Statična in dinamična analiza omrežij na podlagi lokalnih vzorcev

Matija Polajnar

doktorska disertacija

predana

Fakulteti za računalništvo in informatiko

kot del izpolnjevanja pogojev za pridobitev naziva

doktor znanosti

s področja

računalništva in informatike

Ljubljana, 2014

IZJAVA

Izjavljam, da sem avtor dela in da slednje ne vsebuje materiala, ki bi ga kdorkoli predhodno že objavil ali oddal v obravnavo za pridobitev naziva na univerzi ali na drugem visokošolskem zavodu, razen v primerih, kjer so navedeni viri.

— Matija Polajnar —

september 2014

Oddajo so odobrili

dr. Janez Demšar

izredni profesor za računalništvo in informatiko

MENTOR IN ČLAN OCENJEVALNE KOMISIJE

dr. Igor Kononenko

profesor za računalništvo in informatiko

PREDSEDNIK OCENJEVALNE KOMISIJE

dr. Sergio Cabello Justo

izredni profesor za računalniško matematiko

ZUNANJI ČLAN OCENJEVALNE KOMISIJE

Fakulteta za matematiko in fiziko, Univerza v Ljubljani

PREDHODNA OBJAVA

Izjavljam, da so bili rezultati obravnavane raziskave predhodno objavljeni/sprejeti za objavo v recenzirani reviji ali javno predstavljeni v naslednjih primerih:

[1] M. Polajnar, J. Demšar. Small network completion using frequent subnetworks. *Intelligent Data Analysis*, 2014.

Potrjujem, da sem pridobil pisna dovoljenja vseh lastnikov avtorskih pravic, ki mi dovoljujejo vključitev zgoraj navedenega materiala v pričujočo disertacijo. Potrjujem, da zgoraj navedeni material opisuje rezultate raziskav, izvedenih v času mojega podiplomskega študija na Univerzi v Ljubljani.

…mi mirno plavala bi moja barka …
—F. Prešeren

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*
Matija Polajnar
Statična in dinamična analiza omrežij na podlagi lokalnih vzorcev

POVZETEK

V disertaciji analiziramo uporabnost dveh vrst lokalnih vzorcev v omrežjih – t.i. (označenih) pogostih vzorcev ter grafkov – za raziskovanje časovnega razvoja omrežja ter napovedovanje nadaljnjega razvoja oziroma odkrivanje manjkajočih delov.

Analiza kompleksno strukturiranih podatkov, kot so omrežja, zaradi razvoja strojne opreme v zadnjih dveh desetletjih, zaradi enostavnejšega zajema podatkov ter zaradi konkretnih potreb industrije vzbuja vedno več pozornosti raziskovalcev. Spletne družbene storitve, na primer, so na eni strani zelo bogat vir podatkov v obliki omrežij, po drugi strani pa zelo hvaležen odjemalec dobrih hipotez o nadaljnjem razvoju omrežja. Podatke o družbenih mrežah s pridom uporabljajo tudi raziskovalci s področja družboslovnih znanosti, prav tako analiza omrežij veliko prispeva k razvoju znanosti na področju biologije.

Naše delo je bilo motivirano s konkretno aplikativno potrebo po odkrivanju manjkajočih elementov v majhnih označenih omrežjih. V literaturi tak problem še ni bil obravnavan, zato smo morali najprej postaviti teoretične temelje in izpostaviti razlike napram drugim do zdaj obravnavanim problemom. Zaradi ključnih razlik smo predlagali tudi drugačno metodologijo merjenja napovedne uspešnosti, ki testna omrežja gradi z odstranjevanjem po ene povezave iz dane množice omrežij. Definirali smo še metriko uspešnosti na podlagi ranga pravilne napovedi. Utemeljitev problema in postavitev metodologije merjenja uspešnosti štejemo za pomembna prispevka tega dela k znanosti.

Reševanja problema smo se lotili z zasnovo družine metod, ki kot model domene uporabljajo označene pogoste vzorce. Napovedi, tj. hipoteze glede morebitnih manjkajočih elementov v omrežju, izdelujejo in vrednotijo z delnim vpenjanjem pogostih vzorcev v domnevno nepopolno omrežje. Na realnih in sintetičnih množicah izmerjena uspešnost kaže na koristnost ne le zasnovane družine metod, temveč posebne obrav-

nave tako definiranega problema nasploh. Uveljavljene metode, ki rešujejo splošnejše probleme, so namreč pri odkrivanju manjkajočih elementov v majhnih označenih omrežjih dosegle bistveno nižjo uspešnost.

Metoda ni namenjena analizi večjih omrežij. Že pri zmerno velikih omrežjih jo začne dušiti časovna potratnost, poskusi pa kažejo, da začne padati tudi njena napovedna točnost. To je pričakovano, saj metoda ni bila zasnovana za večja omrežja; za te so bolj primerne v obstoječi literaturi opisane metode.

Posvetili smo se še eni vrsti lokalnih vzorcev, ki so se najprej uveljavili na področju bioinformatike, a so v splošnem primerni za analizo lokalnih podstruktur v velikih neusmerjenih neoznačenih omrežjih: grafkom. Problem opazovanja razvoja grafkov v rastočem omrežju skozi čas smo najprej osvetlili s teoretičnega vidika. Pokazali smo, kako se strukture lahko razvijajo iz ene v drugo ter pripravili matematična in algoritmična orodja za empirično obravnavo več realnih in sintetičnih omrežij. Pokazali smo, da določene značilnosti v razvoju veljajo splošno za vsa analizirana omrežja, v drugih pa se ta ločijo med seboj.

Zasnovali smo tudi družino metod za napovedovanje nadaljnjega časovnega razvoja omrežja na podlagi analize dotedanjega razvoja grafkov. Ocena napovedne uspešnosti ne daje rezultatov, na podlagi katerih bi lahko zasnovane algoritme priporočili za praktično uporabo. Na vseh analiziranih omrežjih je obstoječa metoda naključnih sprehodov s ponovnimi začetki dosegla višjo napovedno uspešnost. Tudi pri zlaganju napovedi napovedi razvite družine metod ne prinesejo dodane vrednosti. So pa rezultati poskusov namignili, da bi bilo mogoče napovedno uspešnost metode naključnih sprehodov izboljšati z zlaganjem njenih napovedi pri uporabi različnih vrednosti parametrov.

Ključne besede: analiza omrežij, nadzorovano učenje, pogosti vzorci, grafki

University *of Ljubljana*
Faculty *of Computer and Information Science*
Matija Polajnar
Static and Dynamic Analysis of Networks Based on Local Patterns

ABSTRACT

In this work, we attempt to determine applicability of two kinds of local network patterns, i.e. labelled frequent patterns and graphlets, to the problems of network evolution analysis and prediction of either further temporal development of the network or its missing elements.

Networks and other data with complex structure are attracting ever more attention in the research community. This is partly due to large network analysis finally being feasible with the hardware, developed in the last two decades. Beside that, methods to collect large amounts of network-structured data have also emerged, with industry-fueled need for its analysis following closely. For instance, World Wide Web-based social networking services offer unprecedented insight into social strucutures, and on the other hand provide strong incentive for development of methods that analyse networks and predict their future evolution. The data gathered by those services are a rich source for social sciences researchers, and network analysis also provides useful tools for scientists in the field of biology.

Our work was motivated by our own need of a method to generate sensible hypotheses about missing elements in small labeled networks. No research of such a problem has been done before in the published literature. Therefore, we began by establishing theoretical foundations of the problem by formally defining it and pointing out similarities and differences with regard to other similar problems. Key differences required us to propose a new prediction quality evaluation methodology that constructs test networks by removing an edge from some network in a given set in every possible way. We defined an evaluation metrics based on the rank of correct prediction as well. The justification of the problem definition and the set up of the prediction quality evaluation methodology are important scientific contributions of this work.

We propose a family of methods to solve the defined problem that use frequent pat-

terns to model the domain. Hypotheses about missing elements in the network are constructed and scored by looking for partial embeddings of frequent patterns in what is considered to be an incomplete network. Results of evaluation on real and synthetic data sets indicate not only the usefulness of the proposed family of methods, but also the need to consider this type of problem separately from previously researched network analysis and completion problems. Methods, proposed in the available literature, that solve the latter, performed far worse in our small network completion problem.

The proposed method is not suitable for large network analysis. Its time complexity begins to hinder its usefulness in slightly larger networks. Experiments show that its prediction quality declines with network size as well. That is an expected outcome as the method was not designed to handle large networks.

While working with local network patterns, another type of them caught our attention: graphlets. Those have been first considered in the field of bioinformatics, but their use has since spread to other domains. We considered the process of graphlet evolution in growing networks from a theoretical viewpoint first. We showed how structures can transform into one another and defined mathematical and algorithmic tools for an empirical analysis of multiple real and synthetic networks. We showed how some evolution properties are shared among all analysed networks and others are domain-specific.

Also proposed in this doctoral dissertation is a method for prediction of further temporal evolution of the network, based on analysis of graphlet evolution. Evaluation results do not indicate its usefulness for practical applications. On all analysed networks, the well-known method named *Random Walk with Restarts* (RWR) achieved better prediction. The predictions generated by our method also did not aid with prediction stacking. The evaluation results did, however, surprisingly show that stacking the RWR predictions, obtained using different parameter values, can increase prediction quality.

Key words: network analysis, supervised learning, frequent patterns, graphlets

OPRAVIČILO IN ZAHVALA

Iskreno se opravičujem Janezu in Evi, ker nekaterih projektov nisem izpeljal v dogovorjenih rokih. Zahvaljujem se vama za potrpežljivost in razumevanje.

— Matija Polajnar, Ljubljana, september 2014.

KAZALO

I

Uvod

Z grafi – diskretnimi matematičnimi strukturami, ki jih sestavljajo *vozlišča* in *povezave* med njimi – je moč opisati mnoge kompleksne naravne, tehnične in družboslovne pojave. Osebe, naprave, fizikalne delce, spojine, proteine, kraje in druge entitete popišemo z množico vozlišč. Povezave med njimi predstavljajo odnose, na katere se pri zadani nalogi osredotočamo: prijateljstva, sodelovanja, podobnosti, kemijske vezi, cestne ali telekomunikacijske povezave, odvisnosti in drugo. Na ta način zajamemo odnose, ki jih težko pregledno opišemo v obliki vektorjev, kar je najpogostejša oblika zapisa podatkov, ki jih želimo analizirati [1].

Od prve objave na področju teorije grafov v prvi polovici 18. stoletja, v kateri je Euler reševal problem obhoda, ki danes nosi njegovo ime, preko prve omembe pojma *graf* v tem pomenu leta 1878 pa vse do danes so bili grafi deležni obširne teoretične obravnave. Tako poznamo različne vrste in lastnosti grafov, kar ustreza različnim vrstam in lastnostim odnosov med entitetami. Graf oziroma njegove povezave so lahko usmerjeni ali neusmerjeni. Graf imenujemo drevo, če opisuje strogo hierarhičen odnos. Graf je dvodelen, če popisuje odnose med dvema množicama različnih entitet, ki se nikoli ne povezujejo znotraj iste množice. Graf je lahko povezan ali nepovezan. Če je usmerjen, ločimo še šibko in krepko povezanost. Graf je ravninski, če ga lahko narišemo v dvodimenzionalno ravnino brez križanja povezav. Graf je poln, če ima vse možne povezave. V teoriji grafov so povezave lahko tudi *utežene*, kar pomeni, da je vsaki dodeljeno realno število. Podobno so vozlišča lahko *pobarvana*, torej obstaja (običajno neinjektivna) preslikava iz množice vozlišč v diskretno množico barv.

Te in številne druge lastnosti nam omogočajo analizo velikih matematično definiranih grafov. Pri praktični uporabi metod za *odkrivanje zakonitosti v podatkih* iščemo resnico v naravnih odnosih, ki niso strogo matematično urejeni. Kljub temu je teoretično ozadje nepogrešljivo pri zasnovi algoritmov in njihovi analizi.

Pri popisu odnosov, ki se pojavljajo v naravi in družbi, si po potrebi dovolimo še več svobode. Tako vozliščem kot povezavam lahko dodelimo diskretne ali zvezne vrednosti, lahko pa tudi po več atributov. (Diskretnim vrednostim v tem kontekstu ne rečemo barve, temveč oznake, angl. *labels*.) V literaturi se za grafe z dodatnimi informacijami na gradnikih – ali pa zgolj grafe, katerih topologija ni strogo matematično urejena – uporablja izraz *omrežje*.

Analiza podatkov v obliki omrežij ima korenine v začetku prejšnjega stoletja v sociologiji [2, 3]. Popisovanje in analiza sta seveda potekala ročno, zato je šlo za manjša omrežja. Računalniška analiza omrežij se je zaradi omejene pomnilniške zmogljivosti

pojavila nekoliko kasneje kot klasično odkrivanje zakonitosti v podatkih, a je predvsem v začetku tega stoletja doživela razcvet. Temu je močno botrovala tudi konkretna gospodarska potreba po orodjih za analizo ter za napovedovanje prihodnjega razvoja omrežij.

1.1 Vsebina dela

Področje analize omrežij je raziskovalno izjemno široko in še zdaleč ni izčrpano. V pričujočem delu opisujemo naš prispevek k temu področju, ki ga je motivirala praktična potreba znotraj raziskovalne skupine. Kot podrobneje opisujemo v poglavju 3, smo potrebovali metodo za izdelavo hipotez o manjkajočih elementih v majhnih označenih omrežjih. Metoda, ki bi jo lahko neposredno uporabili, še ni obstajala, zato v sledečih poglavjih natančno opredeljujemo problem, ki ga želimo rešiti, definiramo metodo, ki ga rešuje, ter tehnike za ocenjevanje uspešnosti.

Predmet svojega raziskovanja smo poimenovali *problem dopolnjevanja majhnih omrežij*. Zanima nas, kako bi lahko postavili in ovrednotili hipoteze o morebitnih manjkajočih elementih v majhnem omrežju, za katerega utemeljeno sumimo, da sestoji iz podobnih struktur (fragmentov) kot druga omrežja iz iste domene. Vprašanje postavljamo ob bok v literaturi definiranim problemom dopolnjevanja (velikih) omrežij. Metode za reševanje teh problemov se osredotočajo predvsem na topološke značilnosti omrežja in ga skušajo dopolniti, kvaliteto napovedi pa ocenjujemo z merami za dvojiško uvrščanje. Pri majhnih omrežjih topologija ne pride do izraza, prav tako se v enem omrežju skriva premalo informacij za učenje in je treba pridobljeno znanje prenašati med omrežji znotraj domene.

Podrobno in formalno bomo opisali novo metodo za reševanje takega problema. Kot model, ki bo opisoval lastnosti obravnavane domene, bomo uporabili vzorce (fragmente), ki se v njej pogosto pojavljajo. Te lahko izluščimo z uporabo obstoječih metod. Hipoteze glede manjkajočih gradnikov omrežja bomo postavljali in vrednotili z vpenjanjem odkritih vzorcev v novo omrežje.

V tem delu bomo utemeljili tudi, zakaj je ocenjevanje kvalitete napovedi z AUC in drugimi merami za dvojiško uvrščanje pri zadanem problemu nemogoče. Mero uspešnosti dopolnjevanja majhih omrežij bomo osnovali na rangu edine pravilne napovedi pri posameznem testnem primeru (omrežju).

Predstavili bomo rezultate ocenjevanja napovedne uspešnosti, s katerimi lahko primerjamo svojo metodo s tistimi, zasnovanimi za velika omrežja, ter preprosto referenč-

no metodo. Na vseh uporabljenih majhnih realnih in sintetičnih omrežjih naša metoda izkazuje opazno, pogosto pa celo bistveno boljšo napovedno uspešnost. Slabost našega pristopa se pokaže pri obravnavi večjih omrežij. Če smo zmerni (do nekaj deset povezav na omrežje, a odvisno od gostote), metoda zgolj začne izgubljati napovedno uspešnost, medtem ko jo metode za velika omrežja pridobivajo. Še večja omrežja delo z metodo onemogočijo zaradi previsoke časovne zahtevnosti. Ta je namreč eksponentna glede na velikost vhoda.

Pri raziskovanju problema dopolnjevanja majhnih omrežij so se označeni pogosti vzorci v omrežjih izkazali kot uporaben pripomoček za analizo lokalne strukture le-teh. Zanimalo nas je, če lahko v podobnem duhu tudi v večjih omrežjih koristno uporabimo katero od definicij pogostih vzorcev v takem okolju. Osredotočili smo se na t.i. *grafke*.

V tem delu bomo postavili teoretične temelje za opazovanje razvoja takih struktur v velikem omrežju skozi čas. Predstavili bomo rezultate empirične analize več omrežij, ki smo jo izdelali s pomočjo obstoječih algoritmov, za naše potrebe spremenjenih z matematično utemeljenimi prilagoditvami. Pokazali bomo, katere lastnosti so skupne vsem analiziranim omrežjem, v katerih pa se jih da ločiti.

Dopolnjevanje velikih omrežij je zelo aktivno raziskovalno področje tudi ali predvsem zaradi sodobnih aplikativnih potreb. Analizo časovnega razvoja grafkov smo želeli dopolniti v način za napovedovanje prihodnosti omrežja. Predstavili bomo matematično utemeljeno metodo, ki pa se pri ocenjevanju napovedne uspešnosti ni izkazala. Pri nekaterih podatkovnih množicah je bila bistveno slabša od trenutno ene najboljših obstoječih metod – naključnih sprehodov s ponovnimi začetki –, pri drugih pa se ji je zgolj približala.

Če ima metoda vseeno uporabno vrednost, smo ugotavljali z zlaganjem napovedi te ter metode naključnih sprehodov. Izkazalo se je sicer, da tudi v takem kontekstu metoda ne prinaša ključne prednosti. Smo pa pri poskusih naleteli na pozitivno presenečenje, o katerem bomo poročali na koncu vsebinskega dela disertacije. Kaže namreč, da je napovedi metode naključnih sprehodov možno izboljšati z zlaganjem napovedi te metode pri več različnih vrednostih parametra.

1.2 Prispevki k znanosti

Ta doktorska disertacija je v grobem razdeljena na dve večji poglavji. Izvirne prispevke k znanosti naštevamo za vsakega od le-teh posebej.

- Problem dopolnjevanja majhnih omrežij obravnavamo v poglavju 3. Prispevki k znanosti v njem so naslednji.
  - Definicija problema, podkrepljena z motivacijo, in uvrstitev v kontekst sorodnih problemov. Problem se od slednjih razlikuje predvsem po vrsti vhodnih podatkov (množica učnih omrežij z oznakami ter novo omrežje, nad katerim gradimo hipoteze, namesto bolj običajnega enega omrežja).
  - Definicija nove mere uspešnosti in metodologija za njeno merjenje, vključno s preprosto referenčno metodo.
  - Poglavitni prispevek je družina metod, ki rešuje zadani problem z uporabo v domeni odkritih pogostih vzorcev in v poskusih izkazuje bistveno višjo uspešnost od referenčne metode, pa tudi od drugih v literaturi dostopnih metod, ki ne znajo uporabiti vseh pri reševanju tega problema dostopnih informacij.
- S časovnim razvojem grafkov v omrežju se ukvarjamo v poglavju 4. Tam k razvoju znanosti prispevamo sledeče.
  - Teoretično ogrodje za obravnavo časovnega razvoja grafkov v omrežju.
  - Empirično analizo razvoja grafkov več realnih in sintetičnih omrežjih, ki izpostavlja nekatere lastnosti, ki so obče ali pa se pojavljajo le v določenih vrstah omrežij.
  - Teoretično utemeljeno družino metod, ki za napovedovanje nadaljnjega časovnega razvoja omrežja uporablja informacijo o časovnem razvoju grafkov v tem omrežju. V poskusih se metode niso izkazale z zavidljivo uspešnostjo, kar pušča odprta vprašanja in možnosti nadaljnjega raziskovanja.

2

Pregled področja

Na področjih odkrivanja zakonitosti v podatkih (angl. *data mining*) in strojnega učenja (anlg. *machine learning*) je bila večina vloženega raziskovalnega dela usmerjena v analizo strogo strukturiranih podatkov, predvsem v tabelarični obliki, kjer za vsak *primer* (vrstico) popišemo vrednosti *atributov* (stolpcev). V zadnjih letih se povečuje zanimanje za analizo kompleksnejših in manj strukturiranih podatkov [4]. Ena od vrst takih podatkov so tudi omrežja: grafi – diskretne strukture, sestavljene iz vozlišč in povezav – ki so jim dodane informacije v obliki oznak.

Grafi so diskretne matematične strukture, sestavljene iz vozlišč ter povezav med njimi. Omrežje je sestavljeno iz grafa ter dodatnih podatkov, kot so oznake ali številske vrednosti na vozliščih in/ali povezavah. V literaturi se izraz *omrežje* (tudi *kompleksno omrežje*) pogosto uporablja tudi za grafe brez dodatnih podatkov, če njihova topologija ni „trivialna". V to skupino spadajo tudi grafi, ki popisujejo sociološke, biološke, tehnološke in druge naravne procese. V tem dokumentu bomo izraza *graf* in *omrežje* uporabljali kot sopomenki in posebej pojasnili, kakšne strukture opisujemo, ko bo to pomembno za razumevanje.

Analiza omrežij je zanimiva zaradi obilice podatkov, ki jih je primerno shranjevati v tej obliki. V to kategorijo spadajo podatkovni tokovi, molekule, interakcije med proteini in metabolične mreže, družbena omrežja, kamor prištevamo tudi omrežja soavtorstev, topologija internetnih vozlišč in spletnih strani, odvisnosti med komponentami programskega ali strojnega sistema, sheme elektronskih vezij itd. Prav tako lahko kot omrežje zapišemo poljubno m-krat-n relacijo (formalno: $R \subseteq A \times B, |A| = m, |B| = n$), na primer relacijo med kupci in artikli, ki so jih kupili. Z različnimi oznakami na povezavah lahko znotraj enega omrežja hkrati popišemo tudi več relacij, če na primer želimo v prej omenjenem omrežju opisati še družinske relacije med kupci.

Odvisno od vrste podatkov obravnavamo lahko zgolj eno omrežje; v tem primeru gre tipično za veliko omrežje z nekaj tisoč do nekaj sto tisoč vozlišči. Druga možnost je, da imamo opravka z množico (običajno manjših) neodvisnih omrežij. Lahko opazujemo tudi zaporedje omrežij, ki predstavlja razvoj določenega omrežja z naraščanjem neke spremenljivke, ki jo iz očitnih razlogov poimenujemo *čas*.

