439, str. 509–512, 1999.

[24] M. E. J. Newman, "The physics of networks", *Physics Today*, št. 61, zv. 11, str. 33–38, 2008.

[25] **L. Šubelj**, S. Žitnik, A. Zrnec, A. Kumer, B. Klemenc, D. Lavbič & M. Bajec, "Lastnosti velikih realnih omrežij in primeri uporabe", v *Zbornik Konference Dnevi Slovenske Informatike*, str. 10, 2012.

[26] **L. Šubelj**, N. Blagus, Š. Furlan, B. Klemenc, A. Kumer, D. Lavbič, A. Zrnec, S. Žitnik & M. Bajec, "Analiza kompleksnih omrežij: Osnovni pojmi in primeri uporabe v praksi", v *Zbornik Konference Informatika v Javni Upravi*, str. 10, 2010.

[27] J. O'Madadhain, D. Fisher, S. White, P. Smyth & Y.-B. Boey, "Analysis and visualization of network data using JUNG", *Journal of Statistical Software*, št. 10, zv. 2, str. 1–35, 2005.

[28] L. A. Adamic & N. Glance, "The political blogosphere and the 2004 U.S. election", v *Proceedings of the KDD Workshop on Link Discovery*, Chicago, IL, USA, str. 36–43, 2005.

[29] "Opte Project". http://www.opte.org/, 2003.

[30] U. Kang, C. E. Tsourakakis, A. P. Appel, C. Faloutsos & J. Leskovec, "Radius plots for mining tera-byte scale graphs: Algorithms, patterns, and observations", v *Proceedings of the SIAM International Conference on Data Mining*, Columbus, OH, USA, 2010.

[31] V. Batagelj & A. Mrvar, "Pajek: Program for large network analysis", *Connections*, št. 21, zv. 2, str. 47–57, 1998.

[32] W. de Nooy, A. Mrvar & V. Batagelj, *Exploratory Social Network Analysis with Pajek*. Cambridge: Cambridge University Press, 2005.

[33] M. Bastian, S. Heymann & M. Jacomy, "Gephi: An open source software for exploring and manipulating networks", v *Proceedings of the AAAI International Conference on Weblogs and Social Media*, San Jose, CA, USA, str. 361–362, 2009.

[34] "Network Workbench (NWB)". http://nwb.cns.iu.edu/, 2013.

[35] "Stanford Network Analysis Project (SNAP)". http://snap.stanford.edu/, 2013.

[36] A. A. Hagberg, D. A. Schult & P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX", v *Proceedings of the Python in Science Conference*, Pasadena, CA, USA, str. 11–15, 2008.

[37] D. Hansen, B. Shneiderman & M. A. Smith, *Analyzing Social Media Metworks with NodeXL: Insights from a Connected World*. Burlington: Morgan Kaufmann, 2010.

[38] R. Albert, H. Jeong & A. L. Barabasi, "Error and attack tolerance of complex networks", *Nature*, št. 406, zv. 6794, str. 378–382, 2000.

[39] L. Backstrom, P. Boldi, M. Rosa, J. Ugander & S. Vigna, "Four degrees of separation", v *Proceedings of the ACM International Conference on Web Science*, Evanston, IL, USA, str. 45–54, 2012.

[40] J.-D. J. Han, N. Bertin, T. Hao, D. S. Goldberg, G. F. Berriz, L. V. Zhang, D. Dupuy, A. J. M. Walhout, M. E. Cusick, F. P. Roth & M. Vidal, "Evidence for dynamically organized modularity in the yeast protein-protein interaction network", *Nature*, št. 430, zv. 6995, str. 88–93, 2004.

[41] L. C. Freeman, "Centrality in social networks: Conceptual clarification", *Social Networks*, št. 1, zv. 3, str. 215–239, 1979.

[42] M. E. J. Newman, S. H. Strogatz & D. J. Watts, "Random graphs with arbitrary degree distributions and their applications", *Physical Review E*, št. 64, zv. 2, str. 026118, 2001.

[43] P. J. Laurienti, K. E. Joyce, Q. K. Telesford, J. H. Burdette & S. Hayasaka, "Universal fractal scaling of self-organized networks", *Physica A: Statistical Mechanics and its Applications*, št. 390, zv. 20, str. 3608–3613, 2011.

[44] N. Blagus, **L. Šubelj** & M. Bajec, "Self-similar scaling of density in complex real-world networks", *Physica A: Statistical Mechanics and its Applications*, št. 391, zv. 8, str. 2794–2802, 2012.