2.1 Globalne lastnosti

Globalne statistične lastnosti realnih omrežij so predmet raziskav že relativno dolgo. Dajo nam osnovne informacije o strukturi grafa. Z upoštevanjem lastnosti, ki so bile najdene v vseh realnih omrežjih, si lahko pomagamo pri generiranju umetnih podatkov.

Eno znanih lastnosti, ki jo razvije večina omrežij, imenujemo *small world*; pomeni, da je kljub temu, da večina parov vozlišč ni povezanih, najkrajša razdalja med vsakim parom vozlišč majhna. Lastnost je bila opažena že pred več kot štirimi desetletji [5].

Opaženo je bilo tudi, da mnoga realna omrežja ustrezajo t.i. potenčnemu zakonu. To pomeni, da so stopnje vozlišč (tako vhodne kot izhodne) porazdeljene po potenčni porazdelitvi, torej imajo *dolge repe*: obstaja več vozlišč s stopnjami, ki so mnogo večje od povprečne stopnje, kot bi jih pričakovali pri normalni ali eksponentni porazdelitvi. Takim omrežjem rečemo *brezlestvična omrežja* (angl. *scale-free networks* [6]) in en od mehanizmov nastanka omrežij, katerega rezultat so, je *prednostna povezanost* (angl. *preferential attachment* [7]) oziroma *bogati bogatijo* (angl. *the rich get richer*). V kontekstu omrežij to pomeni, da je verjetnost, da se bo nova povezava v omrežju pripela na določeno vozlišče, premo sorazmerna s stopnjo tega vozlišča.

Tudi v časovnem razvoju grafov so opazili podobne trende v globalnih lastnostih [8]. Število povezav v mnogih realnih grafih narašča potenčno s številom vozlišč pri fiksnem eksponentu strogo večjim od 1 in strogo manjšim od 2, torej se grafi gostijo, hkrati pa s povečevanjem grafa v nasprotju z intuicijo polmer pada. Opaženo je bilo, da tipično večina vozlišč grafa oblikuje veliko povezano komponento (tako v realnih kot generiranih grafih), v uteženih grafih pa uteži s časom ne rastejo enakomerno, temveč v sunkih [9].

2.2 Lokalni vzorci

Osnovna gradnika omrežij sta vozlišče in povezava. Vsakega izmed teh lahko opišemo z nekaterimi topološkimi lastnostmi. Pri vozliščih lahko na primer opazujemo njihovo stopnjo, torej število povezav, v katerih vozlišče nastopa kot krajišče. Že v prejšnjem razdelku smo opisali pomembnost porazdelitve te količine v celotnem omrežju.

Za namen primerjanja omrežij in razvrščanja v gruče so v literaturi definirane še tri vrste nekoliko večjih lokalnih struktur. V večjih neoznačenih omrežjih je Milo s sodelavci [10] definiral *motive* (angl. *motifs*), Pržuljeva s sodelavci [11] pa *grafke* (angl. *graphlets*). Že nekaj let pred tem, ob prelomu tisočletja, sta Kuramochi in Karypis [12] po vzoru *pogostih naborov* (angl. *frequent itemsets*) definirala in odkrivala *pogoste vzorce* (angl. *frequent patterns*) v množicah manjših označenih omrežij. Koncept sta kasneje Bringmann in Nijssen [13] posplošila še na iskanje vzorcev v enem nekoliko večjem označenem omrežju.

2.2.1 *Motivi*

Motivi so inducirani podgrafi, ki so v nekem omrežju *značilni* v smislu, da se v njem pojavijo statistično značilno večkrat kot bi pričakovali, če bi bilo omrežje naključno [10]. Definirani so za usmerjena omrežja. Prvotne raziskave so zajemale motive velikosti do 4 vozlišč, kasnejše pa so s hitrejšimi algoritmi iskale podgrafe do velikosti 7 vozlišč [14–16]. Odkriti so bili nekateri motivi, ki se pojavljajo v različnih realnih omrežjih (npr. usmerjen cikel dolžine 3), in drugi, ki se pojavljajo le v specifičnih vrstah omrežij (npr. usmerjen cikel dolžine 4, ki je značilen za umetne strukture, kot so elektronska vezja in na svetovni splet, ne pa za biološka omrežja). Največ zanimanja so motivi deležni na področju biologije, kjer se je sprva domnevalo, da omrežja genske regulacije s sorodnimi funkcijami vsebujejo enake motive [17] oziroma obratno – kar bi bilo še bolj praktično uporabno – omrežja s podobnimi motivi popisujejo sorodne funkcije [18]. Nekatere kasnejše raziskave so sicer prinesle dvom v to hipotezo [19].

Algoritem, ki so ga razvili utemeljitelji koncepta motivov [10], le-te najde z *golo silo*: izčrpno našteje vse inducirane podgrafe, nato izomorfne združi, postopek ponovi na (običajno vsaj 1000) naključnih omrežjih, pridobljenih z velikim številom naključnih sprememb originalnega omrežja, ki ohranjajo nekatere njegove lastnosti, ter primerja koncentracije. Postopek lahko močno pohitrimo tako, da motivov ne iščemo v celotnem omrežju, pač pa v naključno izbranih podomrežjih, kot predlaga Kashtan s sodelavci [14]. Wernicke je kasneje opozoril na pristranskost Kashtanovega vzorčenja in predlagal drugačno, namesto iskanja motivov v velikem številu naključnih omrežij pa ponuja eksplicitne formule [15]. S takim pristopom se tudi izognemo problemu, kako izbrati število potrebnih naključnih sprememb v omrežju, da lahko trdimo, da smo izdelali naključno omrežje.

2.2.2 *Grafki*

Pri analizi neusmerjenih omrežij, še posebej omrežij interakcij med proteini (angl. *protein-protein interaction, PPI*), so se kot orodje uveljavili *grafki* [11]. Grafek je vsak povezan induciran podgraf do določene velikosti, ne glede na to, kako pogoso se v omrežju pojavi. Običajno se omejimo na velikost 4 ali 5 vozlišč: večji grafki se zaradi lastnosti majhnega sveta raztezajo že preko celotnega omrežja, prav tako pa se preko razumnih meja poveča časovna zahtevnost odkrivanja takih grafkov.

Vektor števila pojavitev različnih (neizomorfnih) grafkov lahko uporabimo za pri-

merjavo struktur omrežij [11], če pa ločimo tudi različne vloge (orbite avtomorfizma) vozlišč, pa za vsako vozlišče dobimo dodatno lastnost, vektor, ki je v nekem smislu posplošitev stopnje vozlišča [20]. Komponentam vektorja zato rečemo *grafkovne stopnje*. Podobno lahko opazujemo tudi vloge povezav. To nam omogoči gručenje povezav, ki ima pri iskanju funkcionalno povezanih podomrežij to prednost pred običajno uporabljenim gručenjem vozlišč, da so vozlišča lahko uvrščena tudi na presek dveh podomrežij, kar pogosto sovpada z njihovo vlogo v opisovanem procesu [21]. Pojme grafka, orbite vozlišča in orbite povezave smo podrobno razložili v 4. poglavju, kjer smo jih uporabili kot orodje za analizo in dopolnjevanje omrežij.

Primat na področju algoritmov za odkrivanje grafkov je do nedavnega držala skupina, ki jih je utemeljila (Pržuljeva s sodelavci [11]). Leta 2012 je Bhuiyan s sodelavci izdelal metodo za enakomerno vzorčenje grafkov [22], ki deluje na osnovi Markovskih verig (naključnih sprehodov) in omogoča štetje grafkov v zelo velikih omrežjih – v manj kot dveh minutah so analizirali omrežje z več kot 15 milijoni povezav. Metoda oceni zgolj število vseh vrst grafkov v celotnem omrežu, ne izračuna pa grafkovnih stopenj vozlišč. Leto kasneje sta Hočevar in Demšar problem prevedla na reševanje sistema enačb, s čimer število grafkov ter grafkovne stopnje vozlišč lahko izračunamo za velikostni red hitreje kot z izčrpnim naštevanjem [23].

2.2.3 *Pogosti vzorci*

Pogosti vzorci v množici označenih grafov so (običajno majhni) označeni povezani grafi, ki so *podgrafno izomorfni* (angl. *subgraph isomorphic*) vsaj določenemu minimalnemu deležu grafov iz te množice [24]. V primeru analize usmerjenih omrežij zadošča šibka povezanost vzorca. Podgrafni izomorfizem pomeni, da je graf izomorfen nekemu podgrafu drugega grafa. V kontekstu označenih grafov se morajo pri tem ujemati tudi oznake na vozliščih in povezavah.

Pogosti podgrafi so bili v literaturi uporabljeni za gručenje grafov, kot značilke za uvrščanje, za indeksiranje in hitro iskanje v grafih, kompresijo zapisa grafov ipd. Z velikostjo podgrafov njihovo število (torej dimenzionalnost iskalnega prostora) eksponentno raste: če je nek podgraf v dani domeni pogost, so pogosti tudi vsi njegovi podgrafi. Posledica eksponentne rasti dimenzionalnosti prostora je omejena uporabnost podgrafov za značilke zaradi t.i. prekletstva dimenzionalnosti. Iz istega vzroka, pa tudi zaradi NP-polnosti problema iskanja izomorfizma, je iskanje pogostih podgrafov računsko zelo zahtevno.

Prvi učinkoviti algoritmi za odkrivanje pogostih podgrafov so se pojavili okrog leta 2000. Inokuchijev [24] in Kuramochijev [12] pristop uporabljata princip algoritma Apriori [25]: večje kandidate za pogoste podgrafe generirata z združevanjem dveh pogostih podgrafov manjše velikosti. Zaradi računske zahtevnosti takšnega združevanja druga skupina pristopov (npr. gSpan [26] za množice poljubnih grafov in Dagma [27] za množice usmerjenih acikličnih grafov) generira večje vzorce s povečevanjem manjših (za eno povezavo).

Obe kategoriji algoritmov izkoriščata z velikostjo podgrafa monotono padajočo (tj. anti-monotono) mero podpore, da lahko med preiskovanjem prostora *odrežeta* podprostore, katerih podpora je zagotovo pod določeno mejo minimalne podpore. Najpogosteje uporabljena mera podpore v množicah grafov je število učnih grafov, ki vsebujejo dan podgraf. V primeru analize enega (večjega) grafa je izračun monotono padajoče podpore težji problem [28].

Z izkoriščanjem monotono padajoče mere podpore omilimo problem časovne zahtevnosti iskanja podgrafov kot posledice eksponentnega naraščanja števila možnih podgrafov. Še vedno ostane problem prekletstva dimenzionalnosti, zato so bile razvite tudi tehnike iskanja specifičnih vrst podgrafov:

- *zaprti* pogosti podgrafi [29] so tisti, ki imajo v dani domeni višjo podporo od vseh grafov, v katerih so vsebovani,
- *maksimalni* pogosti podgrafi [30] zadoščajo pogoju, da noben od grafov, v katerih so vsebovani, v dani domeni ni pogost,
- *pomembni* pogosti podgrafi [31] pa dajo dovolj visoke vrednosti dane *ciljne funkcije*.

Če predpostavimo, da graf s časom zgolj raste, tj. dobiva nova vozlišča in povezave, lahko celotno zaporedje zapišemo z enim grafom, katerega povezave označimo s časom njihove prve pojavitve. V takem grafu lahko najdemo vzorce časovnega razvoja: po vzoru iskanja asociativnih pravil v atributnih podatkovnih množicah algoritem GERM (*Graph Evolution Rules Miner* [32]) obravnava najdene pogoste podgrafe kot pravila in jih razdeli na *glavo* in *telo*. Glava je s telesom enolično določena tako, da pri slednjem odstranimo povezave, ki so se pojavile najkasneje.

2.3 Generiranje podatkov

Z generatorji grafov lahko pridobimo vpogled v mehanizme, katerih rezultat so grafi z določenimi lastnostmi, hkrati pa generiramo podatke, na katerih simuliramo algoritme za analizo in napovedovanje. Za generator rečemo, da generira *realistične* grafe, če imajo rezultirajoči grafi nekatere ali vse globalne lastnosti, opisane v razdelku 2.1 [33]. Generiramo lahko naključne grafe, ki oponašajo neko lastnost opazovanega realnega grafa, a so večji (ko želimo nove postopke, npr. nove omrežne protokole, testirati na stanju omrežja čez nekaj let) ali manjši (ko želimo dobiti rezultat algoritma, ki na velikem grafu ne deluje, na primer zaradi prevelike prostorske ali časovne zahtevnosti).

Najzgodnejši algoritem za generiranje naključnih grafov je model Erdös-Rényi [34], za katerega se je kasneje izkazalo, da ne gradi realističnih grafov. Pred dobrim desetletjem so bile objavljene prve posplošitve tega modela, ki upoštevajo določene odkrite statistične lastnosti realnih grafov. Primer je PLRG [35], ki zagotavlja potenčno porazdelitev stopenj vozlišč in ne nerealistične Poissonove, ki jo izkazujejo grafi, generirani z izvirnim Erdös-Rényi algoritmom.

Bolj napredna družina generatorjev grafov deluje po že omenjenem principu prednostne povezanosti. Najosnovnejši je *Barabási-Albertov model* [6]. Gradi omrežja, ki imajo potenčno porazdelitev vozlišč z eksponentom 3. Obstajajo posplošitve tega modela, pri katerih si lahko izberemo poljuben eksponent, večji ali enak 2.

Alternativna razlaga pojava potenčnih porazdelitev v grafih je optimizacija virov [36]. Primer je povezovanje krajev, upoštevajoč njihovo geografsko lego, v internetno omrežje. Pri predpostavki, da vsak kraj povežemo zgolj z enim drugim krajem, želimo skleniti ustrezen kompromis med dolžino kabla in zakasnitvijo povezave, tj. številom stikal, ki mesto ločijo od ostalih. Tudi pri takem načinu gradnje grafa so stopnje vozlišč porazdeljene potenčno [37].

Z metodami, ki temeljijo na tenzorskih produktih (R-MAT [38] in metoda s Kroneckerjevim produktom [39]), lahko generiramo grafe, ki ustrezajo več globalnim lastnostim oziroma so zelo *podobni* danemu realnemu grafu. Razširitev metode s Kroneckerjevim produktom, RTM [40], generira s časom *samo-podobno* rastoče *utežene* grafe.

Hkrati z vpeljavo pojma pogostih vzorcev v množicah označenih grafov sta Kuramochi in Karypis z namenom performančnega testiranja svojega algoritma predlagala tudi generator takih množic [12]. Tako kot njun algoritem za odkrivanje pogostih vzorcev

se tudi generator zgleduje po obstoječih rešitvah na področju naborov (angl. *itemsets*). Podroben opis algoritma podajamo v razdelku 3.4.1, saj tudi svoj algoritem za dopolnjevanje majhnih označenih omrežij preizkušamo na sintetičnih podatkih, generiranih s tem postopkom.

2.4 Napovedovanje globalnega časovnega razvoja

Pri predpostavki, da graf s časom raste, lahko za napovedovanje njegovega časovnega razvoja na podlagi globalnih lastnosti z generatorjem generiramo večji graf. S tem sicer ne bomo dobili zanesljivih konkretnih informacij o posameznih novih povezavah in vozliščih, v skladu z generatorjevimi predpostavkami glede parametrov pa se bodo razvijali globalni parametri grafa.

Z generiranim grafom tako lahko preverimo naše predpostavke o parametrih in razvoju grafa, če imamo o njem na voljo dejanske podatke v dveh različnih časih. To pomeni, da izvemo nekaj o procesu, ki je generiral realen graf. Analiza grafa soavtorstev in citiranj znanstvenih objav, na primer, nam da podatke o vedênju raziskovalcev v zvezi s citiranjem in medsebojnim sodelovanjem [41].

Z generiranim večjim grafom lahko tudi opazujemo spremembe tistih lastnosti grafa, ki niso neposredno določene s predpostavkami, uporabljenimi v generatorju. Prav tako lahko testiramo delovanje algoritmov na predvidenem stanju grafa v prihodnosti [33].

Mnoge globalne lastnosti rastočih omrežij je možno napovedati tudi bolj neposredno s pomočjo odkritih zakonov njihovega časovnega razvoja [8].

2.5 Napovedovanje lokalnega časovnega razvoja in odpravljanje napak

Vse pogosteje je cilj analize omrežij pridobiti konkretne (lokalne) napovedi sprememb: spremembe povezav, vozlišč ali njihovih oznak. S tem dosežemo simulacijo nadaljnjega razvoja omrežja, odpravljamo napake in pomanjkljivosti v omrežju, napovedujemo verjetnosti določenih akcij posameznih akterjev, na podlagi teh predlagamo akcije (npr. dodajanje prijatelja v socialnem omrežju) itd. Za napovedi lahko uporabljamo posebej v ta namen razvite postopke, kot sta *razširjanje oznak* (angl. *label propagation* [42]) in metoda *naključnega sprehoda* (angl. *random walk*), ali pa neposredno uporabljamo statistična orodja in metode strojnega učenja v kombinaciji z za omrežja specifičnimi atributi. Med slednje seveda spadajo tudi odkriti lokalni vzorci.

Najbolj znan problem s širšega področja je *uvrščanje vozlišč*, na katerega lahko gledamo ko na spreminjanje oznak vozlišč. Pri družbenih mrežah lahko na primer ločimo tiste uporabnike, ki bodo reagirali na določen oglas, in tiste, ki ne bodo. Najsodobnejši pristop k temu problemu je prav razširjanje oznak, ki je predvsem na redkih grafih časovno in prostorska učinkovita tehnika, saj asimptotično ne zahteva več prostora, kot je potrebnega za shranjevanje omrežja, časovna zahtevnost pa je skoraj linearna v številu povezav [43, 44].

Vedno več znanstvenih objav se pojavlja tudi na področju reševanja različnih problemov ocenjevanja verjetnosti prisotnosti določenih dodatnih gradnikov (povezav in včasih vozlišč) v omrežju. Pomembno gonilo raziskav so potrebe industrije, predvsem spletnih ponudnikov storitev mreženja. Za lažje razumevanje opisa različnih vrst problemov s tega področja recimo, da imamo na voljo graf H, za katerega predpostavljamo, da je podgraf nam neznanega grafa H'. Naloga je iz H in morebitnega modela, ki ga zgradimo na tem grafu ali drugih grafih iz iste domene, sklepati o H'.

2.5.1 *Problem napovedovanja povezav*

Pri reševanju *problema napovedovanja povezav* (angl. *link prediction problem* [45]) predpostavimo, da so nam znana vsa vozlišča ($V_{H'} = V_H$). Na vozliščih in povezavah ni oznak ali uteži. Cilj je predvideti manjkajoče povezave $(u, v) \in V_H \times V_H$.

Najosnovnejše metode, ki rešujejo ta problem, temeljijo na podobnosti med vozlišči: bolj *podobnim* vozliščem napovedo višjo verjetnost povezovanja, pri čemer je podobnost na posamezni metodi lasten način izračunana iz topologije omrežja [46].

- Metoda *skupnih sosedov* (angl. *common neighbours*) privzame, da je podobnost dveh vozlišč (in s tem verjetnost njunega povezovanja) sorazmerna številu njunih skupnih sosedov: $s_{uv}^{\mathrm{CN}} = |\Gamma(u) \cap \Gamma(v)|$, kjer je $\Gamma(x)$ množica sosedov vozlišča x. Razni indeksi utežijo to mero tako, da na primer izničijo prednost vozlišč v gostejšem delu omrežja.

- *Saltonov* indeks s_{uv}^{CN} deli s $\sqrt{k_u \cdot k_v}$, kjer je $k_x = |\Gamma(x)|$, *Jaccardov* indeks z $|\Gamma(u) \cup \Gamma(v)|$, *Sørensenov* z $\frac{1}{2}(k_u + k_v)$, *Leicht-Holme-Newmanov* pa s $k_u \cdot k_v$.

- Indeksa, ki nagradita oziroma kaznujeta krajišča z izredno visoko stopnjo (angl. *hubs*) sta *Hub Promoted Index* in *Hub Depressed Index*. Prvi deli s_{uv}^{CN} z $\min\{k_u, k_v\}$, drugi pa z $\max\{k_u, k_v\}$. Slovenskih prevodov za ta dva izraza v

literaturi nismo zasledili. Sami jih nismo kovali, ker pojma zgolj bežno omenjamo v pregledu področja.

- *Princip prednostne povezanosti* (angl. *Preferrential Attachment Rule*) podobnost vozlišč izrazi kot $s_{uv} = k_u \cdot k_v$.
- *Adamic-Adarjev indeks* je v osnovi enak štetju skupnih sosedov, a le-te uteži tako, da imajo manj povezani sosedje višjo težo: $s_{uv} = \sum_{w \in \Gamma(u) \cap \Gamma(v)} (\log k_w)^{-1}$.
- *Indeks dodeljevanja virov* (angl. *Resource Allocation Index*) je utemeljen s teorijo dinamike dodeljevanja virov v kompleksnem omrežju [47], podobnost med vozlišči pa računa podobno kot Adamic-Adarjev indeks: $s_{uv} = \sum_{w \in \Gamma(u) \cap \Gamma(v)} k_w^{-1}$.

Med najboljšimi metodami napovedovanja povezav je *naključni sprehod s ponovnimi začetki* (angl. *Random Walk with Restarts, RWR* [48]). Ta je definirana z naključnim sprehodom iz točke u, ki se v vsakem koraku z verjetnostjo α vrne v izhodišče, v nasprotnem primeru pa se enakomerno naključno premakne v enega od sosedov trenutnega vozlišča. Metoda verjetnost pojavitve povezave med vozliščema u in v izračuna kot verjetnost vozlišča v v stacionarni porazdelitvi naključnega sprehoda. Ta je enaka pričakovanemu deležu časa, ki ga bo ta naključni sprehod prebil v vozlišču v. V praktični implementaciji metode naključnega sprehoda ne simuliramo, temveč uporabimo relevantno statistično teorijo Markovskih verig za neposredni izračun vrednosti z metodami linearne algebre.

V družbenih omrežjih so včasih na vozliščih prisotni dodatni atributi. Te zna upoštevati *nadzorovan naključni sprehod* (angl. *Supervised Random Walk* [49]). Pri tej metodi je naključni sprehod utežen s funkcijo, ki je priučena za vsako posamezno izvorno vozlišče. Tako znanje ni prenosljivo iz enega omrežja na drugega.

2.5.2 *Problem dopolnjevanja omrežja*

Drug popularen problem predikcije v omrežjih je *problem dopolnjevanja omrežja* (angl. *network completion problem*), ki predpostavlja, da je H induciran podgraf grafa H', torej poznamo vse povezave med vidnimi vozlišči, so nam pa prikrita nekatera vozlišča. Cilj je torej napovedati vozlišča $V_{H'} \setminus V_H$ in povezave, ki imajo vsaj eno krajišče v tej množici.

Primer metode, ki rešuje tak problem, je KronEM [50]. Z uporabo metode *maksimizacije pričakovanj* (angl. *expectation maximization*) najde optimalen Kroneckerjev

model omrežja, generira večje omrežje in pri tem poskusi najti ujemanja med generiranimi vozlišči in obstoječimi vozlišči v omrežju. Ostala vozlišča in pripadajoče povezave napove kot nove.

2.5.3 *Napovedovanje časovnega razvoja*

Če imamo na vozliščih in/ali povezavah kot dodatno informacijo na voljo časovne žige njihovega nastanka, si lahko zastavimo problem, ki je soroden prej opisanima, a privzame drugačno strukturo podatkov. Po učenju na časovnem zaporedju omrežij, kjer predpostavljamo, da se povezave in vozlišča skozi čas zgolj dodajajo (to je sorodno opazovanju enega omrežja s časovnimi žigi na povezavah) nas zanima verjetnost pojavitve povezave med določenim parom nepovezanih vozlišč v določeni časovni točki.

Problem napovedovanja razvoja od časa odvisnih omrežij je obravnaval Bringmann s sodelavci [51] in pri tem ni predpostavil niti da poznamo vsa vozlišča niti da poznamo vse povezave poznanih vozlišč. Kot model od časa odvisnega omrežja so uporabili rezultat algoritma GERM [52], ki smo ga opisali v razdelku 2.2.3.

Leskovec s sodelavci je v predznačenih družbenih omrežjih, tj. takih, ki imajo na povezavah pozitivno ali negativno oznako, napovedoval nove povezave s pomočjo preprostega modela za *zapiranje trikotnikov*. Z uporabo klasičnih metod strojnega učenja je nato napovedal tudi predznak povezave – prijateljstvo ali neodobravanje oziroma sovraštvo [53]. Rezultati so se dobro ujemali z realnimi podatki, model pa s sociološkimi teorijami iz 40. in 50. let prejšnjega stoletja. Pri snovanju metode so se opisali na predhodne raziskave rasti družbenih omrežij, ki so empirično pokazale, da je verjetnost pripenjanja nove povezave na določeno vozlišče v linearni zvezi z njegovo stopnjo, pri čemer pa velika večina novih povezav *zapira trikotnike*, torej poveže vozlišči, ki sta bili prej na razdalji 2 [54].

3

Dopolnjevanje majhnih omrežij

V kontekstu dela z množicami majhnih označenih omrežij iz iste domene, ki jih na vozliščih in povezavah bogatijo diskretne oznake, je bilo predvsem po letu 2000 opravljenih precej raziskav. Pozornost so namenjale analizi takih množic in odkrivanju vzorcev (angl. *patterns*), ki se v posamezni množici pogosto pojavljajo, po vzoru asociativnih pravil.

Bistveno manj se obstoječe raziskave posvečajo dinamični analizi, tj. pomenu in načinu predvidevanja, kako se bi se omrežje iz določene domene lahko spremenilo npr. pod vplivom časa ali v okviru odpravljanja napak in pomanjkljivosti v omrežju. Pričujoče poglavje je eden prvih kamenčkov v mozaiku za zapolnitev te vrzeli.

V nadaljevanju natančno definiramo problem, ki ga želimo rešiti, ter opišemo metodo oziroma družino metod, ki smo jo v ta namen razvili. Sledi opis realnih in sintetičnih podatkov, na katerih smo ocenili točnost družine metod. Ker gre za nerazvito področje, smo v ta namen morali definirati tudi ocenjevalno metriko.

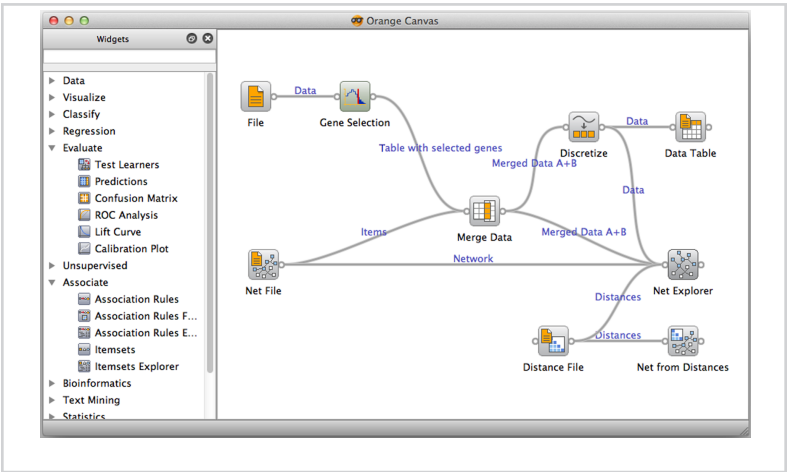
3.1 Problem

Dinamična analiza majhnih označenih omrežij ponuja široko paleto možnosti za raziskovalno obravnavo. Zastavili smo si problem, ki je precej ožji od celotnega področja in tako daje upanje, da je obvladljiv, hkrati pa je daleč od trivialnega ali neuporabnega. Slednjo trditev bomo v tem razdelku podkrepili s praktičnimi primeri.

Problem, ki smo si ga zastavili, imenujemo *dopolnjevanje majhnih omrežij*, kjer pod pojmom *omrežje* razumemo *graf z oznakami*, *majhen* graf pa je tak z do nekaj deset vozlišči. Na podlagi analize statističnega vzorca (angl. *sample*) majhnih omrežij želimo pri novem omrežju iz iste domene napovedati povezave in vozlišča, ki so bili pri zapisu omrežja pomotoma opuščeni ali pa se bodo v omrežju pojavili šele s časom.

Primere naravnih in družbenih sistemov, katerih naraven opis je množica majhnih omrežij, smo že podali pri pregledu področja (poglavje 2). Dopolnjevanje prej nevidenega omrežja iz domene neke take množice je lahko koristno v več kontekstih.

Glavna aplikativna motivacija za naše delo sta bila dva pomensko sorodna problema, s katerima smo se srečevali v praksi. Delo na področju odkrivanja zakonitosti iz podatkov in strojnega učenja je podprto tudi z računalniškimi sistemi z grafičnimi uporabniškimi vmesniki za vizualno programiranje, kot sta Taverna [55] in Orange [56]. Tovrstni vmesniki omogočajo osebam, ki niso vešče uporabe klasičnih programskih jezikov, izvesti analizo podatkov z risanjem podatkovnih tokov. Slednji predstavljajo omrežja medsebojno povezanih funkcijskih enot za nalaganje, obdelavo, prikaz in



Slika 3.1: Orange Canvas – grafični uporabniški vmesnik programa Orange za odkrivanje zakonitosti v podatkih in strojno učenje. Uporabnik v njem definira shemo za procesiranje podatkov z gradnjo omrežja – podatkovnega toka. Vozlišča predstavljajo funkcijske enote, ki komunicirajo preko podatkovnih kanalov (povezav).

shranjevanje podatkov. Primer je prikazan na sliki 3.1.

Dopolnjevanje tovrstnih omrežij z metodami strojnega učenja bi lahko omogočilo samodejno (algoritmično) pomoč uporabniku pri odpravljanju pomanjkljivosti oziroma nadaljnji gradnji podatkovnega toka. Učne podatke bi v tem primeru predstavljali podatki o pretekli uporabi dotičnega sistema za vizualno programiranje s strani tega uporabnika ali večje skupnosti uporabnikov.

Vsebinsko nekoliko drugačen primer realnih podatkov v obliki množice majhnih omrežij so še sheme elektronskih vezij. Tudi tu si je možno zamisliti sistem, ki bi z uporabo strojnega učenja asistiral inženirju pri risanju načrta za tiskano vezje. Ni sicer takoj jasno, kako vezje opisati v obliki omrežja na način, ki bo koristen v smislu reševanja problema dopolnjevanja majhnih omrežij. Ker smo delovanje družine metod,

ki smo jo zasnovali, ocenili tudi na tovrstnih podatkih, bomo več besed predpripravi in zbiranju posvetili pri opisu podatkov v razdelku 3.4.1.

3.1.1 Umestitev problema v kontekst sorodnih problemov

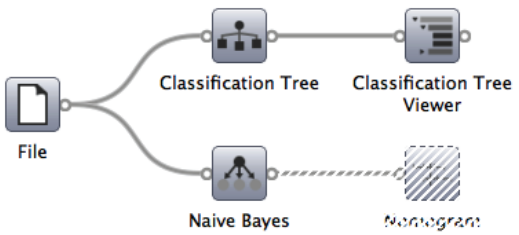
Napovedovanje povezav v velikih omrežjih oziroma dopolnjevanje velikih omrežij sta relativno dobro raziskana problema tudi po zaslugi komercialnih potreb podjetij, ki ponujajo storitve internetnih socialnih omrežij. Dopolnjevanje majhnih omrežij se od omenjenega problema bistveno razlikuje. Na to kažejo tudi slabi rezultati, ki jih metode za napovedovanje povezav v velikih omrežjih izkazujejo v naših poskusih.

Algoritmi, razviti za velika omrežja, se pri napovedovanju zanašajo na topološke informacije in, ko so znani dodatni podatki o vozliščih, na podobnost med vozlišči [46]. Pogosto je uporabljen princip *homofilije* [57], ki pravi, da se vozlišča rada povezujejo s sebi podobnimi. Majhna omrežja sicer pogosto vsebujejo dodatne podatke o vozliščih, vendar v obliki oznak, ki jih je brez predhodne ureditve v hierarhijo ali informacije o podobnosti med oznakami težko uporabiti za izračun podobnosti med vozlišči. Prav tako v podatkih, ki smo jih zbrali, ne vidimo očitne razlage, zakaj bi bila homofilija pomemben dejavnik pri vzpostavljanju povezav. Po drugi strani pa je zelo očitno, da nam oznake pomagajo ločiti med različnimi vrstami vozlišč in definirati vzorce, po katerih se ti pogosto medsebojno povezujejo.

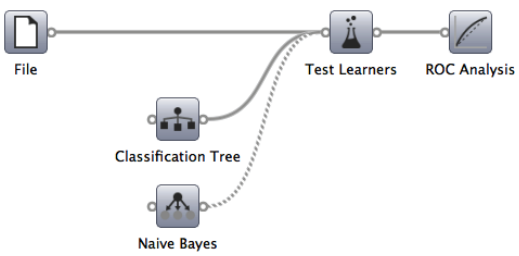
Velika omrežja pogosto predstavljajo socialne mreže, v katerih je bil opažen princip *zapiranja trikotnikov*: če sta dve vozlišči povezani s tretjim vozliščem, je precej verjetno, da sta tudi medsebojno povezani (ali pa se bosta povezali v prihodnosti). Metode za napovedovanje povezav se pogosto zanašajo na to opazko [54]. V velikih omrežjih je to razumljivo, majhna pa popisujejo drugačne pojave, pri katerih ta princip ne velja ali ni prevladujoč.

Še ena pomembna razlika med analizo majhnih in velikih omrežij je, da se pri slednjih pogosto učimo na danem omrežju in zatem na njem tudi napovedujemo prihodnje dogodke. Iz enega majhnega omrežja ne moremo izluščiti veliko informacij, zato jih za strojno učenje potrebujemo več. To nam tudi dovoljuje uporabo takšnih metod, ki na velikem omrežju ne bi delovale zaradi časovne in/ali prostorske potratnosti, na posameznih majhnih omrežjih pa s tem nimamo težav.

Razlika v velikosti omrežij ima posledice tudi za vzorce (podomrežja), ki jih lahko uporabimo v analizi. Če za velika omrežja velja, da vzorci ne smejo imeti več kot 3–5 vozlišč [58], sicer bi se zaradi *pojava majhnega sveta* raztezali preko celotnega omrežja,



(a) Majhno omrežje z napovedjo novega vozlišča in povezavo.



(b) Majhno vozlišče z napovedjo nove povezave.

Slika 3.2: Ilustraciji problema dopolnjevanja majhnih omrežij.

pri majhnih omrežjih to ni težava. Vzorec lahko vsebuje več vozlišč in morda v katerem od primerov zavzema kar celotno omrežje, a je še vedno *lokalen* v smislu, da je prisoten le v nekaterih omrežjih v množici, v drugih pa ne.

3.1.2 Ilustrativen primer naloge

Za lažjo predstavo praktične aplikacije zastavljenega problema sta na sliki 3.2 prikazani konkretni nalogi na primeru sistema za vizualno programiranje, ki je del paketa za odkrivanje zakonitosti v podatkih Orange.

V prvem delu slike lahko vidimo, da je uporabnik naložil podatke iz datoteke (gradnik *File*), zgradil klasifikacijsko drevo (gradnik *Classification Tree*) in si zgrajen model ogledal s prikazovalnikom dreves (*Classification Tree Viewer*). Z gradnikom *Naive Bayes* je zgradil še en model z naivnim Bayesovim klasifikatorjem. Upravičeno lahko pričakujemo, da si bo tudi ta model želel ogledati. Primerna vizualizacija naivnega Bayesovega

klasifikatorja je nomogram, zato bi bilo smiselno predlagati gradnik *Nomogram*.

Na primer lahko gledamo z vsaj dveh nivojev splošnosti. En je konkreten in s tem uporaben za dopolnjevanje omrežij, ki vsebujejo točno navedene štiri gradnike (poleg morebitnih drugih): *če shema vsebuje gradnik za nalaganje datotek, ki je povezan s klasifikacijskim drevesom, to pa je povezano s pregledovalnikom dreves, poleg tega pa je na nalaganje datotek povezan tudi naivni Bayesov model, potem je pričakovano, da bo na sledjega priključen še gradnik za prikaz nomograma.* Drugi pogled je bolj splošen: *če je vir podatkov priključen na algoritem za gradnjo modela, ta pa na pripadajoči pregledovalnik, poleg tega pa isti vir podatkov črpa še en model, se pričakuje, da bo tudi ta priključen na svoj pregledovalnik.*

V opisani ilustraciji s pomočjo vzorca, ki je bil v praksi odkrit kot pogost, napovemo novo vozlišče in pripadajočo povezavo. Primer na sliki 3.2b prikazuje napoved povezave med obstoječimi vozlišči. Če je uporabnik sestavil shemo, ki vsebuje gradnika za gradnjo klasifikacijskih dreves in naivnega Bayesovega modela in ki ocenjuje napovedno točnost prvega modela na določenih podatkih, potem bo verjetno želel na istih podatkih preizkusiti tudi drugi model.

Dani primeri nazorno kažejo, da se dopolnjevanje majhnih omrežij bistveno razlikuje od napovedovanja vozlišč in povezav v velikih omrežjih. Pri napovedih uporabljamo vzorce, ki pogosto zavzemajo velik del omrežja, napovedi povezav pa niso osnovane na principu zapiranja trikotnikov prijateljstva ali povezovanju sorodnih (podobnih) vozlišč, temveč upoštevajo oznake oziroma kategorije vozlišč in/ali povezav glede na odkrite pogoste vzorce.

Zaradi različnosti problemov dopolnjevanja velikih in majhnih omrežij niso prenosljivi niti postopki in metrike za ocenjevanje kvalitete metod. Pri velikih omrežjih lahko zakrijemo del omrežja, nato pa za oceno uspešnosti metode pri rekonstrukciji omrežja uporabimo metrike za ocenjevanje dvojiškega uvrščanja, kot sta ploščina pod ROC krivuljo (angl. *Area Under Curve, AUC*) in preciznost (angl. *precision* [46]). Pri majhnih omrežjih prikrivanje več kot ene povezave pogosto korenito spremeni strukturo omrežja. Po drugi strani pa pri prikrivanju ene same povezave AUC in preciznost nista več uporabni meri za ocenjevanje kvalitete napovedi.

3.1.3 Koncepti in notacija

V nadaljevanju želimo formalno definirati problem. V tem razdelku bomo najprej spoznali koncepte in določili notacijo za strukture, ki jih potrebujemo pri definiciji

problema.

Omrežje, ki ga sestavljata graf in dodatne informacije v obliki oznak na vozliščih in povezavah, imenujemo tudi *označen graf*. Zapišemo ga kot trojko $G = (V_G, E_G, \ell_G)$, v kateri sta prvi dve komponenti množici vozlišč in (usmerjenih ali neusmerjenih) povezav tega grafa, $\ell_G : V_G \cup E_G \to L$ pa je *označevalna funkcija*, ki vozliščem in povezavam določa oznake iz neke množice L.

Recimo, da za domeno majhnih označenih grafov, znotraj katere želimo dopolnjevati omrežja, poznamo vzorec reprezentativnih grafov – množico $\mathscr{G}$.

Označimo s $P = (V_P, E_P, \ell_P)$ vzorec (angl. *pattern*), tj. (šibko) povezan označen graf. Vzorec je *podgrafno izomorfen* (angl. *subgraph isomorphic* [59]) grafu G, če je izomorfen nekemu podgrafu grafa G. V kontekstu označenih grafov seveda velja, da se morajo pri izomorfizmu ujemati tudi oznake.

Funkcija vpetja je injektivna preslikava vozlišč vzorca P v vozlišča grafa G, ki zadošča pogoju, da je vzorec P podgrafno izomorfen grafu, vpetemu na točkah, v katere slika ϵ. Naj poudarimo, da še vedno govorimo o podgrafnem izomorfizmu in ne o pravem izomorfizmu, saj še vedno dovolimo, da v G med vozlišči, v katere slika ϵ, obstajajo še dodatne povezave. Zopet seveda zahtevamo ujemanje oznak na vozliščih in povezavah.

Enačbi 3.1 in 3.2 strnita lastnosti, ki jim mora zadoščati funkcija vpetja v kontekstu označenih grafov.

$$\forall v \in V_P \quad : \quad \ell_P(v) = \ell_G(\epsilon(v)) \tag{3.1}$$

$$\forall e_P = (u, v) \in E_P \quad : \quad e_G = (\epsilon(u), \epsilon(v)) \in E_G \ \wedge \ \ell_P(e_P) = \ell_G(e_G) \tag{3.2}$$

Naj bo rezultat funkcije $\phi(P, G)$ množica (lahko tudi prazna) vseh veljavnih vpetij vzorca P v grafu G.

Vsakemu vzorcu, ki ga odkrijemo v množici majhnih grafov, lahko določimo *podporo* (angl. *support*), tj. številsko količino, ki za vsak vzorec pove, kako pomemben, zanesljiv, oziroma pogost je. Podporo vzorca P v množici grafov $\mathscr{G}$ označimo $\sigma(P, \mathscr{G})$. Najpreprostejša in najpogosteje uporabljena mera podpore na področju odkrivanja pogostih vzorcev v grafih je frekvenca, tj. število opazovanih grafov, ki vsebujejo dani vzorec:

$$\sigma(P, \mathscr{G}) = \left| \left\{ G \mid G \in \mathscr{G},\ \phi(P, G) \neq \varnothing \right\} \right|. \tag{3.3}$$

P je *pogost vzorec* (angl. *frequent pattern*) v $\mathscr{G}$, če je njegova podpora zadosti visoka. Konkretno, za vnaprej izbrano minimalno podporo μ mora biti $\sigma(P, \mathscr{G}) > \mu$.

3.1.4 Formalna definicija problema

Recimo, da je $H' = (V_{H'}, E_{H'}, \ell_{H'})$ označen graf, ki ga ne moremo opazovati, lahko pa opazujemo njegov podgraf $H = (V_H, E_H, \ell_H)$. H torej ne vsebuje nekaterih vozlišč in povezav grafa H'; lahko rečemo, da je nepopoln. Vzrok za to je bodisi naše nenatančno poznavanje pravega grafa H' oziroma napake pri zajemu podatkov, bodisi H' predstavlja graf v prihodnosti in se nekateri elementi še niso pojavili. S predvidevanjem manjkajočih povezav v resnici lahko tudi vplivamo na njihovo dejansko pojavitev, npr. v kontekstu spletnih socialnih omrežij, kjer uporabnik lahko preprosto sprejme predlog za prijateljstvo. Za definicijo problema vzrok za odsotnost elementov ni pomemben. Povezave $E_{H'} \setminus E_H$ imenujemo *manjkajoče povezave*, vozlišča $V_{H'} \setminus V_H$ pa *manjkajoča vozlišča*.

Predpostavimo, da množica $\mathscr{G}$ vsebuje majhne označene grafe. Ti so reprezentativni za domeno, s katero se ukvarjamo. Naj bo H na povezave induciran (angl. *edge-induced*) podgraf latentnega grafa H'. Povedano drugače: H ne vsebuje nekaterih povezav iz H', vozlišče pa je brisano le, če ostane brez povezav.

Problem dopolnjevanja majhnih omrežij pomeni napovedovanje manjkajočih povezav in vozlišč v grafu H. Ker je točno napovedovanje prezahtevna naloga, pri reševanju tega problema napovemo seznam potencialnih manjkajočih povezav in vozlišč, urejen po ocenah njihovih verjetnosti.

3.2 Orodja

Razvili smo metodo *Hyspan*, ki rešuje problem dopolnjevanja majhnih omrežij. Pri tem smo se opirali na obstoječe raziskave in že razvite metode s področja analize omrežij. Pred opisom naše metode je zato koristno nekaj besed nameniti orodjem, ki jih uporabljamo za njeno delovanje.

Hyspan za svoje delovanje uporablja pogoste vzorce v omrežjih. Odkrivanje takih vzorcev iz podatkov zahteva identifikacijo podgrafnih izomorfizmov ob upoštevanju oznak na vozliščih in/ali povezavah. To je NP-težek problem [60]. Do neke mere je reševanje problema vseeno možno, predvsem po zaslugi pristopov za oženje iskalnega prostora.