[45] P. Gleiser & L. Danon, "Community structure in jazz", *Advances in Complex Systems*, št. 6, zv. 4, str. 565–573, 2003.

[46] G. Palla, I. Derényi, I. Farkas & T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society", *Nature*, št. 435, zv. 7043, str. 814–818, 2005.

[47] L. K. Gallos, H. A. Makse & M. Sigman, "A small world of weak ties provides optimal global integration of self-similar modules in functional brain networks", *Proceedings of the National Academy of Sciences of United States of America*, št. 109, zv. 8, str. 2825–2830, 2012.

[48] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai & A. L. Barabási, "Hierarchical organization of modularity in metabolic networks", *Science*, št. 297, zv. 5586, str. 1551–1555, 2002.

[49] A. Lancichinetti, M. Kivela, J. Saramaki & S. Fortunato, "Characterizing the community structure of complex networks", *PLoS ONE*, št. 5, zv. 8, str. e11976, 2010.

[50] G. W. Flake, S. Lawrence, C. L. Giles & F. M. Coetzee, "Self-organization and identification of Web communities", *IEEE Computer*, št. 35, zv. 3, str. 66–71, 2002.

[51] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto & D. Parisi, "Defining and identifying communities in networks", *Proceedings of the National Academy of Sciences of United States of America*, št. 101, zv. 9, str. 2658–2663, 2004.

[52] L. Danon, A. Díaz-Guilera, J. Duch & A. Arenas, "Comparing community structure identification", *Journal of Statistical Mechanics: Theory and Experiment*, št. P09008, 2005.

[53] M. A. Porter, J.-P. Onnela & P. J. Mucha, "Communities in networks", *Notices of the American Mathematical Society*, št. 56, zv. 9, str. 1082–1097, 2009.

[54] S. Fortunato, "Community detection in graphs", *Physics Reports: Review Section of Physics Letters*, št. 486, zv. 3-5, str. 75–174, 2010.

[55] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski & D. Wagner, "On finding graph clusterings with maximum modularity", v *Proceedings of the International Workshop on Graph Theoretic Concepts in Computer Science*, Dornburg, Germany, str. 121–132, 2007.

[56] S. E. Schaeffer, "Graph clustering", *Computer Science Review*, št. 1, zv. 1, str. 27–64, 2007.

[57] B. W. Kernighan & S. Lin, "An efficient heuristic procedure for partitioning graphs", *Bell System Technical Journal*, št. 49, zv. 2, str. 291–308, 1970.

[58] L. Donetti & M. A. Munoz, "Detecting network communities: A new systematic and efficient algorithm", *Journal of Statistical Mechanics: Theory and Experiment*, št. P10012, 2004.

[59] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices", *Physical Review E*, št. 74, zv. 3, str. 036104, 2006.

[60] M. E. J. Newman & E. A. Leicht, "Mixture models and exploratory analysis in networks", *Proceedings of the National Academy of Sciences of United States of America*, št. 104, zv. 23, str. 9564, 2007.

[61] E. M. Airoldi, D. M. Blei, S. E. Fienberg & E. P. Xing, "Mixed membership stochastic blockmodels", *Journal of Machine Learning Research*, št. 9, zv. 9, str. 1981, 2008.

[62] A. Clauset, M. E. J. Newman & C. Moore, "Finding community structure in very large networks", *Physical Review E*, št. 70, zv. 6, str. 066111, 2004.

[63] J. Duch & A. Arenas, "Community detection in complex networks using extremal optimization", *Physical Review E*, št. 72, zv. 2, str. 027104, 2005.

[64] J. Reichardt & S. Bornholdt, "Statistical mechanics of community detection", *Physical Review E*, št. 74, zv. 1, str. 016110, 2006.

[65] P. Ronhovde & Z. Nussinov, "Local resolution-limit-free Potts model for community detection", *Physical Review E*, št. 81, zv. 4, str. 046114, 2010.

[66] U. N. Raghavan, R. Albert & S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks", *Physical Review E*, št. 76, zv. 3, str. 036106, 2007.

[67] M. Rosvall & C. T. Bergstrom, "Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems", *PLoS ONE*, št. 6, zv. 4, str. e18209, 2011.

[68] V. D. Blondel, J.-L. Guillaume, R. Lambiotte & E. Lefebvre, "Fast unfolding of communities in large networks", *Journal of Statistical Mechanics: Theory and Experiment*, št. P10008, 2008.