Že samo število možnih vzorcev, ki se pojavljajo v grafih, narašča eksponentno glede na velikost vhodnih grafov. To eksplozijo običajno omejimo tako, da določimo najnižjo podporo, ki jo mora vzorec imeti, da ga štejemo za *pogost* vzorec. Omenili smo že, da

je najosnovnejša mera za podporo vzorca kar število grafov, v katerih se pojavlja. Ta mera ima zelo koristno lastnost – je namreč *monotono padajoča* (angl. *anti-monotone*) v smislu da ima vzorec R, ki vsebuje vzorec P, lahko kvečjemu nižjo podporo: $\sigma(R, \mathcal{G}) \leqslant \sigma(P, \mathcal{G})$. Pomembna posledica te lastnosti je, da lahko vzorce v množici odkrivamo od manjših proti večjim (torej večje vzorce najdemo tako, da z novimi povezavami razširjamo manjše) in neko vejo preiskovanja prekinemo takoj, ko vzorec nima zadostne podpore. Tudi vsi iz njega izpeljani večji vzorci je namreč ne bodo imeli.

Frekvenca ni edina monotono padajoča mera. Nasploh je primerna zgolj za odkrivanje pogostih vzorcev v množici grafov, ne pa npr. v enem velikem grafu. Za reševanje našega problema to zadošča, vseeno pa naj omenimo, da obstajajo tudi monotono padajoče mere, ki delujejo v kontekstu analize enega grafa. Med drugim so to moč *največje neodvisne množice* (angl. *maximum independent set, MIS* [61]) in podpora, ki upošteva *škodljivo prekrivanje* (angl. *harmful overlap* [62]).

3.2.1 gSpan – metoda za odkrivanje pogostih vzorcev

Prva metoda, ki uporablja ta princip za odkrivanje pogostih vzorcev v grafih, je gSpan [26] (*Graph-Based Substructure Pattern Mining*). Vzorce gradi povezavo za povezavo in se s tem izogne potrebi po zahtevnem združevanju dveh večjih vzorcev, na katerem so temeljile zgodnejše metode, ki so se zgledovale po algoritmu Apriori [24] za odkrivanje pogostih vzorcev v naborih (angl. *itemsets*).

Algoritem gSpan za dano množico označenih grafov $\mathcal{G}$ zgradi drevo pogostih vzorcev T. Vse najdene vzorce v celotni množici grafov popisuje eno samo drevo. Posamezen vzorec je zapisan s kanoničnim kodom, imenovanim *najmanjša koda na osnovi iskanja v globino* (angl. *minimum DFS encoding*). Kod zapiše vzorec s seznamom povezav, kot jih našteje *postopek iskanja v globino*, pri čemer je točen vrstni red povezav določen z zahtevo, da pri več enakovrednih kodah za isti vzorec izberemo tistega, ki je *leksikografsko najmanjši*. Vsaka povezava je zapisana s peterko, ki vsebuje dve celi števili in tri oznake z vozlišč oziroma povezav vzorca, zato je tako peterke kot sezname peterk možno leksikografsko primerjati, če je možno primerjati oznake.

Peterka $(i_1, i_2, l_1, l_e, l_2)$ za zapis posamezne povezave vzorca vsebuje:

- zaporedni številki (i_1 in i_2) krajišč povezave, pri čemer je vrstni red določen z vrstnim redom odkrivanja vozlišč pri izvajanju iskanja v globino; začetno vozlišče nosi zaporedno številko 0,

- pripadajoči oznaki vozlišč (l_1 in l_2) in
- oznako same povezave (l_e).

Pri usmerjenih grafih je postopek enak, le da povezave zapisujemo s šestericami; dodatna komponenta označuje smer povezave (tj. od vozlišča i_1 do i_2 ali obratno [63]).

Označimo s $\xi_P(e)$ terko, ki predstavlja povezavo e v neki kodi vzorca P. Koda celotnega vzorca je seznam $\xi_P = [\xi_P(e)]_{e \in E_P}$. Vrstni red povezav v seznamu določa zahteva po leksikografski minimalnosti kode, opozoriti pa je potrebno, da se s tem vrstnim redom spreminjajo tudi terke za posamezne povezave. Spremenijo se zaporedne številke vozlišč, prav tako pa lahko povezavo odkrijemo „z druge strani", torej z zamenjanimi krajišči. Od tod dalje bomo notaciji $\xi_P(e)$ in ξ_P vedno uporabljali za zapis leksikografsko najmanjše kode, ki je pri uporabljenem kodu tudi edina veljavna.

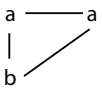
gSpan zgradi drevo takih kod (*drevo kod na osnovi iskanja v globino*, angl. *DFS code tree*) z vsemi pogostimi vzorci v množici grafov. Vsa vozlišča tega drevesa na isti globini predstavljajo vzorce z istim številom povezav. Koda ξ_P je v tem drevesu otrok kode ξ_R natanko tedaj, ko se ξ_P začne s ξ_R in nadaljuje z natanko eno dodatno terko. Tedaj je tudi R podgraf P, kjer je $|E_R| = |E_P| - 1$. Obratno ne velja nujno: če je R podgraf P in slednji vsebuje natanko eno povezavo več, lahko to zaradi zahteve po minimalni kodi spremeni vrstni red naštevanja povezav.

Primer drevesa kod na osnovi iskanja v globino je prikazan na sliki 3.3.

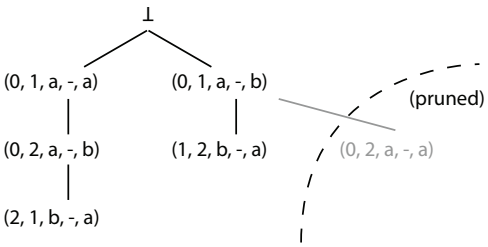
gSpan drevo gradi po nivojih. Začne z izčrpnim naštevanjem oziroma preštevanjem vzorcev z eno samo povezavo. Ko zaključi gradnjo enega nivoja drevesa, izloči vse vzorce, katerih podpora ne zadošča nastavljeni najnižji podpori. Ker je podpora monotono padajoča, ni možnosti, da bi bil s tem zavržen pogost vzorec.

Otroke vsakega vozlišča v drevesu algoritem generira tako, da v vseh grafih iz množice najde $\mathscr{G}$ vsa vpetja vzorca, ki ga vozlišče predstavlja, in jih nato razširja s po eno povezavo v okolici.

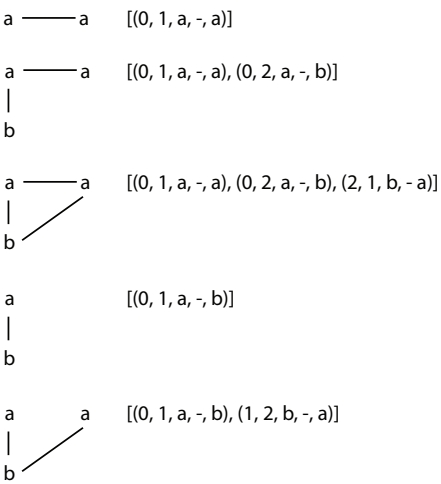
Ko algoritem zgradi kodo, ki ni leksikografsko najmanjša, to pomeni, da je minimalen opis tega vzorca že odkril. Tega ni potrebno ugotavljati s časovno zahtevno primerjavo vseh parov odkritih kod, temveč je razvidno že iz dejstva, da je za opisani vzorec moč najti manjšo kodo. Pogosto niti ni potrebno izčrpno naštevati vseh kod za določen vzorec, saj mnogih neveljavnih kod algoritem zaradi vgrajenih hevristik sploh ne generira.



(a) Vhodni graf, katerega vzorce popisuje drevo



(b) Drevo kod



(c) Vzorci v grafu, ki jih popisujejo vozlišča drevesa

Slika 3.3: Primer drevesa kod na osnovi iskanja v globino. V vozliščih so zaradi preglednosti zapisane le terke, ki opisujejo zadnjo odkrito povezavo. Celotna koda vzorca v posameznem vozlišču je seznam, ki vsebuje po vrsti vse terke od korena drevesa do tega vozlišča. Prvo in zadnje vozlišče na drugem nivoju drevesa predstavljata isti vzorec: vozlišče z oznako *a*, ki je povezano s še enim vozliščem z oznako *b* in enim z oznako *b*. Zadnje vozlišče ne vsebuje leksikografsko najmanjše kode, ki jo gSpan zavrže še preden jo poskuša razširiti v večji vzorec.

Ob koncu izvajanja algoritma drevo vsebuje vse vzorce z zadostno podporo v $\mathscr{G}$. Rezultat algoritma je *seznam najdenih vzorcev* (v obliki seznama označenih grafov) skupaj s pripadajočimi vrednostmi podpore.

Edini parameter algoritma gSpan je najmanjša podpora (μ) najdenih vzorcev. Če je mera podpore frekvenca, torej število grafov iz $\mathscr{G}$, ki vzorec vsebujejo, lahko minimalno podporo izrazimo absolutno (kot celo število) ali relativno (kot delež števila vseh grafov v $\mathscr{G}$).

3.3 Hyspan – metoda za dopolnjevanje majhnih označenih omrežij

Metoda Hyspan (*Hypotheses from Substructure Patterns*, slov. *hipoteze na podlagi podstrukturnih vzorcev*), ki smo jo razvili [64], je po zunanji zgradbi tipičen postopek strojnega učenja, saj ima dva dela. Prvi del je *učenje*, pri katerem algoritem analizira učne podatke in zgradi *model*, ki predstavlja zgoščeno znanje oziroma hipoteze o mehanizmih, ki so pripeljali do videnih učnih podatkov. Drugi del je *napovedovanje*. Vhodna podatka za ta del sta prej zgrajeni model ter nov, še neviden označen graf, domnevno iz iste domene kot so bili učni podatki. Izhod tega dela metode so hipoteze o manjkajočih povezavah in vozliščih v novem grafu.

Glavna ideja metode je, da je možno vzorce, ki se v domeni pogosto pojavljajo, uporabiti za dopolnjevanje novih grafov. Če opazimo, da v grafu manjka le nek del vzorca, lahko postavimo hipotezo, da je graf tam potrebno dopolniti. Manjkajoče dele lahko časovno učinkovito odkrijemo tako, da poskušamo vpeti vzorce, pri čemer dopustimo določeno število odstopanj v smislu manjkajočih povezav in vozlišč.

Pri natančnejšem opisu postopka si bomo pomagali z novodefiniranim konceptom: vpetje vzorca P v razširjen graf $\widetilde{H} = (V_H \cup \widetilde{V}, E_H \cup \tilde{E}, \ell'_H)$ bomo poimenovali *delno vpetje* vzorca P v graf H je. Dodatna vozlišča $\widetilde{V}$ in povezave $\tilde{E}$ imenujemo *navidezna vozlišča in povezave* (angl. *virtual vertices and edges*). Označevalna funkcija ℓ'_H je ustrezna razširitev ℓ_H. Število v delnem vpetju uporabljenih navideznih povezav je omejeno s konstanto ν.

3.3.1 Učenje

Model metode Hyspan predstavljajo vzorci (označeni podgrafi), ki so pogosti v učni množici in vrednosti njihove podpore (tj. njihove frekvence). Slednje metoda uporabi

za smiselnejše rangiranje hipotez. Vzorci morajo biti v taki podatkovni strukturi, ki omogoča učinkovito iskanje delnih vpetij.

Seznam vzorcev z informacijami o podpori lahko zgradimo z uporabo algoritma gSpan. Še več: *drevo kod na osnovi iskanja v globino*, ki ga ta gradi med svojim delovanjem, je struktura, s pomočjo katere lahko učinkovito iščemo delna vpetja vzorcev. gSpan je torej potrebno prilagoditi, da kot rezultat ne vrača seznama, temveč kar drevo. Tako prilagojen algoritem nam služi kot učni del metode Hyspan.

Tako kot gSpan ima tudi Hyspan parameter μ – najmanjšo vrednost podpore, ki jo mora vzorec dosegati, da ga štejemo za pogostega. Kasneje bomo pokazali, da pri uporabi vzorcev za dopolnjevanje omrežij nižje vrednosti μ praviloma dajejo boljše rezultate za ceno daljšega časa izvajanja. To sicer pomeni, da vrednosti parametra ne moremo izbirati z uporabo notranjega prečnega preverjanja (angl. *internal cross validation, ICV*), a lahko uporabimo še enostavnejši pristop. Ena možnost je, da začnemo z nizkimi vrednostmi in postopek prekinemo, če se ne izvede v vnaprej določenem še sprejemljivem času, ter poskusimo z nekoliko višjo vrednostjo. V naših poskusih smo uporabili obraten pristop – začeli smo z visoko vrednostjo parametra (npr. $\mu = \frac{1}{2} \cdot |\mathcal{G}|$) in jo nato po občutku zniževali, dokler se je ocena kvalitete napovedi še bistveno spreminjala oziroma dokler ni čas izvajanja postal nesprejemljivo visok.

Predvidevamo, da bi bilo v popolnoma avtonomnih sistemih, ki običajno delujejo znotraj ene domene, možno izbrati fiksno vrednost, ki daje dobre rezultate. Če to ne bi bilo mogoče, bi opisani postopek iskanja primerne vrednosti lahko avtomatizirali.

3.3.2 Napovedovanje – postavljanje hipotez

Napovedni del metode kot vhod zahteva nov graf H iz iste domene (torej predvidoma sorodne strukture), kot so bili grafi v množici na vhodu učnega dela algoritma, in drevo kod, ki predstavlja vzorce, ter sestavi seznam morebitnih manjkajočih povezav. Algoritem, ki smo ga zasnovali, najde vsa delna vpetja pogostih vzorcev v danem grafu in s tem odkrije kandidatne povezave. Te nato z izbrano funkcijo točkuje glede na količino in kvaliteto indicev ter jih sortira po padajočem številu točk.

Naj spomnimo, da je vsak odkriti pogosti vzorec predstavljen s kodo v drevesu T, vsaka koda pa je seznam terk, ki ustrezajo povezavam v vzorcu. Naj bo $P[i]$ i-ta povezava v kodi vzorca P. Funkcijo vpetja ϵ_P, ki slika vozlišča vzorca P v vozlišča razširjenega grafa $\widetilde{H}$, opišemo s seznamom povezav tega grafa, pri čemer i-ta povezava ustreza sliki $P[i]$.

Hyspan za vsak pogost vzorec z eno samo povezavo (tj. na prvem nivoju drevesa T) najde vsa možna vpetja v razširjenem grafu $\widetilde{H}$. To je trivialno, saj je potrebno le najti povezave z ustreznimi oznakami na povezavi sami ter na krajiščih. Za vsako odkrito vpetje Hyspan kliče rekurzivno funkcijo, ki:

- se takoj konča, če do zdaj sestavljeno vpetje vsebuje več virtualnih povezav, kot je dovoljeno (ν), sicer
- izračuna in prišteje vpliv tega vpetja na točkovanje virtualnih povezav, ki bodo predlagane kot morebitne manjkajoče povezave,
- najde vsa vpetja za vsak pogost vzorec, ki je v drevesu T otrok vzorca P, tako da razširi vpetje vzorca P z dodatno povezavo v $\widetilde{H}$; ob vsakem odkritju vpetja se funkcija rekurzivno pokliče.

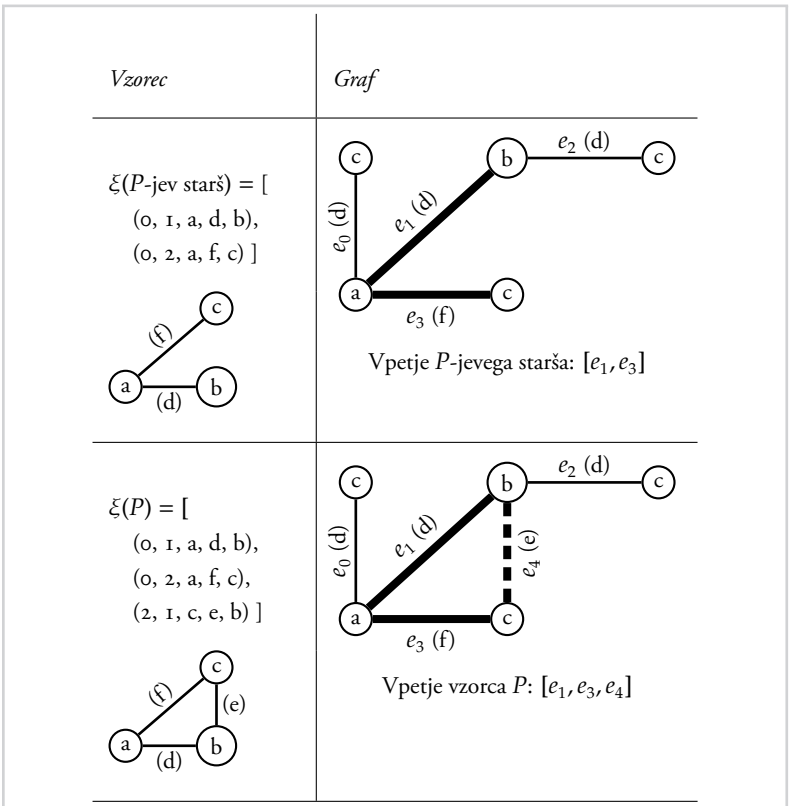
Na ta način je vsako delno vpetje v grafu H vsakega vzorca iz drevesa T obiskano natanko enkrat, če je *veljavno* v smislu, da ne vsebuje več navideznih povezav, kot je dovoljeno. Neveljavna delna vpetja so zavržena in se jih ne razširja v delna vpetja še večjih vzorcev, saj bi število uporabljenih navideznih povezav s tem kvečjemu povečali.

Odkrito vpetje vpliva na točkovanje manjkajočih povezav le, če je *zasidrano* v vidnem delu grafa H, tj. če vsebuje vsaj eno nenavidezno vozlišče. Tako vpetje prinese točke vsem navideznim povezavam, ki jih vsebuje, zato so v končnem seznamu povezav uvrščene višje. Pri izbiri načina točkovanja imamo široko paleto možnosti; mi smo si zamislili tri različne funkcije in jih uporabili v naših poskusih. Opisane so v razdelku 3.3.3.

Nezasidrana delna vpetja nimajo neposrednega vpliva na točkovanj povezav, kljub temu pa jih algoritem rekurzivno razširi v vpetja večjih vzorcev. Z razširitvijo nezasidranega vpetja lahko namreč dobimo zasidrano vpetje, ki ga moramo upoštevati.

Pred začetkom postopka v graf $\widetilde{H}$ dodamo ν navideznih vozlišč za vsako možno oznako. Navidezna vozlišča, ki niso uporabljena v delnih vpetjih, niso predlagana kot manjkajoča, saj niso vpletena kot krajišča v končnem seznamu povezav. Vseh možnih navideznih povezav je v splošnem mnogo več kot možnih navideznih vozlišč, zato se te dodajajo sproti. Rekurzivni korak iskanja delnega vpetja, ki privede do dodajanja navidezne povezave, je prikazan na sliki 3.4.

Vse navidezne povezave z več kot 0 točkami so vključene v končni seznam hipotez.



Slika 3.4: Prikaz enega od možnih vpetij vzorca P (spodaj), do katerega pridemo z razširitvijo vpetja P-jevega starša (zgoraj) v drevesu kod. Vpetje vzorca P vsebuje tudi navidezno povezavo e_4, ki je bila v graf dodana v 14. vrstici algoritma. Obe vpetji sta izraženi v obliki seznama povezav grafa, ki ustrezajo terkam v kodi vzorca.

Če želimo napovedati en korak v razvoju omrežja, je primerna vrednost parametra $\nu = 1$. To omeji tudi število navideznih vozlišč na 1. Če predpostavimo, da poznamo vsa vozlišča omrežja, lahko izpustimo korak dodajanja navideznih vozlišč.

Celoten algoritem za napovedni del metode Hyspan je opisan na sliki 3.5.

3.3.3 *Točkovanje hipotez o manjkajočih povezavah*

Algoritem vrne veliko število povezav, za katere oceni, da bi lahko manjkale v vhodnem grafu. Intuicija nam pravi, da imamo v nekatere hipoteze več zaupanja, oziroma jih podpira več dokazov kot druge. Metoda zato napovedane povezave točkuje glede na

1: *funkcija* HYSPAN(H, T, ν)

▯ Pri danem grafu H in drevesu kod T vrne seznam točkovanih hipotez o manjkajočih povezavah v grafu.

▯ $H = (V, E, \ell)$ je vhodni graf.
▯ T je model (drevo kod).
▯ ν je največje še dovoljeno število uporabljenih navideznih povezav v delnem vpetju.

2: $\widetilde{V} \leftarrow$ množica ν navideznih vozlišč za vsako oznako $l \in L$
3: $\tilde{E} \leftarrow \emptyset$
4: $\widetilde{H} \leftarrow (V \cup \widetilde{V}, E \cup \tilde{E}, \tilde{\ell})$ ▯ $\tilde{\ell}$ je razširitev ℓ, ki vsebuje tudi oznake dodanih vozlišč in povezav
5: $\varsigma \leftarrow$ prazen seznam (dodanih povezav s pripadajočimi števili točk)
6: *za* vsak vzorec P iz T z zgolj 1 povezavo *:*
7: $\epsilon_0 \leftarrow$ prazno vpetje
8: Hyspan_Expand($\widetilde{H}, P, \epsilon_0, \varsigma$)
9: *konec zanke*
10: *vrni* ς
11: *konec funkcije*

Slika 3.5: Psevdokoda napovednega dela metode Hyspan. Rekurzivna funkcija za iskanje vpetij je opisana na sliki 3.6.

12: *procedura* HYSPAN_EXPAND($\widetilde{H}, P, \epsilon, \varsigma$)

□ Posodobi seznam povezav (ς) z vpetji vzorca P in postopek rekurzivno ponavlja s P-jevimi otroki (tj. iz P izpeljanimi večjimi vzorci).

□ $\widetilde{H} = (V \cup \widetilde{V}, E \cup \tilde{E}, \ell)$ je vhodni graf, razširjen s povezavami, dodanimi med dosedanjimi klici te procedure.

□ P je vzorec, ki ga želimo vpeti.

□ ϵ je vpetje P-jevega starša, dano v obliki seznama povezav iz $E \cup \tilde{E}$.

□ ς je seznam grafu dodanih povezav s pripadajočimi števili točk.

13: $\tilde{E} \leftarrow \tilde{E} \cup \{$nove navidezne povezave, ki ustrezajo zadnji terki v $\xi(P)\}$

14: $\widetilde{H} \leftarrow (V \cup \widetilde{V}, E \cup \tilde{E}, \tilde{\ell})$

15: *za* vsako vpetje ϵ' vzorca P v $\widetilde{H}$ *:*

□ ϵ' je razširitev ϵ, ki vključuje vpetje P-jeve zadnje povezave

16: *če* ϵ' vsebuje več kot v navideznih povezav *potem*

17: *nadaljuj z naslednjo iteracijo zanke*

18: *konec*

19: *če* ϵ' vsebuje vsaj eno nenavidezno vozlišče *potem*

20: *za* vsako navidezno povezavo e iz vpetja ϵ' *:*

21: ustrezno posodobi $\varsigma(e)$

22: *konec zanke*

23: *konec*

24: *za* vsakega otroka P' vzorca P *:*

25: Hyspan_Expand($\widetilde{H}, P', \epsilon', \varsigma$)

26: *konec zanke*

27: *konec zanke*

28: *konec procedure*

Slika 3.6: Psevdokoda napovednega dela metode Hyspan: rekurzivna funkcija za iskanje vpetij.

količino indicev, ki jim kažejo v prid.

Načini točkovanja hipotez so osnovani na velikosti vzorcev, na podlagi katerih so nastale ($|E_P|$), ter na njihovi podpori v učni množici ($\sigma(P, \mathscr{G})$). Označimo s $\mathscr{P}(e)$ množico vzorcev, za katere vsaj eno vpetje potrebujemo povezavo e. Zasnovali smo tri točkovalne funkcije.

Prvo točkovanje je osredotočeno na velike vzorce, zato opazuje le velikosti vzorcev, za katerih vpetje v grafu manjka povezava e.

$$s_{\mathrm{E}}(e) := \sum_{P \in \mathscr{P}(e)} \exp\left(|E_P|\right). \tag{3.4}$$

Recimo, da smo našli delno vpetje vzorca z velikim številom vozlišč in povezav. Ker gre za pogost vzorec v učnih podatkih, zaupamo v kompleksnost vzorca: če so prisotne vse povezave razen ene (oziroma največ ν), predpostavimo, da bi tudi manjkajoča povezava morala obstajati.

Eksponentna funkcija je bila uporabljena, ker pogostost naključnih vzorcev pada eksponentno z velikostjo vzorcev.

Drugo točkovanje je ravno nasprotno prvemu: ne zanima ga velikost vzorcev temveč daje poudarek njihovi pogostosti v učni množici. Pogost vzorec z dvema povezavama ima v tem točkovanju enako dokazno moč kot enako pogost vzorec s šestimi povezavami.

$$s_{\mathrm{S}}(e) := \sum_{P \in \mathscr{P}(e)} \sigma(P, \mathscr{G}). \tag{3.5}$$

Pri uporabi tega točkovanja bi bilo smiselno upoštevati kar vse vzorce, ne le pogoste. Omejitev najnižje še sprejemljive podpore vzorca je vseeno potrebna za omejitev časovne zahtevnosti postopka ter prostorske zahtevnosti drevesa kod.

Zadnje točkovanje je kombinacija prejšnjih dveh.

$$s_{\mathrm{X}}(e) := \max_{P \in \mathscr{P}(e)} \sigma(P, \mathscr{G}) \exp(|E_P|). \tag{3.6}$$

Uporabljen je največji produkt, ker so že preprosti poskusi pokazali, da je vsota produktov preveč občutljiva na vzorce, ki so bili odkriti po naključju. Pri tem točkovanju dobi veliko točk tista povezava, ki jo napove velik vzorec, ki ima hkrati visoko podporo.

21. vrstica algoritma posodobi število točk hipotez in je neodvisna od izbire točkovalne funkcije.

3.3.4 Časovna zahtevnost

Tako učni kot napovedni del metode rešujeta problem podgrafnega izomorfizma, zato sta NP-težki. Teoretična časovna zahtevnost torej narašča eksponentno z velikostjo vhoda.

Dejanski čas izvajanja je težko oceniti, ker je odvisen od mnogih dejavnikov oziroma spremenljivk, katerih vrednosti niso znane vnaprej. Čas za učni del metode je kar enak času izvajanja algoritma gSpan, ki so ga avtorji v svoji analizi [59] ocenili na $\mathcal{O}(kFS + rF)$, kjer je:

- k največje število različnih vpetij katerega izmed vzorcev v nek učni graf,
- F je število pogostih vzorcev v učni množici,
- S je velikost učne množice, tj. število učnih grafov,
- r je največje število različnih kod za nek pogost vzorec, do katerih lahko pridemo z razširitvijo minimalne kode nekega drugega pogostega vzorca za eno povezavo.

Prvi seštevanec v oceni časovne zahtevnosti predstavlja *čas odkrivanja vzorcev*, drugi del pa *čas rezanja neminimalnih kod*. Pred izvajanjem algoritma točno poznamo le vrednost spremenljivke S. Na čas izvajanja močno vpliva tudi število podgrafnih izomorfizmov, ki jih odkrije algoritem. To število je v izračun zajeto v spremenljivki k, nanj pa močno vpliva gostota grafov in število različnih oznak na vozliščih in povezavah učnih grafov. Bolj redki kot so grafi in več kot je različnih oznak, hitreje se bo postopek končal.

Časovna zahtevnost napovednega dela metode Hyspan je podobna zahtevnosti učnega dela s tremi pomembnimi razlikami. Prva je, da so pogosti vzorci podani kot vhodni podatek, zato ni potrebno izvajati rezanja neminimalnih kod. Druga razlika je, da imamo pri napovedovanju opravka le z enim grafom, zato lahko časovno zahtevnost zapišemo kot $\mathcal{O}(kF)$. Tretja razlika pa je, da pri napovednem delu v graf dodajamo navidezne povezave. S tem graf gostimo, kar, kot smo že omenili, povečuje število podgrafnih izomorfizmov in s tem spremenljivke k. Natančno razliko je v splošnem nemogoče izračunati.

Na število pogostih vzorcev (F) lahko vplivamo s parametrom najnižje dovoljene podpore vzorca (μ). Pri praktični uporabi metode torej iščemo ustrezno ravnotežje med hitrostjo in kvaliteto delovanja.

3.4 Ocena uspešnosti

V tem razdelku najprej opišemo podatke, zatem pa še druge metode, s katerimi smo primerjali rezultate naše metode. Rezultate poskusov in njihovo interpretacijo podajamo v naslednjem razdelku.

3.4.1 Množice podatkov

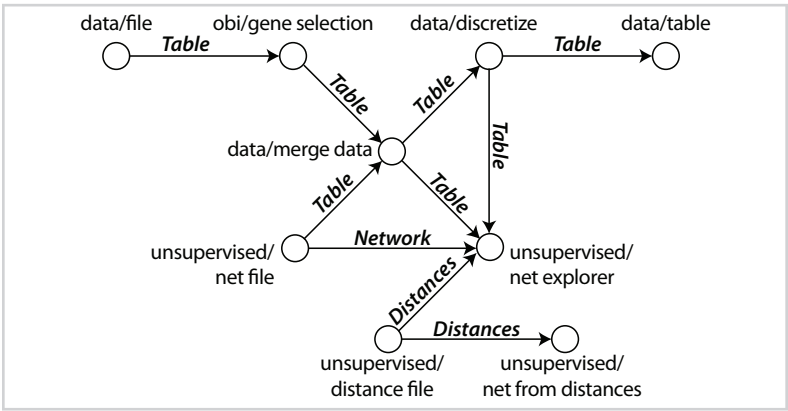
Predlagano metodo smo preizkusili na več množicah podatkov, ki opisujejo resnične pojave (*realni podatki*), za analizo vpliva lastnosti omrežij na delovanje metode pa tudi na umetno sestavljenih podatkih (*sintetični podatki*). Zbrane in zgenerirane podatke smo javno objavili kot del dopolnilnega gradiva znanstvenega članka [64] na naslovu `http://www.biolab.si/supp/hyspan`.

Realni podatki

Osnovna motivacija za razvoj metode je bila želja po obdelavi podatkovnih tokov iz grafičnega okolja programa za odkrivanje zakonitosti v podatkih Orange [56]. Iz dnevniških zapisov tega programa smo rekonstruirali podatkovne tokove rednih uporabnikov, ki so nam to dovolili. Nato smo sestavili množico majhnih omrežij, recimo ji *Orange*, v katerih vozlišča predstavljajo gradnike (podatkovne filtre, algoritme strojnega učenja, prikaze podatkov in modelov itd.), ki delajo s podatki, usmerjene povezave med njimi pa simbolizirajo tok podatkov. V množici je 594 usmerjenih označenih grafov, od katerih vsak vsebuje vsaj tri vozlišča in tri povezave. 417 grafov sestavlja ena sama povezana komponenta. Večina grafov je zelo redkih: mediana števila vozlišč v grafu je 6, mediana števila povezav pa 5. V množici je 75 različnih oznak na vozliščih in 13 na povezavah. Primer omrežja iz te množice je prikazan na sliki 3.7.

Množica grafov *myExperiment* je podobna prejšnji in vsebuje 1729 podatkovnih tokov programa *Taverna 2* – sistema za upravljanje s podatkovnimi tokovi v znanosti [55]. Surovi podatki so javno objavljeni na spletni strani *myExperiment.org*. Vozlišča predstavljajo algoritme ter vhode in izhode podatkovnega toka. Poleg 105 oznak za različne algoritme je torej na vozliščih še dodatna oznaka za slednje. Povezave predstavljajo pretok podatkov med vozlišči, podobno kot pri množici *Orange*, nimajo pa oznak. Mediana števila vozlišč v grafu je 9, povezav pa 8.

Tretja in zadnja realna množica podatkov, na kateri smo preizkusili svojo metodo, je *Vezja*. Gre za v obliki omrežij predstavljenih 98 shem elektronskih vezij, pridoblje-



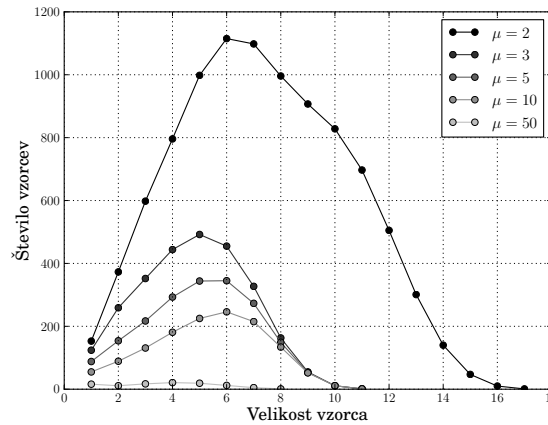
Slika 3.7: Primer označenega grafa iz množice *Orange*. Graf je abstrakcija podatkovnega toka, ki je prikazan na sliki 3.1; vozlišča so označena z imeni gradnikov, povezave pa s tipom podatkov, ki se pretaka čez povezavo.

nih s spletnega portala eCircuit Center (*ecircuitcenter.com*). Vozlišča predstavljajo 12 različnih elektronskih komponent, stičišča električnih povezav ter ozemljitev (v vsaki shemi je eno tako vozlišče). Skupaj je torej na vozliščih 14 različnih oznak. Neusmerjene povezave grafa predstavljajo električne povezave. Oznaka povezave označuje vrsto priključka elektronske komponente: na primer, napetostni vir ima pozitivni in negativni pol in temu ustrezno dva priključka. Različnih oznak na povezavah je 30. Grafi v tej množici so dvodelni, saj smo jo zasnovali tako, da ima vsaka povezava na eni strani elektronsko komponento, na drugi pa bodisi ozemljitev bodisi stičišče električnih povezav – četudi na shemi stičišča ni in bi povezavo lahko nadaljevali neposredno do naslednje elektronske komponente. Mediana števila vozlišč v grafu je 23, povezav pa 26.

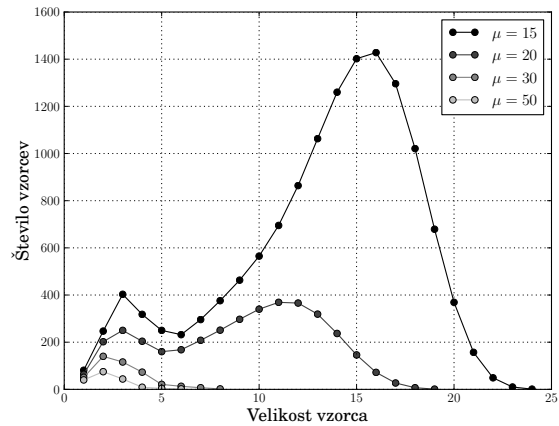
Slika 3.8 prikazuje histogram velikosti pogostih vzorcev, ki jih v podatkovnih množicah *Orange* in *myExperiment* odkrije algoritem gSpan pri različnih vrednostih parametra μ. Prav tak histogram za podatkovno množico *Vezja* je na sliki 3.9a.

Sintetični podatki

Sintetične podatke smo sestavljali z metodo, ki jo je predlagal Kuramochi [12]. Njen rezultat so neusmerjeni označeni grafi. Metoda najprej sestavi L potencialnih pogostih



(a) Množica podatkov Orange



(b) Množica podatkov myExperiment

Slika 3.8: Histogram velikosti z algoritmom gSpan odkritih vzorcev v množicah *Orange* in *myExperiment* pri različnih vrednostih parametra μ.

vzorcev. Število vozlišč v vsakem od teh je vnaprej izbrano kot naključno število iz eksponentne distribucije s pričakovano vrednostjo I. Struktura vzorcev je naključna: za vsako novo povezavo je kot eno krajišče enakomerno naključno izbrano obstoječe vozlišče, kot drugo pa prav tako, le da je med izbire dodana še možnost novega vozlišča. Vsako dodano vozlišče ali povezava dobi enakomerno naključno izbrano eno od N oznak.

Za vsak graf, ki ga metoda želi sestaviti, je najprej izbrana velikost (tj. število povezav) kot naključno število iz eksponentne porazdelitve s pričakovano vrednostjo T. Metoda nato v graf vstavlja vzorce, ki jih izbira naključno z vračanjem. To počne toliko časa, dokler graf ne doseže ali preseže prej izbranega števila povezav. Če bi jih z dodajanjem zadnjega vzorca presegel, je ta vzorec dodan s polovično verjetnostjo. Izbrana velikost grafa je tako le statistično pričakovana.

Vzorcev, ki jih dodaja v grafe, metoda ne izbira enakomerno naključno, pač pa že pred tem vsakemu vzorcu naključno dodeli tudi verjetnost, s katero bo izbran. Nekateri vzorci tako morda sploh ne bodo pogosti.

Kako točno vzorec vstaviti v graf, Kuramochi v opisu metode ne pojasni. Naša implementacija je za vsako vozlišče v izbranem vzorcu enakomerno naključno izbrala bodisi ustrezno vozlišče v grafu bodisi dodala novo vozlišče. Obstoječe vozlišče je primerno, če zadošča dvema pogojema:

- ima enako oznako kot vozlišče v vzorcu,
- morebitne že obstoječe povezave do ostalih vozlišč, ki so bila že izbrana kot slike vozlišč v trenutnem vzorcu, morajo prav tako nositi take oznake kot povezave v vzorcu.

Z drugimi besedami, naključno izbrane vzorce v graf dodamo tako, da oznake na obstoječih vozliščih in povezavah ustrezajo tistim v vzorcu, naključno ali po potrebi pa tudi dodamo nova vozlišča.

S tem algoritmom smo sestavili eno množico podatkov (recimo ji *Sintetični*), ki smo jo postavili ob bok realnim podatkom. Poleg tega smo zgradili še serijo množic podatkov različnih velikosti, s pomočjo katere smo lahko ocenili delovanje metode pri podatkih različnih velikosti.

Množica *Sintetični* sestoji iz $L = 10$ naključnih pogostih vzorcev s povprečno velikostjo $I = 5$ povezav. Na vozliščih in povezavah je $N = 5$ različnih oznak. Iz vzorcev

je bilo sestavljenih 500 grafov s pričakovano povprečno velikostjo $T = 10$ povezav; dejanska mediana števila vozlišč v grafih je 6, povezav pa 9. Dejanski histogram velikosti z algoritmom gSpan odkritih vzorcev je prikazan na sliki 3.9b. Primer naključnega grafa je na sliki 3.10.

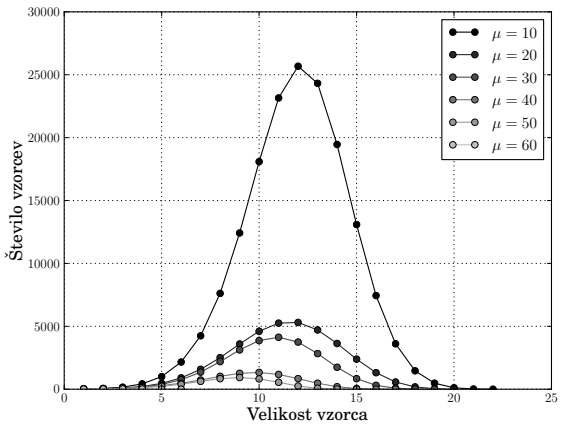
Za oceno delovanja metode pri različnih velikostih vhodnih grafov smo sestavili tudi zaporedje množic naključnih omrežij z vrednostjo parametra I med 2 in 5. Poskuse smo izvajali pri $T = 3 \cdot I$, $T = 4 \cdot I$ in $T = 5 \cdot I$. Uporabili smo $L = 5$ vzorcev in $N = 10$ oznak. Za vsako kombinacijo parametrov smo za izničenje vpliva naključnosti sestavili 50 množic, vsako s po 50 grafi.

3.4.2 *Način testiranja in mera uspešnosti*

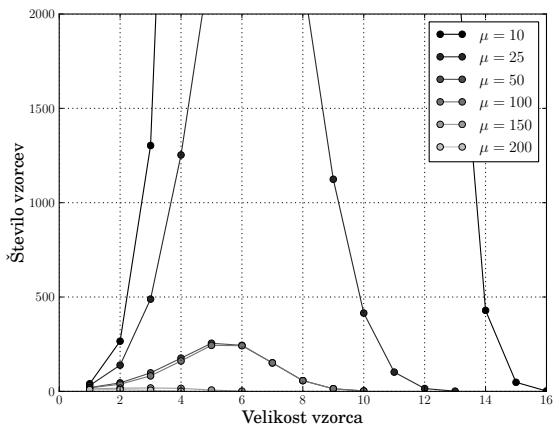
Na vsaki množici smo izvedli 10-kratno prečno preverjanje. Pri tem množice, ki je bila v posamezni iteraciji določena za testno množico, nismo neposredno uporabili. Dejanska testna množica je bila sestavljena tako, da smo vsak graf iz prej omenjene množice uporabili večkrat; vsakokrat z izbrisano po eno povezavo. Dejansko število testnih grafov je bilo tako večje od števila vseh grafov v množici: 3578 za množico *Orange*, 21366 za *myExperiments*, 2820 za *Vezja* in 5867 za množico *Sintetični*.

Cilj je bil pri vsakem testnem grafu napovedati odstranjeno povezavo. V podobnih poskusih [51, 53] je bila kot mera uspešnosti uporabljena *funkcija izgube* (angl. *loss function*). Pri dopolnjevanju majhnih omrežij je primerneje ocenjevati s pomočjo rangov. Pri vsakem testnem grafu smo napovedi, ki jih je dala metoda, uredili padajoče po točkah. Kot mero uspešnosti smo upoštevali *rang pravilne napovedi*. Pri enako točkovanih napovedih smo izračunali povprečni rang. Če dejansko izbrisana povezava sploh ni bila napovedana, torej pravilna napoved sploh ni bila našteta, pa smo kot rang pravilne napovedi upoštevali ∞.

Rezultate poskusov prikazujemo v obliki kumulativnih histogramov rangov pravilnih napovedi. To pomeni, da imamo na vodoravni osi naravna števila, ki predstavljajo range. Višina krivulje (y) pri vsaki vrednosti na vodoravni osi (x) ustreza deležu testnih grafov, pri katerih je bil rang pravilne napovedi enak x ali nižji. V idealnem primeru bi torej krivulja prečkala levi zgornji kot grafa, kar bi pomenilo, da je bila pravilna napoved vedno uvrščena najvišje. Histograme izrisujemo le do ranga 100, saj za še slabše napovedi menimo, da so neuporabne. V mnogih praktičnih uporabah metode bi pravzaprav upoštevali, pogledali oziroma uporabniku prikazali še manj napovedi (npr. 10 ali 20), zato je v praksi najpomembnejše gibanje krivulje v skrajno levem delu

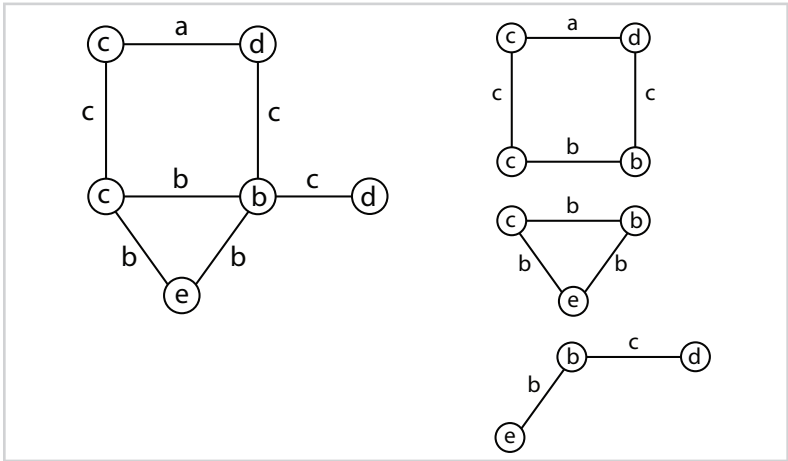


(a) Množica podatkov Vezja



(b) Množica podatkov Sintetični

Slika 3.9: Histogram velikosti z algoritmom gSpan odkritih vzorcev v množicah *Vezja* in *Sintetični* pri različnih vrednostih parametra μ. V slednji število vzorcev velikosti 8 pri $\mu = 10$ presega 60.000, zato je zaradi preglednosti prikaz omejen na manjšo zalogo vrednosti ordinate.