[69] M. E. J. Newman & M. Girvan, "Finding and evaluating community structure in networks", *Physical Review E*, št. 69, zv. 2, str. 026113, 2004.

[70] S. Fortunato & M. Barthelemy, "Resolution limit in community detection", *Proceedings of the National Academy of Sciences of United States of America*, št. 104, zv. 1, str. 36–41, 2007.

[71] B. H. Good, Y. A. de Montjoye & A. Clauset, "Performance of modularity maximization in practical contexts", *Physical Review E*, št. 81, zv. 4, str. 046106, 2010.

[72] **L. Šubelj** & M. Bajec, "Ubiquitousness of link-density and link-pattern communities in real-world networks", *European Physical Journal B*, št. 85, zv. 1, str. 32, 2012.

[73] **L. Šubelj** & M. Bajec, "Clustering assortativity, communities and functional modules in real-world networks", *e-print arXiv:12082518v1*, str. 1–21, 2012.

[74] T. Nishikawa & A. E. Motter, "Discovering network structure beyond communities", *Scientific Reports*, št. 1, zv. 151, str. 1–7, 2011.

[75] B. Long, X. Xu, Z. Zhang & P. Yu, "Community learning by graph approximation", v *Proceedings of the IEEE International Conference on Data Mining*, Omaha, NE, USA, str. 232–241, 2007.

[76] J. Reichardt & D. R. White, "Role models for complex networks", *European Physical Journal B*, št. 60, zv. 2, str. 217–224, 2007.

[77] F. Lorrain & H. C. White, "Structural equivalence of individuals in social networks", *Journal of Mathematical Sociology*, št. 1, zv. 1, str. 49–80, 1971.

[78] D. R. White & K. P. Reitz, "Graph and semigroup homomorphisms on networks of relations", *Social Networks*, št. 5, zv. 2, str. 193–234, 1983.

[79] M. G. Everett & S. P. Borgatti, "Regular equivalence: General theory", *Journal of Mathematical Sociology*, št. 19, zv. 1, str. 29–52, 1994.

[80] P. Doreian, V. Batagelj & A. Ferligoj, *Generalized Blockmodeling*. Cambridge: Cambridge University Press, 2005.

[81] H. C. White, S. A. Boorman & R. L. Breiger, "Social structure from multiple networks: Block models of roles and positions", *American Journal of Sociology*, št. 81, zv. 4, str. 730–779, 1976.

[82] C.-Y. Lin, J.-L. Koh & A. L. P. Chen, "A better strategy of discovering link-pattern based communities by classical clustering methods", v *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Hyderabad, India, str. 56–67, 2010.

[83] B. Karrer & M. E. J. Newman, "Stochastic blockmodels and community structure in networks", *Physical Review E*, št. 83, zv. 1, str. 016107, 2011.

[84] S. Pinkert, J. Schultz & J. Reichardt, "Protein interaction networks: More than mere modules", *PLoS Computional Biology*, št. 6, zv. 1, str. e1000659, 2010.

[85] M. Rosvall & C. T. Bergstrom, "An information-theoretic framework for resolving community structure in complex networks", *Proceedings of the National Academy of Sciences of United States of America*, št. 104, zv. 18, str. 7327–7331, 2007.

[86] C. Ding, T. Li, W. Peng & H. Park, "Orthogonal nonnegative matrix tri-factorizations for clustering", v *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Philadelphia, PA, USA, str. 126–135, 2006.

[87] S. P. Borgatti & M. G. Everett, "Two algorithms for computing regular equivalence", *Social Networks*, št. 15, zv. 4, str. 361–376, 1993.

[88] V. Batagelj & A. Ferligoj, "Clustering relational data", v *Data Analysis*, str. 3–15, Berlin: Springer, 2000.

[89] A. Žiberna, "Generalized blockmodeling of valued networks", *Social Networks*, št. 29, zv. 1, str. 105–126, 2007.

[90] D. Chakrabarti, S. Papadimitriou, D. S. Modha & C. Faloutsos, "Fully automatic cross-associations", v *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, USA, str. 79–88, 2004.

[91] K. Henderson, T. Eliassi-Rad, S. Papadimitriou & C. Faloutsos, "HCDF: A hybrid community discovery framework", v *Proceedings of the SIAM International Conference on Data Mining*, Columbus, OH, USA, str. 754–765, 2010.

[92] T. S. Evans & R. Lambiotte, "Line graphs, link partitions and overlapping communities", *Physical Review E*, št. 80, zv. 1, str. 016105, 2009.