Slika 3.10: Umetno sestavljen označen graf (levo), za katerega so bili uporabljeni trije vzorci (desno).

histograma.

Vrednosti iste mere uspešnosti prikažemo tudi kot delež testnih grafov, na katerih je bila pravilna napoved uvrščena *dovolj visoko*. Da so rezultati na grafih različne velikosti bolj primerljivi, *dovolj visoko* definiramo kot zgornjih 5% vseh možnih napovedi, kjer se število vseh možnih napovedi razlikuje od grafa do grafa in je enako $|V| \cdot (|V| - 1) - |E|$.

Pri rezultatih poskusov podajamo tudi čase, ki jih je računalnik potreboval za izvedbo posameznega dela poskusa. Izvedba je potekala na enem procesorskem jedru Intelovega procesorja s frekvenco 3 GHz v računalniku z 8 GB delovnega pomnilnika. Ob časih podajamo tudi število pogostih vzorcev, ki jih je algoritem odkril v učni množici; podano je povprečje preko vseh 10 iteracij prečnega preverjanja.

3.4.3 *Referenčna in ostale metode*

V literaturi nismo zasledili metod, ki bi bile specializirane za reševanje problema dopolnjevanja majhnih omrežij. Hyspan smo zato primerjali s preprosto referenčno (angl. *baseline*) metodo, ki predlaga vsako možno vrsto povezave med obstoječimi in navideznimi vozlišči (vrsta povezave je določena z njeno oznako in oznakama njenih krajišč) in jih točkuje s številom učnih grafov, v katerih se taka povezava pojavi vsaj enkrat. To je ekvivalentno metodi Hyspan pri parametru $\mu = 1$, če omejimo velikost vzorcev na 1 povezavo in uporabimo točkovalno funkcijo s_S (enačba 3.5). Naj opomnimo, da sta

ostali dve točkovalni funkciji nesmiselni, ker so vsi vzorci enake velikosti.

Prednosti uporabe oznak, ki so na voljo v grafih, pri dopolnjevanju omrežij, smo preizkusili s primerjavo naše metode z drugimi metodami, ki ne upoštevajo oznak in so zasnovane za velika omrežja. Rešujejo torej *problem napovedovanja povezav*, ki smo ga opisali v razdelku 2.5.1. Najpreprostejše tovrstne metode temeljijo na podobnosti vozlišč. Predpostavljajo, da se bosta dve vozlišči povezali z večjo verjetnostjo, če sta si bolj *podobni*. Podobnost je odvisna od topoloških značilnosti grafa, različne metode pa se razlikujejo ravno po značilnostih, ki jih upoštevajo pri izračunu podobnosti. Preizkusili smo vse tovrstne metode, naštete v razdelku 2.5.1. Na naših podatkih so vse dale zelo podobne rezultate, zato objavljamo le rezultate Saltonovega indeksa, ki je bil od ostalih malenkostno boljši. Ta podobnost med vozliščema x in y izračuna tako:

$$s(x,y) = \frac{\left|N(x) \cap N(y)\right|}{\sqrt{|N(x)| \cdot \left|N(y)\right|}},$$

kjer je $N(v)$ množica sosedov vozlišča v.

Novejša, zapletenejša in pri delu z velikimi omrežji uspešnejša metoda je *naključni sprehod s ponovnimi začetki* (RWR [48]). Verjetnost pojavitve nove povezave med vozliščema x in y izračuna kot verjetnost vozlišča y v stacionarni porazdelitvi naključnega sprehoda iz točke x, ki se v vsakem koraku z verjetnostjo α vrne v izhodišče. α je edini parameter te metode. Preizkusili smo različne vrednosti med vključno $\alpha = 0{,}01$ in $\alpha = 0{,}99$. Za prvi poskus, kjer so vključene realne množice podatkov, smo nato uporabili vrednost 0,01, ki je dajala dobre rezultate in je tudi sicer v dostopni literaturi pogosto uporabljena vrednost. Pri drugem poskusu, kjer smo ugotavljali vpliv velikosti množic na delovanje metod, smo v vsakem delu poskusa uporabili tisto vrednost, ki je dajala najboljše rezultate. Praviloma bi morali najboljšo vrednost izbrati z notranjim prečnim preverjanjem (angl. *internal cross-validation*), a smo poskuse poenostavili in s tem metodi RWR dali rahlo prednost pred našo metodo.

Nadzorovanega naključnega sprehoda, ki se je v družbenih omrežjih izkazal še bolje, pri zastavljenem problemu ni možno uporabiti. Metoda namreč znanja, pridobljenega na enem omrežju, ni sposobna prenesti na drugo omrežje.

Nobena od naštetih metod za velika omrežja, razen *princip prednostne povezanosti* ne zna napovedati povezave med dvema nepovezanima komponentama grafa. Zato so bili iz vseh poskusov, kjer so bile uporabljene tudi te metode, izločeni testni grafi z več kot eno komponento. Prav tako pri ocenjevanju metod v teh poskusih niso bile upoštevane

napovedane oznake, saj jih metode za velika omrežja ne znajo napovedovati. Hyspan je sicer oznake upošteval in jih tudi napovedoval, a smo med vsakima dvema vozliščema upoštevali le najvišje točkovano napoved, ne glede na napovedano oznako.

3.4.4 *Izbira vrednosti parametrov metode Hyspan*

Kot smo že omenili v razdelku 3.3.1, smo za parameter μ (najnižja dovoljena podpora vzorca) preizkusili več različnih vrednosti. Začeli smo z višjimi vrednostmi, kar je zaradi nizkega števila odkritih vzorcev dalo slabe rezultate, a je bil čas izvajanja kratek. Vrednost parametra smo nato nižali, dokler niso bili časi napovedi za en graf višji od delčka sekunde. Prikazani rezultati torej govorijo o kvaliteti napovedi, ki jih metoda da v doglednem času. Z nadaljnjim nižanjem vrednosti parametra se čas izvajanja dvigne nerazumno visoko. Na vseh podatkovnih množicah smo izvedli poskuse še pri vrednosti $\mu = 1$, a z omejitvijo velikosti odkritih pogostih vzorcev na 5 povezav. Za analizo pomena večjih vzorcev smo z isto omejitvijo izvedli poskus tudi pri najnižji že sicer uporabljeni vrednosti μ za neposredno primerjavo.

Kot kažejo poskusi (npr. na slikah 3.12 in 3.13), v praksi izbira primerne vrednosti μ za posamezno domeno ni zahtevna. V vseh poskusih se namreč kvaliteta napovedi ni bistveno izboljšala, če smo μ zmanjševali pod vrednosti, pri katerih je čas izvajanja nerazumno narasel.

Hyspan ima še en parameter, ν, ki označuje število dovoljenih navideznih povezav v delnem vpetju. Zaradi mehanizma, ki smo ga uporabili za sestavljanje testnih množic, smo v teh poskusih vedno uporabili $\nu = 1$.

3.5 *Rezultati poskusov*

V tem razdelku bomo prikazali rezultate poskusov, ki kažejo, da:

- zmanjševanje zahtevane spodnje meje podpore vzorcev izboljša delovanje metode Hyspan, saj model sestavlja večje število vzorcev, a ima to negativen vpliv na čas izvajanja;
- čas izvajanja je moč skrajšati tudi z omejitvijo velikosti uporabljenih vzorcev, a prav tako za ceno dosežene uspešnosti;
- Hyspan daje boljše rezultate od drugih preizkušenih metod, iz česar zaključujemo, da je uporaba pogostih vzorcev z oznakami na vozliščih in povezavah v

kontekstu dopolnjevanja majhnih omrežij koristna;

- z večanjem grafov, ki jih želimo dopolniti, Hyspan izgublja svojo prednost pred drugimi metodami;
- Hyspan je kompleksnejša metoda od metode naključnih sprehodov in zato do rezultatov pride v bistveno daljšem času, predvsem pri večjih omrežjih.

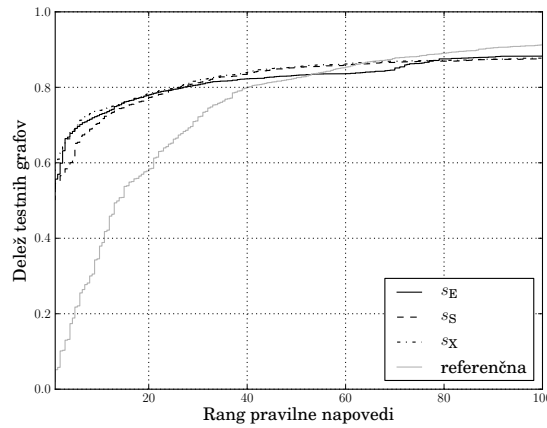
V prvem poskusu smo najprej primerjali delovanje treh točkovalnih funkcij (s_E, s_S and s_X) na dveh množicah podatkov, *Orange* in *Sintetični*. Rezultate prikazuje slika 3.11). Funkcija s_E daje slabše rezultate od preostalih dveh, ki sta si približno enakovredni. Pri nadaljnjih poskusih smo zato za točkovanje uporabljali funkcijo s_X.

Hyspan z uporabo večjih vzorcev premaga referenčno metodo. Pri dovolj visokih rangih sicer referenčna metoda dohiti ali celo prehiti Hyspan, saj predlaga tudi povezave, osnovane na vzorcih, ki se pojavljajo preredko, da bi jih Hyspan sploh upošteval. V praksi to ni pomembno, saj je za uporabnost metode bistvena njena sposobnost pravilne razvrstitve napovedi, tako da je pravilna napoved pogosto med prvimi.

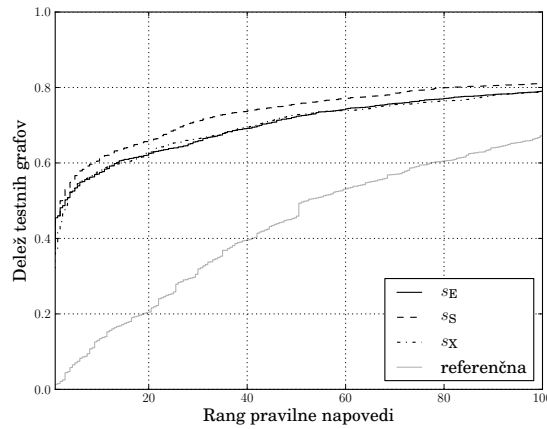
Sliki 3.12 in 3.12 ter tabela 3.1 prikazujeta iskanje kompromisa med številom vzorcev, na katerega vplivamo preko nastavitve parametra najnižje dovoljene podpore vzorca (μ), in kvaliteto delovanja metode Hyspan. Z zniževanjem vrednosti tega parametra dosežemo višjo uspešnost za ceno daljšega časa izvajanja. Previsoka vrednost parametra prepreči odkritje ustreznih vzorcev, kar onemogoča izdelavo dobrih hipotez. Ta pojav je viden na sliki 3.13b pri vrednosti $\mu = 200$, kjer je uspešnost metode nižja od uspešnosti referenčne metode. Po drugi strani uspešnost pri $\mu = 50$ ni bistveno višja od tiste pri $\mu = 100$, čas izvajanja pa se je več kot podvojil, kar je razvidno iz tabele 3.od.

Podoben pojav opazimo pri vseh množicah podatkov: ko pridemo do dovolj nizke vrednosti μ (ki se od množice do množice razlikuje), še nižje vrednosti močno podaljšujejo čase izvajanja in bistveno ne višajo dosežene uspešnosti. Izjemi sta množica *Sintetični* in nekoliko manj izrazito množica *myExperiment*, kjer omejitev velikosti vzorca na 5 in upoštevanje vseh podgrafov v množici ($\mu = 1$) da pri višjih rangih opazno boljši rezultat, a zopet za ceno bistvenega podaljšanja časa izvajanja.

To bi lahko pomenilo, da na določenih množicah veliki vzorci ne igrajo vloge pri dopolnjevanju omrežij. Na slikah 3.14 in 3.15 so histogrami, ki za vsako množico prikazujejo porazdelitev velikosti vseh pogostih vzorcev (črna krivulja) ter tistih pogostih vzorcev, ki so bili vsaj enkrat vpeti v testno omrežje preko tiste navidezne povezave, ki

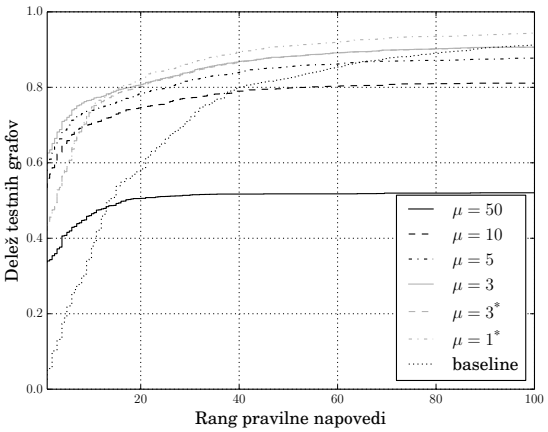


(a) Množica podatkov Orange, $\mu = 5$

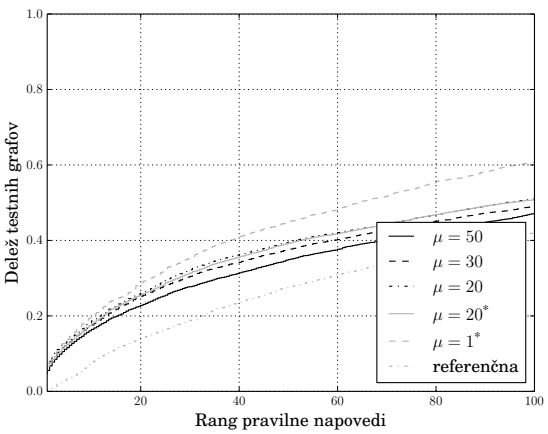


(b) Množica podatkov Sintetični, $\mu = 100$

Slika 3.11: Primerjava točkovalnih funkcij. Za vsak rang (os x) ordinata prikazuje delež testnih grafov, pri katerih je določena metoda pravilno napoved uvrstila med prvih x.

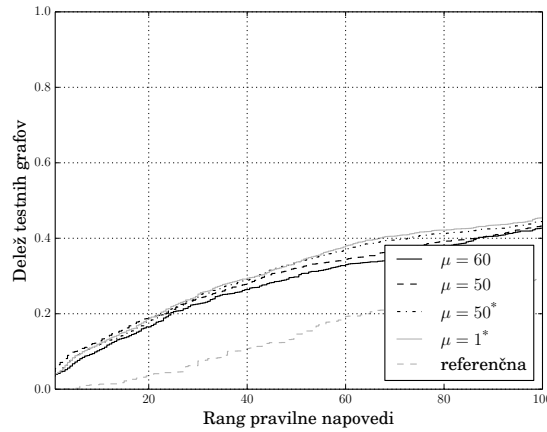


(a) Množica podatkov *Orange*

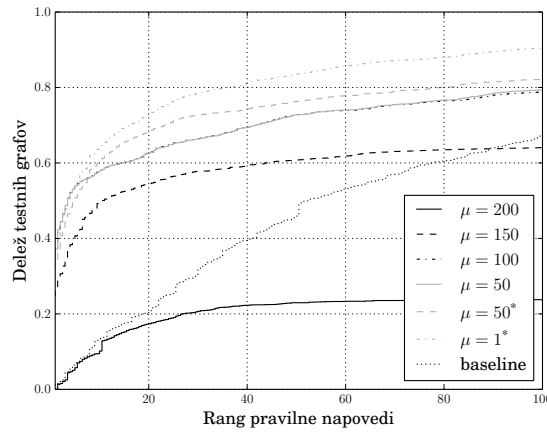


(b) Množica podatkov *myExperiment*

Slika 3.12: Vpliv števila vzorcev na uspešnost metode Hyspan na množicah podatkov *Orange* in *myExperiment*. Število vzorcev je regulirano preko parametra μ. Uporabljena je bila točkovalna funkcija s_X. Podroben opis načina prikaza rezultatov je v opisu slike 3.11. *: omejitev velikosti vzorcev na 5 povezav



(a) Množica podatkov *Vezja*



(b) Množica podatkov *Sintetični*

Slika 3.13: Nadaljevanje slike 3.12. Vpliv števila vzorcev na uspešnost metode Hyspan na množicah podatkov *Vezja* in *Sintetični*.
*: omejitev velikosti vzorcev na 5 povezav

Tabela 3.1: Delež testnih grafov, pri katerem je posamezna metoda pravilno napoved uvrstila med prvih pet, izvajalni časi za vseh 10 iteracij prečnega preverjanja in povprečno število odkritih (in za napovedovanje uporabljenih) vzorcev. (Za podatkovno množico *Vezja* so vrednosti μ nad 60 neprimerne zaradi nizkega števila omrežij, pri vrednostih pod 50 pa je čas izvajanja algoritma neprimerno visok.)
*: omejitev velikosti vzorcev na 5 povezav

(a) Množica podatkov *Orange*

μ	50	10	5	3	3*	1*	(ref. m.)
Rang prav. nap. $\leqslant 5$	0,40	0,65	0,68	0,71	0,58	0,58	0,18
Čas učenja [s]	1,99	6,20	8,50	9,81	4,77	15,04	2,79
Čas napovedovanja [s]	3,3	20	37	41	39	250	8,3
Povp. št. vzorcev	81	983	1843	2539	1565	15473	196

(b) Množica podatkov *myExperiment*

μ	50	30	20	20*	1*	(ref. m.)
Rang pravilne napovedi $\leqslant 5$	0,11	0,12	0,13	0,12	0,13	0,03
Čas učenja [s]	50	177	4307	766	4188	21
Čas napovedovanja [s]	540	1419	4780	4750	744205	54
Povp. št. vzorcev	143	360	1340	689	124016	398

(c) Množica podatkov *Vezja*

μ	60	50	50*	1*	(referenčna metoda)
Rang pravilne napovedi $\leqslant 5$	0,07	0,10	0,08	0,08	0,00
Čas učenja [s]	1,1	17	5	41	0,4
Čas napovedovanja [s]	45	480	209	20143	19
Povp. št. vzorcev	33	528	221	18095	45

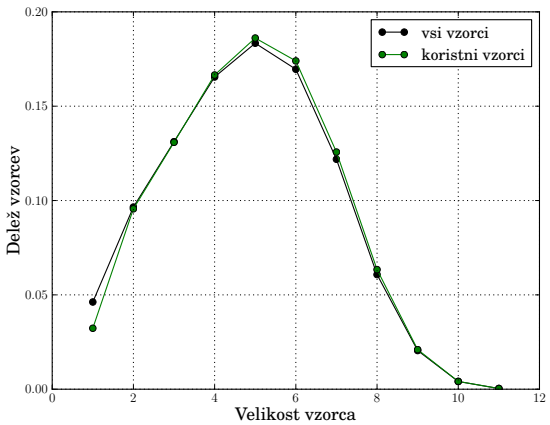
(d) Množica podatkov *Sintetični*

μ	200	150	100	50	50*	1*	(r. m.)
Rang pravilne napovedi $\leqslant 5$	0,05	0,40	0,53	0,53	0,50	0,54	0,06
Čas učenja [s]	0,6	11	106	259	91	812	0,84
Čas napovedovanja [s]	2,7	58	912	1936	762	183065	13,2
Povp. št. vzorcev	3	69	1009	1038	572	965500	42

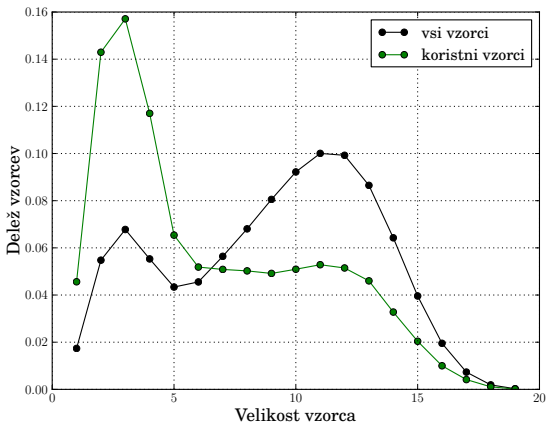
je dala pravilno napoved (zelena krivulja). Čeprav slike kažejo na večjo pomembnost manjših vzorcev na dveh podatkovnih množicah, pa nekoliko večji vzorci vseeno niso nepomembni. Na to kaže tudi dosežena uspešnost dopolnjevanja omrežij, ki je na vseh uporabljenih vrstah podatkov pri najvišjih rangih boljša, če ne omejimo velikosti vzorcev. Le na sintetični množici kaže, da večji vzorci zmedejo točkovalno funkcijo na težjih testnih primerih. Posledično omejitev velikosti vzorcev izboljša uspešnost pri višjih rangih, kar pa ni bistvenega pomena za praktično uporabo.

Hyspan smo primerjali tudi z metodami, zasnovanimi za velika omrežja, ki ne upoštevajo oznak na vozliščih in povezavah. Rezultati so podani na slikah 3.16 in 3.17. Zaradi jasnosti prikaza je od metod, ki temeljijo na podobnosti med vozlišči, podan le rezultat za Saltonov indeks, saj je izmed vseh tovrstnih metod dosegel najvišjo uspešnost. Iz rezultatov je razvidno, da so najvišje rangirane napovedi metod za velika omrežja običajno napačne. Celo naša referenčna metoda je z uporabo oznak dosegla boljši rezultat. Na množici podatkov *Orange*, ki vsebuje precej majhne grafe, metode za velike grafe dohitijo delež pravilnih napovedi, kot ga dosega Hyspan, okrog ranga 20. V takih grafih pri napovedovanju brez oznak namreč ni veliko možnih povezav, zato tudi pri slabem rangiranju napovedi pravilna napoved zasede relativno visoko mesto. Na množicah podatkov *myExperiment* in *Vezja* je razlika v prid metodi Hyspan še bolj očitna.

Preostale poskuse smo namenili raziskavi uporabnosti razvite metode za različne velikosti omrežij. Tabela 3.2 prikazuje povprečne uspešnosti preko 50 ponovitev poskusa za vsako kombinacijo parametrov (I je pričakovana velikost pogostega vzorca, T pa pričakovana velikost posameznega omrežja v množici). Hyspan je uspešen na množicah majhnih grafov. Z večanjem grafov je njegova uspešnost čedalje slabša, medtem ko metoda naključnih sprehodov s ponovnimi začetki daje vedno boljše rezultate. Za parameter μ smo uporabili vrednost 10 pri množicah z vzorci pričakovane velikosti 2 oziroma 3 povezave in 20 pri vzorcih velikosti 4 oziroma 5 povezav.

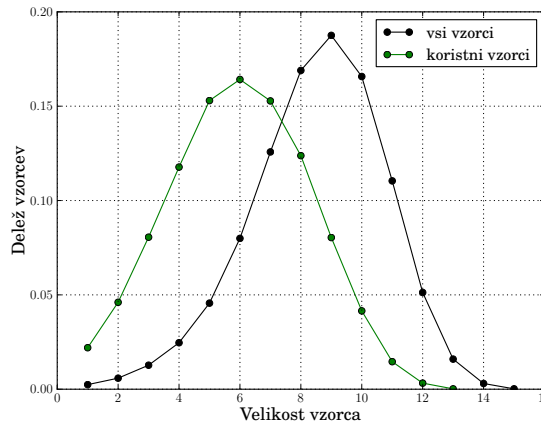


(a) Množica podatkov Orange, $\mu = 3$

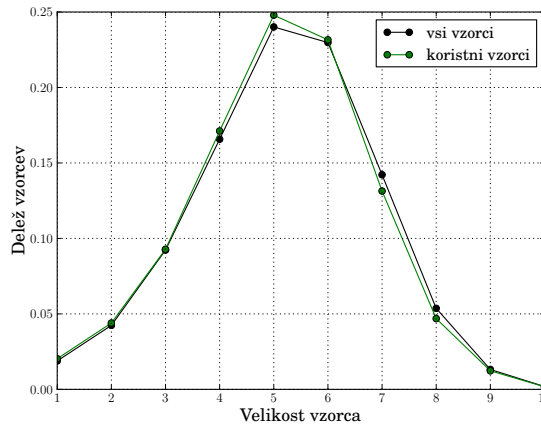


(b) Množica podatkov myExperiment, $\mu = 20$

Slika 3.14: Porazdelitev velikosti vseh pogostih vzorcev ter vzorcev, ki so z vpetjem preko prave navidezne povezave vsaj enkrat glasovali za pravilno napoved.

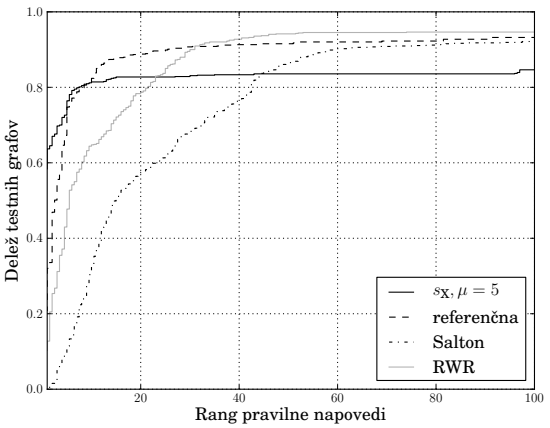


(a) Množica podatkov Vezja, $\mu = 50$

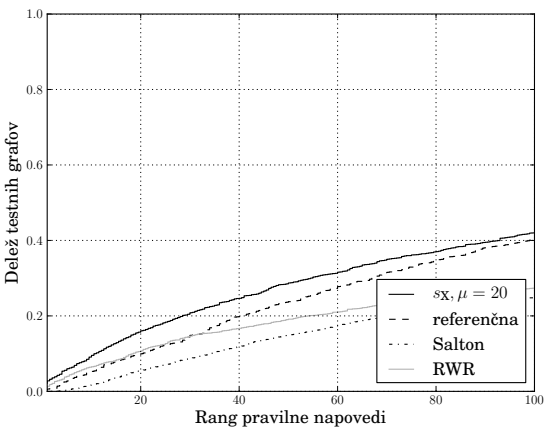


(b) Množica podatkov Sintetični, $\mu = 50$

Slika 3.15: Porazdelitev velikosti vseh pogostih vzorcev ter vzorcev, ki so z vpetjem preko prave navidezne povezave vsaj enkrat glasovali za pravilno napoved.

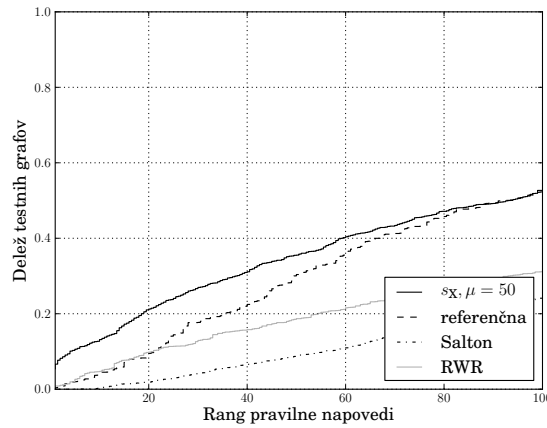


(a) Množica podatkov *Orange*, $\mu = 5$

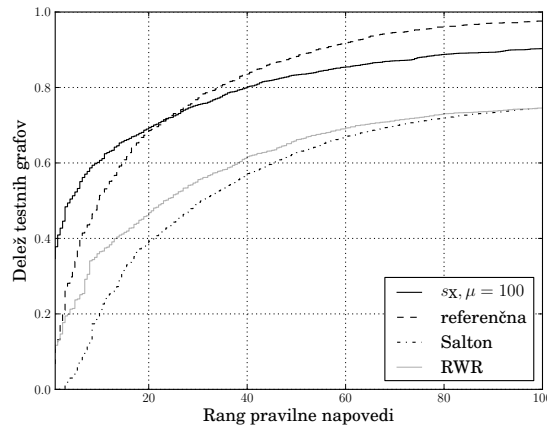


(b) Množica podatkov *myExperiment*, $\mu = 20$

Slika 3.16: Primerjava z metodami za velika omrežja, ki ne upoštevajo oznak, na množicah podatkov *Orange* in *myExperiment*. Hyspan kot točkovalno funkcijo uporablja s_X. Rezultati za Hyspan in referenčno metodo se razlikujejo od tistih na sliki 3.11, ker so bili testni grafi z več nepovezanimi komponentami odstranjeni. Podroben opis načina prikaza rezultatov je v opisu slike 3.11.



(a) Množica podatkov *Vezja*, $\mu = 50$



(b) Množica podatkov *Sintetični*, $\mu = 100$

Slika 3.17: Nadaljevanje slike 3.16. Primerjava z metodami za velika omrežja na množicah podatkov *Vezja* in *Sintetični*.

Tabela 3.2: Primerjava Hyspana z referenčno metodo ter metodo naključnih sprehodov s ponovnimi začetki. V tabeli je prikazan delež testnih grafov, pri katerih je metoda pravilno napoved uvrstila med prvih 5 % možnih napovedi.

	velikost vzorcev (I)			
metoda	2	3	4	5
pričakovana povprečna velikost omrežja: $T = 3 \cdot I$ povezav				
referenčna	0,1461	0,1550	0,1071	0,1145
Hyspan	0,5205	0,4510	0,3919	0,3790
RWR	0,2224	0,2694	0,2795	0,3033
pričakovana povprečna velikost omrežja: $T = 4 \cdot I$ povezav				
rreferenčna	0,1857	0,1661	0,1564	
Hyspan	0,4421	0,4155	0,3945	
RWR	0,1732	0,2106	0,2623	
pričakovana povprečna velikost omrežja: $T = 5 \cdot I$ povezav				
rreferenčna	0,1761	0,1783		
Hyspan	0,4366	0,3639		
RWR	0,1862	0,2381		

4

Časovni razvoj lokalnih struktur

Pogosti označeni vzorci (podgrafi) so uporabno orodje za analizo omrežij, a imajo seveda tudi pomanjkljivosti. Preveč kompleksnih lokalnih lastnosti zelo velikih omrežij ne moremo opazovati zaradi prevelike časovne in pomnilniške zahtevnosti algoritmov za njihovo odkrivanje, a iskanje pogostih vzorcev odpove že pri zmerno velikih omrežjih z nekaj tisoč povezavami. V tem poglavju si bomo ogledali drug opis lokalne strukture omrežja, tako imenovane *grafke* oziroma z njimi povezane *orbite vozlišč*. Razviti so bili za potrebe analize omrežij s področja bioinformatike. Nato bomo pokazali, kako *orbite povezav* lahko uporabimo za opazovanje razvoja neoznačenih grafkov v omrežjih, ki se spreminjajo s časom.

Namen raziskav, opisanih v tem poglavju, je bil odkriti splošne zakonitosti časovnega razvoja grafkov v omrežjih ter postopke za analizo konkretnih pojavov v posameznih omrežjih. Rezultati take analize so potencialno uporabni na primer za napovedovanje časovnega razvoja omrežja ali primerjavo sorodnih omrežij.

4.1 Grafki in orbite

Definicija 1: *Grafki* (tudi *grafleti* [65], angl. *graphlets* [11]) so *majhni neusmerjeni povezani inducirani podgrafi* velikosti od 2 do 5 vozlišč.

Opis lokalne strukture omrežja z grafki pomeni opazovanje vsake povezane majhne podmnožice vozlišč. Vsaka taka podmnožica točk enolično določa induciran graf.

Definicija 2: Na množico vozlišč $S \subseteq V$ induciran podgraf (angl. *induced subgraph* ali *vertex-induced subgraph*) grafa $G(V, E)$ je graf z množico vozlišč S in vsemi povezavami iz G, ki imajo obe krajišči v tej množici. Označimo ga z $G[S]$ oziroma $G\{s_1, s_2, ...\}$:

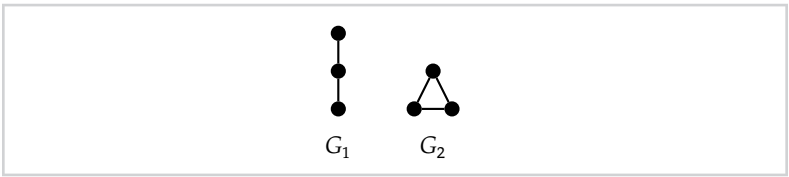
$$G[S] = G\{s_1, s_2, ..., s_n\} = \left(S, \left\{ij \in E | i, j \in S\right\}\right). \tag{4.1}$$

Kot bomo videli v nadaljevanju, je različnih (neizomorfnih) grafkov velikosti od 2 do 5 vozlišč 30. Označimo jih z $G_0, \ldots G_{29}$. Če preštejemo pojavitve različnih induciranih grafkov v omrežju, dobimo 30 števil, ki opisujejo njegovo lokalno strukturo:

$$\forall i = 0, \ldots, 29 : N_i(G) := |\{S \subseteq V(G) \mid G[S] \cong G_i\}|, \tag{4.2}$$

kjer simbol $\cong$ označuje izomorfnost.

Definicijo 1 bi v principu lahko razširili tako, da bi obravnavali tudi podgrafe z več kot 5 vozlišči, a so se v dostopni literaturi vsi raziskovalci omejili na 5 vozlišč. Princip



Slika 4.1: Dva grafka s po tremi vozlišči. V G_1 je možno najti dve različni vlogi vozlišč, v trikotniku pa so vsa vozlišča enakovredna.

dela z večjimi grafki bi bil sicer enak, a zaradi znane lastnosti omrežij, *majhnega sveta* (angl. *small world* [5]), bi se raztezali preko celotnega omrežja. Tako to ne bi bila več lokalna lastnost. Poleg tega število neizomorfnih grafkov narašča kot eksponentna funkcija kvadrata števila vozlišč.

Število pojavitev nekega grafka v omrežju lahko postavimo ob bok globalnim lastnostim omrežja, kot so število vozlišč, število povezav, število povezanih komponent, polmer idr. Število grafkov tipa nič (0) je natanko enako kar številu povezav v omrežju. Da vrednosti med različnimi omrežji postanejo primerljive, opazujemo *relativno frekvenco grafkov*:

$$R_i(G) = \frac{N_i(G)}{\sum_{j=1}^{29} N_j(G)}. \tag{4.3}$$

4.1.1 Orbite vozlišč

Čeprav bi v principu lahko šteli tudi, kolikokrat se *posamezno vozlišče* pojavi v določenem grafku, in tako vsako vozlišče opisali z vektorjem 30 števil, se nam tu ponuja še bolj informativna količina. Štejemo lahko, kolikokrat se vozlišče pojavi v posamezni *vlogi*.

Neformalno sta vlogi dveh vozlišč v grafku različni, če ju je možno razločiti. Poglejmo si primer dveh grafkov na treh točkah na sliki 4.1. Medtem ko vlog vozlišč v trikotniku (G_2) ne moremo razločiti, na poti dolžine dveh povezav (G_1) lahko neko vozlišče nastopa bodisi kot krajno (začetno ali končno) bodisi kot sredinsko vozlišče.

Opišimo to intuitivno idejo na formalen način. Zapišimo nek grafek kot $G_i = (V(G_i), E(G_i))$.

Definicija 3: Avtomorfizem f je bijektivna preslikava množice vozlišč $V(G_i)$ v to isto množico, ki ohranja povezanost:

$$(u,v) \in E(G_i) \;\Leftrightarrow\; \big(f(u), f(v)\big) \in E(G_i)\,. \tag{4.4}$$

V matematični teoriji diskretnih struktur je znano, da avtomorfizmi tvorijo permutacijsko grupo pod operacijo kompozicije. Za množico, nad katero deluje permutacijska grupa, obstaja razbitje v *orbite*; avtomorfizmi vedno preslikavajo elemente znotraj ene orbite. Ta koncept ustreza prej opisanemu neformalnemu konceptu vlog vozlišč.

Definicija 4: Grafkovne stopnje (angl. *graphlet degrees* [20]) vozlišča merijo število pojavitev tega vozlišča v različnih vlogah oziroma *orbitah* (angl. *orbits*) v grafkih glede na grupo avtomorfizma grafka.

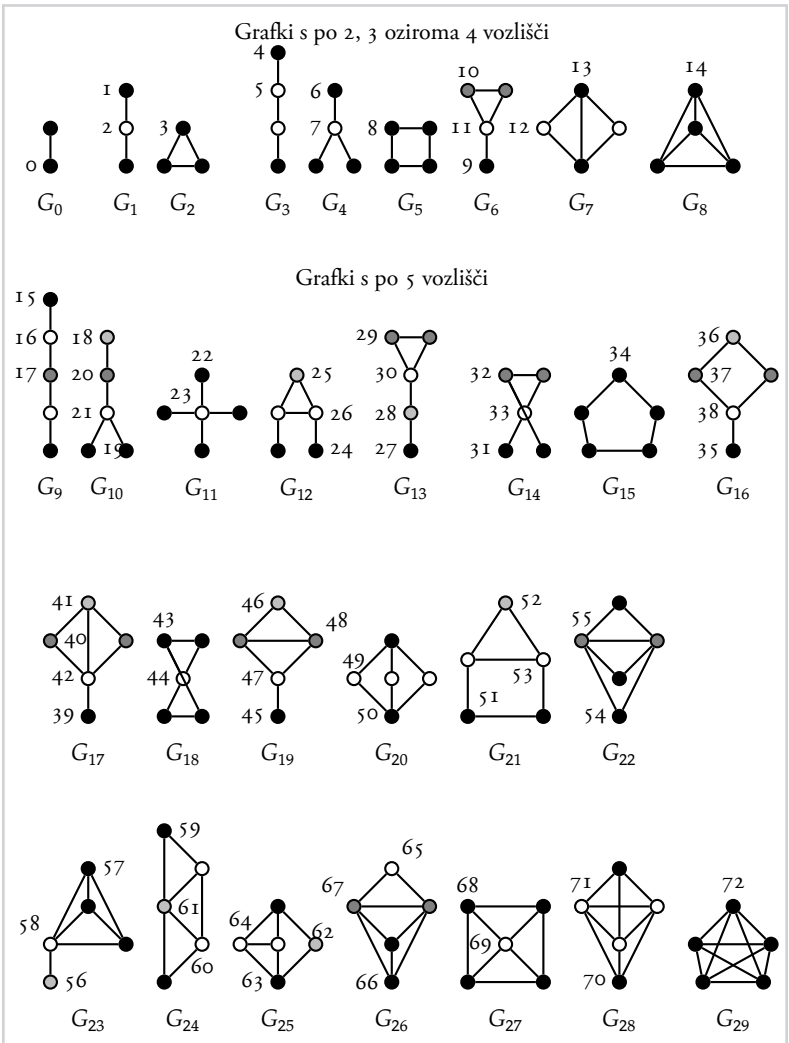
V grafkih velikosti do 5 vozlišč obstaja 73 orbit vozlišč, kot kaže slika 4.2. Vsako vozlišče omrežja torej lahko opišemo s 73-razsežnim vektorjem (*GDV*, *Graphlet Degree Vector*), ki vsebuje vse njegove grafkovne stopnje. Ime *grafkovne stopnje* izhaja iz dejstva, da gre za posplošitev običajne stopnje vozlišča. Ničta grafkovna stopnja oziroma število pojavitev vozlišča v ničti orbiti namreč ustreza številu povezav, katerih krajišče je, torej stopnji vozlišča. Tako kot opazujemo porazdelitev stopenj vozlišč in na podlagi tega primerjamo omrežja, lahko opazujemo tudi porazdelitve posameznih grafkovnih stopenj [20]. (Naj na tem mestu omenimo, da je bila skoraj istočasno s strani drugih avtorjev uvedena tudi nekoliko ožja posplošitev stopnje vozlišča. V realnih omrežjih so namreč opazovali zgolj porazdelitve pojavitev vozlišč v klikah velikosti od 2 do 5 vozlišč, torej grafkovne stopnje za grafke G_0, G_2, G_8 in G_{29} [66].)

Ta razdelek lahko povzamemo v dveh definicijah, kjer bomo določili tudi notacijo za definirane koncepte.

Definicija 5: Orbita vozlišča v v grafku G_i je orbita $\mathrm{Orb}^{G_i}(v)$ avtomorfizmov nad vozlišči tega grafka.

Orbite so množice vozlišč, pogosto pa je z njimi lažje operirati, če jih oštevilčimo. Vrstni red orbit na sliki 4.2 je arbitraren, a konsistentno uporabljan v vsej znanstveni literaturi, povezani z grafki.

Definicija 6: Grafkovna stopnja j *vozlišča* v v omrežju G je število grafkov v G, v katerih vozlišče v nastopa v orbiti j.



Slika 4.2: Vsi grafki velikosti do 5 vozlišč. Vozlišča so oštevilčena z orbitami. Znotraj vsakega grafka ista barva predstavlja isto orbito. Vrstni red grafkov in orbit je povzet po [20].

4.1.2 Orbite povezav

Na enak način kot pri vozliščih lahko definiramo tudi vloge oziroma orbite povezav [21].

Definicija 7: Orbita povezave e v grafku G_i je orbita $\mathrm{Orb}_e^{G_i}(e)$ avtomorfizmov grafka, ki delujejo nad povezavami in ohranjajo dotikanje povezav. (Za dotikajoči se povezavi štejemo povezavi, ki imata eno skupno krajišče.)

Avtorji te definicije v svojem delu omenjajo še drugo, za katero brez dokaza navajajo, da je ekvivalentna:

$$\mathrm{Orb}_e^{G_i}(uv) = \left\{zw \in E(G_i) \mid \exists \text{ avtomorfizem } f \text{ grafa } G_i : z = f(u), w = f(v)\right\}. \tag{4.5}$$

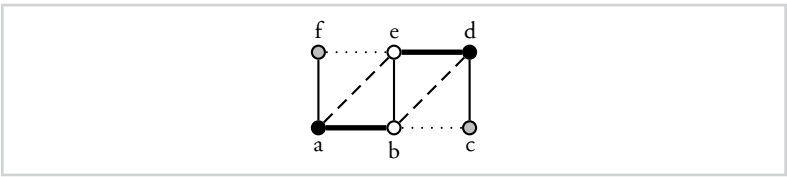
(Ker so grafki neusmerjeni, vrstni red naštevanja vozlišč pri povezavi ni pomemben: $uv \approx vu$.)

Da sta definiciji na majhnih grafkih ekvivalentni, lahko preverimo z izčrpnim naštevanjem. Za grafke velikosti do 4 vozlišč to brez težav storimo ročno. Za večje (5 in, če bi želeli, več vozlišč) pa to ni potrebno, saj si lahko pomagamo s krepkim Whitneyevim izrekom o izomorfizmu grafov oziroma Jungovim konstruktivnim dokazom [67]. Izrek pravi, da vsakemu izomorfizmu f med grafoma z vsaj 5 vozlišči pripada natanko en izomorfizem nad povezavami g in obratno. Dokaz pokaže, da je prehod med enim in drugim *naraven*: $g(uv) = \big(f(u)f(v)\big)$. To kaže na ekvivalentnost obeh definicij.

Iz tega očitno sledi, da imata dve povezavi v isti orbiti tudi enaki orbiti krajišč. Povedano drugače, iz orbite povezave lahko razberemo orbiti njenih krajišč – ne pa tudi katero od krajišč pripada kateri od orbit, če sta različni.

Ekvivalentnost zgornjih definicij nič ne pove o tem, ali iz orbit dveh vozlišč lahko razberemo orbito povezave med njima. Za nek graf namreč lahko obstajata avtomorfizma $f_1 : u \mapsto u'$ in $f_2 : v \mapsto v'$, zato so u in u' ter v in v' paroma v istih orbitah, vendar ne obstaja en izomorfizem, ki bi slikal tako $u \mapsto u'$ kot tudi $v \mapsto v'$. Posledično ne obstaja izomorfizem nad povezavami, ki bi slikal $uv \mapsto u'v'$ in sta povezavi uv in $u'v'$ v različnih orbitah.

Najmanjši graf, kjer se to zgodi, vsebuje 6 vozlišč in je prikazan na sliki 4.3. Če bi se iz orbit vozlišč dalo enolično razbrati orbito povezave, bi morali biti povezavi ab in bd v isti orbiti. Avtor primera je Tomaž Hočevar.



Slika 4.3: Graf s 6 vozlišči z označenimi orbitami. Orbite ločimo po barvah v primeru vozlišč oziroma šrafurah v primeru povezav.