[93] Y.-Y. Ahn, J. P. Bagrow & S. Lehmann, "Link communities reveal multiscale complexity in networks", *Nature*, št. 466, zv. 7307, str. 761–764, 2010.

[94] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii & U. Alon, "Network motifs: Simple building blocks of complex networks", *Science*, št. 298, zv. 5594, str. 824–827, 2001.

[95] N. Pržulj, D. A. Wigle & I. Jurisica, "Functional topology in a network of protein interactions", *Bioinformatics*, št. 20, zv. 3, str. 340–348, 2004.

[96] M. E. J. Newman, "Communities, modules and large-scale structure in networks", *Nature Physics*, št. 8, zv. 1, str. 25–31, 2012.

[97] A. Barrat, M. Barthelemy & A. Vespignani, *Dynamical Processes on Complex Networks*. Cambridge: Cambridge University Press, 2008.

[98] A. Vespignani, "Modelling dynamical processes in complex socio-technical systems", *Nature Physics*, št. 8, zv. 1, str. 32–39, 2012.

[99] **L. Šubelj** & M. Bajec, "Unfolding communities in large complex networks: Combining defensive and offensive label propagation for core extraction", *Physical Review E*, št. 83, zv. 3, str. 036103, 2011.

[100] C. R. Myers, "Software systems as complex networks: Structure, function, and evolvability of software collaboration graphs", *Physical Review E*, št. 68, zv. 2, str. 046116, 2003.

[101] D. Hyland-Wood, D. Carrington & S. Kaplan, "Scale-free nature of Java software package, class and method collaboration graphs", v *Proceedings of the International Symposium on Empirical Software Engineering*, Rio de Janeiro, Brazil, str. 1–10, 2006.

[102] N. LaBelle & E. Wallingford, "Inter-package dependency networks in open-source software", v *Proceedings of the International Conference on Complex Systems*, Boston, MA, USA, str. 1–8, 2006.

[103] S. Valverde & R. V. Solé, "Hierarchical small worlds in software architecture", *Dynamics of Continuous, Discrete and Impulsive Systems: Series B*, št. 14, str. 1–11, 2007.

[104] K.-Y. Cai & B.-B. Yin, "Software execution processes as an evolving complex network", *Information Sciences*, št. 179, zv. 12, str. 1903–1928, 2009.

[105] S. Valverde, R. F. Cancho & R. V. Solé, "Scale-free networks from optimal design", *Europhysics Letters*, št. 60, zv. 4, str. 512, 2002.

[106] Y. Gao, G. Xu, Y. Yang, J. Liu & S. Guo, "Disassortativity and degree distribution of software coupling networks in object-oriented software systems", v *Proceedings of the IEEE International Conference on Progress in Informatics and Computing*, Shanghai, China, str. 1000–1004, 2010.

[107] S. Valverde & R. V. Solé, "Logarithmic growth dynamics in software networks", *Europhysics Letters*, št. 72, zv. 5, str. 858–864, 2005.

[108] M. A. Fortuna, J. A. Bonachela & S. A. Levin, "Evolution of a modular software network", *Proceedings of the National Academy of Sciences of United States of America*, št. 108, zv. 50, str. 19985–19989, 2011.

[109] S. Valverde & R. V. Solé, "Network motifs in computational graphs: A case study in software architecture", *Physical Review E*, št. 72, zv. 2, str. 026107, 2005.

[110] G. Baxter, M. Frean, J. Noble, M. Rickerby, H. Smith, M. Visser, H. Melton & E. Tempero, "Understanding the shape of Java software", v *Proceedings of the ACM International Conference on Object-Oriented Programming, Systems, Languages, and Applications*, Portland, OR, USA, str. 397–412, 2006.

[111] M. E. J. Newman, "Assortative mixing in networks", *Physical Review Letters*, št. 89, zv. 20, str. 208701, 2002.

[112] M. E. J. Newman, "Mixing patterns in networks", *Physical Review E*, št. 67, zv. 2, str. 026126, 2003.

[113] A. A. Gorshenev & Y. M. Pis'mak, "Punctuated equilibrium in software evolution", *Physical Review E*, št. 70, zv. 6, str. 1–4, 2004.

[114] K. He, R. Peng, J. Liu, F. He, P. Liang & B. Li, "Design methodology of networked software evolution growth based on software patterns", *Journal of Systems Science and Complexity*, št. 19, zv. 2, str. 157–181, 2006.

[115] A. Zaidman, T. Calders, S. Demeyer & J. Paredaens, "Applying webmining techniques to execution traces to support the program comprehension process", v *Proceedings of the European Conference on Software Maintenance and Reengineering*, Manchester, United Kingdom, str. 134–142, 2005.

[116] G. A. Kohring, "Complex dependencies in large software systems", *Advances in Complex Systems*, št. 12, zv. 6, str. 565–581, 2009.

[117] **L. Šubelj** & M. Bajec, "Community structure of complex software systems: Analysis and applications", *Physica A: Statistical Mechanics and its Applications*, št. 390, zv. 16, str. 2968–2975, 2011.

[118] L. G. Moyano, M. L. Mouronte & M. L. Vargas, "Communities and dynamical processes in a complex software network", *Physica A: Statistical Mechanics and its Applications*, št. 390, zv. 4, str. 741–748, 2011.

[119] S. Valverde & R. V. Solé, "On the nature of design", v *Complex Engineered Systems*, str. 72–100, New York: Springer-Verlag, 2006.

[120] **L. Šubelj** & M. Bajec, "Software systems through complex networks science: Review, analysis and applications", v *Proceedings of the KDD Workshop on Software Mining*, Beijing, China, str. 9–16, 2012.

[121] D. Li, Y. Han & J. Hu, "Complex network thinking in software engineering", v *Proceedings of the International Conference on Computer Science and Software Engineering*, Wuhan, China, str. 264–268, 2008.

[122] G. Tibély & J. Kertész, "On the equivalence of the label propagation method of community detection and a Potts model approach", *Physica A: Statistical Mechanics and its Applications*, št. 387, zv. 19-20, str. 4982–4984, 2008.

[123] I. X. Y. Leung, P. Hui, P. Liò & J. Crowcroft, "Towards real-time community detection in large networks", *Physical Review E*, št. 79, zv. 6, str. 066107, 2009.

[124] S. Gregory, "Finding overlapping communities in networks by label propagation", *New Journal of Physics*, št. 12, zv. 10, str. 103018, 2010.

[125] Z.-H. Wu, Y.-F. Lin, S. Gregory, H.-Y. Wan & S.-F. Tian, "Balanced multi-label propagation for overlapping community detection in social networks", *Journal of Computer Science and Technology*, št. 27, zv. 3, str. 468–479, 2012.

[126] **L. Šubelj** & M. Bajec, "Unfolding network communities by combining defensive and offensive label propagation", v *Proceedings of the ECML PKDD Workshop on the Analysis of Complex Networks*, Barcelona, Spain, str. 87–104, 2010.

[127] M. J. Barber & J. W. Clark, "Detecting network communities by propagating labels under constraints", *Physical Review E*, št. 80, zv. 2, str. 026129, 2009.

[128] X. Liu & T. Murata, "Advanced modularity-specialized label propagation algorithm for detecting communities in networks", *Physica A: Statistical Mechanics and its Applications*, št. 389, zv. 7, str. 1493, 2009.

[129] X. Liu & T. Murata, "Detecting communities in K-partite K-uniform (hyper)networks", *Journal of Computer Science and Technology*, št. 26, zv. 5, str. 778–791, 2011.

[130] S. Pang, C. Chen & T. Wei, "A realtime clique detection algorithm: Time-based incremental label propagation", v *Proceedings of the International Conference on Intelligent Information Technology Application*, Nanchang, China, str. 459–462, 2009.

[131] J. Soman & A. Narang, "Fast community detection algorithm with GPUs and multicore architectures", v *Proceedings of the IEEE International Parallel Distributed Processing Symposium*, Anchorage, AK, USA, str. 568 –579, 2011.

[132] B. S. Rees & K. B. Gallagher, "Detecting overlapping communities in complex networks using swarm intelligence for multi-threaded label propagation", v *Proceedings of the International Workshop on Complex Networks*, Melbourne, FL, USA, str. 111–119, 2012.

[133] C. Pang, F. Shao, R. Sun & S. Li, "Detecting community structure in networks by propagating labels of nodes", v *Proceedings of the International Symposium on Neural Networks*, Wuhan, China, str. 839–846, 2009.

[134] J. Xie & B. K. Szymanski, "Community detection using a neighborhood strength driven label propagation algorithm", v *Proceedings of the IEEE International Workshop on Network Science*, West Point, NY, USA, str. 188–195, 2011.

[135] G. Cordasco & L. Gargano, "Label propagation algorithm: A semi–synchronous approach", *International Journal of Social Network Mining*, št. 1, zv. 1, str. 3–26, 2011.

[136] P. Boldi, M. Rosa, M. Santini & S. Vigna, "Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks", v *Proceedings of the International World Wide Web Conference*, Hyderabad, India, str. 587–596, 2011.

[137] P. Boldi, M. Rosa & S. Vigna, "Robustness of social networks: Comparative results based on distance distributions", v *Proceedings of the International Conference on Social Informatics*, Singapore, Republic of Singapore, str. 8–21, 2011.

[138] **L. Šubelj** & M. Bajec, "Robust network community detection using balanced propagation", *European Physical Journal B*, št. 81, zv. 3, str. 353–362, 2011.

[139] **L. Šubelj** & M. Bajec, "Generalized network community detection", v *Proceedings of the ECML PKDD Workshop on Finding Patterns of Human Behaviors in Network and Mobility Data*, Athens, Greece, str. 66–84, 2011.

[140] I. A. Kovács, R. Palotai, M. S. Szalay & P. Csermely, "Community landscapes: An integrative approach to determine overlapping network module hierarchy, identify key nodes and predict network dynamics", *PLoS ONE*, št. 5, zv. 9, str. e12528, 2010.

[141] G. Cordasco & L. Gargano, "Community detection via semi-synchronous label propagation algorithms", v *Proceedings of the IMSAA Workshop on Business Applications of Social Network Analysis*, Bangalore, India, str. 1–8, 2010.

[142] S. Brin & L. Page, "The anatomy of a large-scale hypertextual Web search engine", *Computer Networks and ISDN Systems*, št. 30, zv. 1-7, str. 107–117, 1998.

[143] P. Bonacich, "Power and centrality: A family of measures", *American Journal of Sociology*, št. 92, zv. 5, str. 1170–1182, 1987.

[144] D. V. Foster, J. G. Foster, P. Grassberger & M. Paczuski, "Clustering drives assortativity and community structure in ensembles of networks", *Physical Review E*, št. 84, zv. 6, str. 066117, 2011.

[145] S. N. Soffer & A. Vázquez, "Network clustering coefficient without degree-correlation biases", *Physical Review E*, št. 71, zv. 5, str. 057101, 2005.

[146] M. E. J. Newman & J. Park, "Why social networks are different from other types of networks", *Physical Review E*, št. 68, zv. 3, str. 036122, 2003.

[147] M. Savić, M. Radovanović & M. Ivanović, "Community detection and analysis of community evolution in Apache Ant class collaboration networks", v *Proceedings of the Balkan Conference in Informatics*, Novi Sad, Serbia, str. 229–234, 2012.

[148] W. P. Stevens, G. J. Myers & L. L. Constantive, "Structured design", *IBM Systems Journal*, št. 38, zv. 2, str. 231–256, 1999.

[149] S. Jenkins & S. Kirk, "Software architecture graphs as complex networks: A novel partitioning scheme to measure stability and evolution", *Information Sciences*, št. 177, zv. 12, str. 2587–2601, 2007.

[150] R. A. Holley & T. M. Liggett, "Ergodic theorems for weakly interacting infinite systems and the voter model", *Annals of Probability*, št. 3, zv. 4, str. 643–663, 1975.

[151] R. Pastor-Satorras & A. Vespignani, "Epidemic spreading in scale-free networks", *Physical Review Letters*, št. 86, zv. 14, str. 3200–3203, 2001.

[152] R. Potts, "The mathematical investigation of some cooperative phenomena", PhD thesis, Oxford University, Oxford, 1951.

[153] G. Siganos, S. L. Tauro & M. Faloutsos, "Jellyfish: A conceptual model for the AS Internet topology", *Journal of Communications and Networks*, št. 8, zv. 3, str. 339–350, 2006.

[154] J. Leskovec, K. J. Lang, A. Dasgupta & M. W. Mahoney, "Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters", *Internet Mathematics*, št. 6, zv. 1, str. 29–123, 2009.

[155] G. Casella & R. L. Berger, *Statistical Inference*. Belmont: Duxbury Press, 1990.

[156] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai & A.-L. Barabási, "The large-scale organization of metabolic networks", *Nature*, št. 407, zv. 6804, str. 651–654, 2000.

Izjavljam, da sem pričujočo doktorsko disertacijo izdelal samostojno pod vodstvom mentorja izr. prof. dr. Marka Bajca.

— Lovro Šubelj, Ljubljana, 2013