Trditev 1: Iz orbit krajišč lahko pri grafkih velikosti do vključno 5 vozlišč enolično določimo orbito povezave.

Trditev lahko dokažemo z izključnim naštevanjem.

Na sliki 4.4 so naštete vse orbite povezav na grafkih z od 2 do 5 vozlišči. Vsaka orbita je izpisana zgolj enkrat; orbite povezav, na katerih ni izpisana številka, je možno razbrati iz orbit (barv) vozlišč s pomočjo zadnje trditve. Vrstni red orbit sledi leksikografsko urejenim parom orbit krajišč in ni enak vrstnemu redu v [21].

Podobno kot pri vozliščih tudi številu pojavitev povezave v orbiti z indeksom i rečemo *i-ta grafkovna stopnja povezave*. Vse grafkovne stopnje ene povezave sestavimo v vektor edge-*GDV* (*Edge Graphlet Degree Vector* [21]).

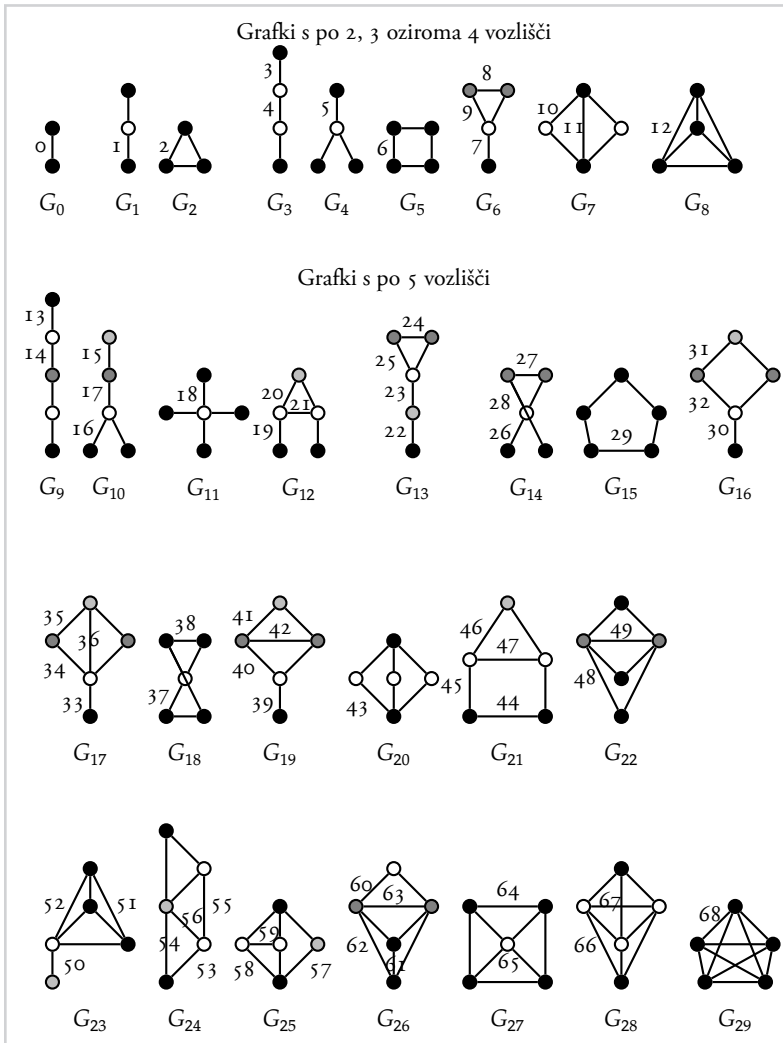
4.2 Razvoj grafkov v rastočih omrežjih

Pri svojem delu smo se osredotočali na omrežja, ki se spreminjajo, na primer s časom. Podobno kot v prejšnjem poglavju smo se tudi pri opazovanju časovnega razvoja lokalnih struktur omejili na rast omrežij. Osnoven dogodek, ki pomeni spremembo v smislu rasti omrežja, je dodajanje nove povezave:

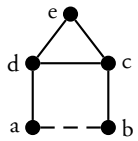
$$G(V, E) + uv \coloneqq G'(V, E \cup \{uv\}), \text{ kjer } u, v \in V \text{ in } uv \notin E. \tag{4.6}$$

Za potrebe analize časovnega razvoja lokalnih struktur lahko predpostavimo, da G že na začetku vsebuje vsa vozlišča, ki bodo kadarkoli nastopala kot krajišča katere od povezav. Posamezno vozlišče brez povezav namreč ni izomorfno nobenemu od grafkov.

Nova povezava vpliva na lokalno strukturo omrežja, tudi na število induciranih grafkov določenega tipa. Nekateri grafki izginejo, še več pa se jih pojavi. Poglejmo si



Slika 4.4: Orbite povezav na vseh grafkih velikosti do 5 vozlišč. Različne barve vozlišč znotraj enega grafka označujejo različne orbite. Dve povezavi z enakima orbitama krajišč sta v isti orbiti. Vrstni red orbit sledi leksikografsko urejenim parom orbit krajišč.



(a) Slika omrežja

podmnožica točk	sprememba grafkov	nove orbite točk
$\{a,b,c,d,e\}$	$G_{21} \leftarrow G_{21}$	$a:51, b:51, c:53, d:53, e:52$
$\{a,b,c,d\}$	$G_5 \leftarrow G_3$	$a:8, b:8, c:8, d:8$
$\{a,b,c,e\}$	G_3 (nov grafek)	$a:4, b:5, c:5, e:4$
$\{a,b,d,e\}$	G_3 (nov grafek)	$a:5, b:4, d:5, e:4$
$\{a,b,c\}$	G_1 (nov grafek)	$a:1, b:2, c:1$
$\{a,b,d\}$	G_1 (nov grafek)	$a:1, b:2, d:1$
$\{a,b\}$	G_0 (nov grafek)	$a:0, b:0$

(b) Seznam sprememb grafkov v omrežju

Slika 4.5: Primer omrežja z novo povezavo med vozliščema a in b.

primer na sliki 4.5, kjer je ilustrirano dodajanje povezave med vozliščema a in b v majhnem omrežju. Seznam sprememb grafkov, do katerih je pri tem prišlo, je izpisan pod sliko. Do le ene spremembe je prišlo pri grafkih velikosti pet vozlišč, saj jih ima le toliko tudi celo omrežje. Do treh sprememb je prišlo pri grafkih velikosti štirih vozlišč, pri čemer se ob nastanku grafkov G_3 ni zmanjšalo število drugih grafkov, saj podgrafa, inducirana na četvericah (a,b,c,e) in (a,b,d,e), pred dodajanjem povezave sploh nista bila povezana. Očitno je, da natanko en grafek G_0 nastane pri vsaki dodani povezavi.

4.2.1 Orodja

Trditev 2 (Lokalnost spremembe): Recimo, da v omrežje $G = G(V,E)$ dodamo novo povezavo in dobimo $G + uv = G'$. Števila grafkov posameznega tipa v G' lahko izračunamo iz števil grafkov v G, če najdemo vse grafke, inducirane v G' na take množice vozlišč, ki vsebujejo u in v.

Dokaz: Naj bo $\mathscr{S}$ množica vseh v grafu G' povezanih podmnožic točk $S \subseteq V$ pri pogoju $2 \leqslant |S| \leqslant 5$. Za vsako množico $S \in \mathscr{S}$ obstaja izomorfizem med $G'[S]$ ter nekim grafkom G_i in velja natanko ena od treh možnosti.

1. $\{u,v\} \nsubseteq S$, torej $G'[S]$ ne vsebuje nove povezave in zato $G[S] \cong G'[S] \cong G_i$. Grafek i ostane nespremenjen z dodajanjem povezave uv.

2. $u,v \subseteq S$ in $G[S]$ je povezan, torej $\exists j : G[S] \cong G_i$. Na teh točkah v G' ni induciran G_j, temveč $G_i \cong G'[S]$. G_i zagotovo obstaja, saj smo na sliki 4.2 našteli vse povezane grafke na 2-5 vozliščih.

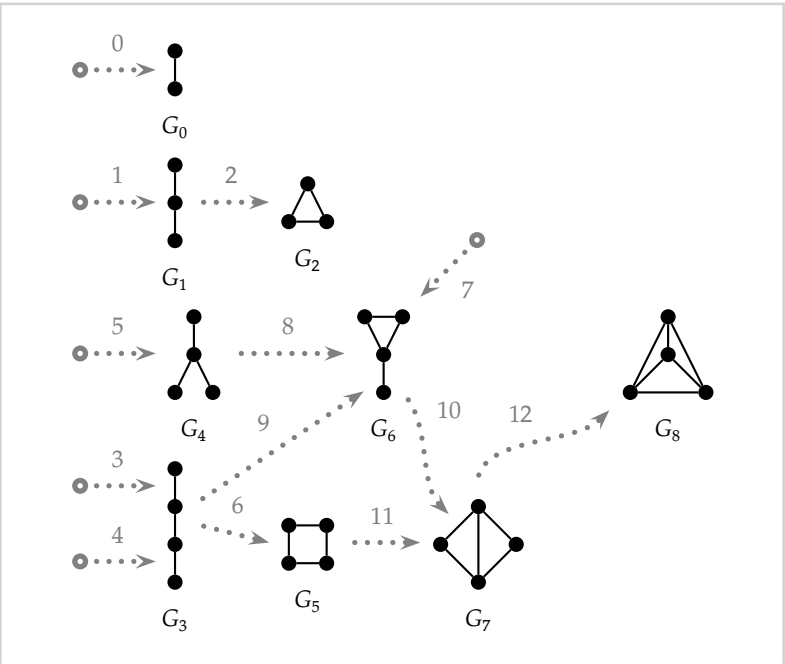
3. $u,v \subseteq S$ in $G[S]$ ni povezan. Na točkah S torej v grafu G ni bil induciran noben grafek, iz definicije množice S pa sledi, da $\exists i : G'[S] \cong G_i$. □

Trditev 3 (Informativnost orbit): Za vsak povezan grafek v $G' = G + uv$, induciran na množici točk $S \subseteq V(G') = V(G)$, lahko ugotovimo, kateri grafek je induciran na istih točkah v G, brez eksplicitnega iskanja izomorfizma. Dovolj je, da poznamo orbito povezave uv, ki jo pri grafkih velikosti do 5 vozlišč lahko razberemo tudi iz orbit njenih krajišč.

Dokaz: Če $\{u,v\} \nsubseteq S$, potem $uv \notin G'[S]$ in je očitno $G[S] = G[S']$. V nasprotnem primeru si lahko za grafke do neke fiksne velikosti predpripravimo preslikavo $\mathrm{Orb}_e^{G'[S]}(uv) \mapsto G_j$, kjer je $G_i \cong G'[S]$ in $G_j \cong G[S]$. Pri tem je lahko tudi $G_j = \perp$ (nedefiniran), če G_i po brisanju povezave iz te orbite razpade v več nepovezanih komponent. Preslikava je res enolično določena z orbito, saj iz nje lahko trivialno razberemo G_i in nato najdemo grafek G_j z brisanjem poljubne povezave xy iz te orbite. Če bi za brisanje izbrali drugo povezavo zw iz iste orbite, bi dobili isti rezultat, saj obstaja avtomorfizem, ki slika $zw \mapsto xy$. □

Dogodki, ki jih želimo opazovati v rastočem omrežju, so nastanki novih grafkov s povezovanjem vozlišč v določenih orbitah, oziroma evolucija grafkov v druge, gostejše grafke. Povedano drugače, zanimajo nas pojavitve povezav v posameznih orbitah skozi čas, saj:

- če se pojavi povezava v orbiti 0, vemo, da sta se povezali dve vozlišči,
- če se na primer pojavi povezava v orbiti 5, vemo, da je nastal nov grafek (na množici točk, ki prej ni inducirala povezanega grafka) G_4,
- če se pojavi povezava v orbiti 8, vemo, da se je grafek G_4 razvil v G_6 itd.



Slika 4.6: Možne evolucije grafkov velikosti do 4 vozlišč. Na puščicah, ki predstavljajo nastanke grafkov, so zapisane orbite nazadnje nastalih povezav v njih.

Slika 4.6 prikazuje vse možne razvoje grafkov. Na puščicah, ki predstavljajo nastanke grafkov, so zapisane orbite dodanih povezav. Za grafke s po 5 vozlišči bi bila slika nepregledna, zato so razmerja med njimi predstavljena v tabeli 4.1.

4.2.2 *Prepletenost s sorodnimi raziskavami*

Za nekatere dogodke se zdi, da jih ni smiselno opazovati. Pojavitev povezave v orbiti 30, na primer, pomeni pripenjanje vozlišča k ciklu dolžine 4. Taka sprememba je trivialna, njenega pomena za strukturo omrežja pa ni lahko intuitivno pojasniti, kot bomo v nadaljevanju storili za nekatere druge. Obstajajo pa netrivialne spremembe, ki nimajo intuitivne razlage, zato jih dosedanje raziskave niso zajele, morda pa vseeno nosijo informacijo, ki je uporabna pri neki nalogi s področja analize omrežij.

Pojavitev povezave v orbiti 0 pomeni zgolj in samo dodajanje nove povezave. Vsaka povezava uv v omrežju G nastopa v tej orbiti natanko enkrat, tj. v induciranem

Tabela 4.1: Možne evolucije grafkov s po 5 vozlišči.

$\mathrm{Orb}_e^{G+e}(e)$	$G \to G+e$	$\mathrm{Orb}_e^{G+e}(e)$	$G \to G+e$	$\mathrm{Orb}_e^{G+e}(e)$	$G \to G+e$
13	$\perp \to G_9$	32	$G_9 \to G_{16}$	51	$G_{17} \to G_{23}$
14	$\perp \to G_9$	33	$\perp \to G_{17}$	52	$G_{19} \to G_{23}$
15	$\perp \to G_{10}$	34	$G_{12} \to G_{17}$	53	$G_{17} \to G_{24}$
16	$\perp \to G_{10}$	35	$G_{14} \to G_{17}$	54	$G_{19} \to G_{24}$
17	$\perp \to G_{10}$	36	$G_{16} \to G_{17}$	55	$G_{18} \to G_{24}$
18	$\perp \to G_{11}$	37	$G_{13} \to G_{18}$	56	$G_{21} \to G_{24}$
19	$\perp \to G_{12}$	38	$G_{14} \to G_{18}$	57	$G_{19} \to G_{25}$
20	$G_{10} \to G_{12}$	39	$\perp \to G_{19}$	58	$G_{21} \to G_{25}$
21	$G_9 \to G_{12}$	40	$G_{13} \to G_{19}$	59	$G_{20} \to G_{25}$
22	$\perp \to G_{13}$	41	$G_{12} \to G_{19}$	60	$G_{23} \to G_{26}$
23	$\perp \to G_{13}$	42	$G_{16} \to G_{19}$	61	$G_{22} \to G_{26}$
24	$G_{10} \to G_{13}$	43	$G_{16} \to G_{20}$	62	$G_{24} \to G_{26}$
25	$G_9 \to G_{13}$	44	$G_{12} \to G_{21}$	63	$G_{25} \to G_{26}$
26	$\perp \to G_{14}$	45	$G_{13} \to G_{21}$	64	$G_{24} \to G_{27}$
27	$G_{11} \to G_{14}$	46	$G_{16} \to G_{21}$	65	$G_{25} \to G_{27}$
28	$G_{10} \to G_{14}$	47	$G_{15} \to G_{21}$	66	$G_{26} \to G_{28}$
29	$G_9 \to G_{15}$	48	$G_{17} \to G_{22}$	67	$G_{27} \to G_{28}$
30	$\perp \to G_{16}$	49	$G_{20} \to G_{22}$	68	$G_{28} \to G_{29}$
31	$G_{10} \to G_{16}$	50	$\perp \to G_{23}$		

podgrafu $G\{u, v\}$.

Prva grafkovna stopnja povezave je enaka številu povezav, s katerimi si deli eno od krajišč. Količina je za novonastalo povezavo enaka vsoti stopenj njenih krajišč pred njenim nastankom. To nekoliko spominja na verjetnost, da se dve vozlišči povežeta pri modelu prednostne povezanosti (angl. *preferential attachment* [6]), ki je enaka *produktu* njunih stopenj.

Druga grafkovna stopnja povezave pove število trikotnikov, v katerih nastopa, če gre za novonastalo povezavo, pa število trikotnikov, ki jih z nastankom *zapira*. Predvsem v družbenih omrežjih so trikotniki in *tranzitivne povezave*, ki jih zapirajo, pomemben pojav, na katerem temeljijo tudi generatorji omrežij [68] in napovedni modeli [54].

Dodajanje povezave v orbiti 2, 12 ali 68 pomeni nastanek klike s 3, 4 oziroma 5 vozlišči. Klike se uporabljajo pri vizualizaciji metaboličnih omrežij [69], odkrivanju tesno povezanih skupin [70] in podobno.

4.3 Empirična analiza časovnega razvoja

Namen naših raziskav ni bila zgolj teoretična obravnava možnih poti razvoja grafkov v omrežju, temveč tudi analiza konkretnih omrežij, ki se spreminjajo s časom. Rezultate analize smo nato poskusili uporabiti za napovedovanje nadaljnjega časovnega razvoja.

4.3.1 Opazovane količine

Časovno komponento razvoja smo zajeli s količino, ki evolucijo grafkov povezuje s časom: merili smo čas, ki preteče, preden pride do neke točno določene spremembe grafka. Ker smo se osredotočali le na dogodke dodajanja povezav, smo spremembe med seboj ločevali po orbiti novonastale povezave. V prejšnjem razdelku smo vsak dogodek take spremembe enolično povezali z nastankom enega grafka in morebitnim izginotjem drugega; čase, ki jih bomo opazovali, torej znamo interpretirati.

Opozoriti moramo, da vsaka v omrežje dodana povezava povzroči več kot le eno spremembo (razen, če je izolirana). Običajno namreč obstaja več podmnožic treh vozlišč, na katerih nova povezava tvori pot dolžine dve ali trikotnik in podobno velja tudi za večje podmnožice.

Recimo, da opazujemo časovno zaporedje omrežij $G_t = G(V, E_t)$ od časa $t_0 = 0$ naprej. Zaradi poenostavitve zapisa predpostavimo, da že na začetku obstajajo vsa vozlišča, a so izolirana. Čas obravnavamo kot zvezno spremenljivko, a do sprememb (tako

dodajanja kot brisanja povezav) pride le v diskretnih časih $t_1, t_2, \ldots$ Čas zadnje spremembe (dodajanja ali brisanja povezave) med vozliščema u in v pred časom t označimo $c_t(uv)$.

Po dodajanju povezave $e = uv$ v času t_i za vsako povezano podmnožico vozlišč $S, \{u, v\} \subseteq S \subseteq V$ velikosti od 2 do 5 najdemo $\mathrm{Orb}_e^{(G_{t_{i-1}}+e)[S]}(e)$, torej orbito povezave e v omrežju $G_{t_{i-1}} + e$, in čas, ki je pretekel od zadnje spremembe v tem grafku:

$$t_i - c_{t_i}(S) \coloneqq t_i - \max_{\substack{x,y \in S \\ x \neq y}} c_{t_i}(xy). \tag{4.7}$$

Na spremembo grafka v množici S gledamo kot na verjetnostni dogodek C_o (kjer $o = \mathrm{Orb}_e^{(G_{t_{i-1}}+e)[S]}(e)$), tj. dogodek pojavitve nove povezave v orbiti o, na vrednost izraza 4.7 pa kot na vrednost slučajne spremenljivke T_o. Opazovanje razvoja nekega omrežja nam postreže z vzorcem vrednosti za vsako spremenljivko $T_o, o = 1, \ldots, 68$, iz katerega lahko ocenimo porazdelitev vsake od teh spremenljivk. Iz vzorca vrednosti spremenljivke T_2, na primer, lahko razberemo, koliko časa traja *zapiranje trikotnika*, torej sprememba iz grafka G_1 (pot dolžine 2) v G_2 (trikotnik).

Razlog, da opazujemo $G_{t_{i-1}} + e$ in ne G_{t_i}, je v tem, da želimo istočasne spremembe opazovati izolirano drugo od druge. Četudi hkratni dogodki pojavitve več povezav morda niso medsebojno neodvisni, saj so lahko neke okoliščine v omrežju povod za več dogodkov, pa predpostavljamo, da so *neodvisni pogojno na stanje omrežja v prejšnjem koraku*: če A_e označuje dogodek, da se v času t_i pojavi povezava e, za povezavi e in f velja:

$$P\left(A_e \,\middle|\, A_f \cap G_{i-1}\right) = P\left(A_e \mid G_{i-1}\right) \tag{4.8}$$

Slučajne spremenljivke T_0 ne opazujemo, saj pogosto ni definirana: ob prvi pojavitvi povezave (torej edinega grafka na 2 vozliščih) ne moremo smiselno definirati časa zadnje spremembe grafka.

4.3.2 Podatki

Tako definirane slučajne spremenljivke T_o smo opazovali na srednje velikih sintetičnih in realnih omrežjih. Vseh 8 omrežij smo objavili na spletnem mestu `http://www.biolab.si/supp/mpolaj`.

Sintetična omrežja

Vsako sintetično omrežje smo zgradili tako, da smo najprej zgenerirali končno stanje omrežja $G_n = G(V, E_n)$ z n vozlišči z algoritmom za generiranje grafov. Časovni razvoj omrežja smo definirali rekurzivno:

$$G_{t-1} = G(V, E_t \setminus e) \text{ za enakomerno naključno izbran } e \in E_t = E(G_t). \tag{4.9}$$

Uporabili smo tri različne algoritme za generiranje grafov: *Erdős–Rényijev model* [34], *Barabási-Albertov model* [6] in geometrijski model [71].

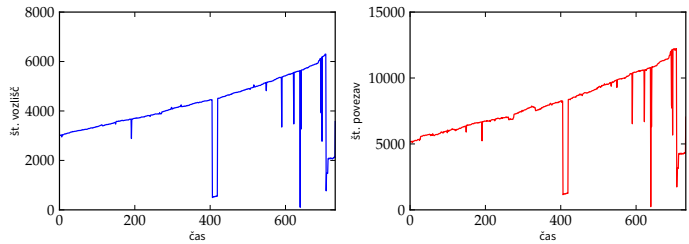
Prvi je najzgodnejši model za generiranje naključnih omrežij, za katerega se je izkazalo, da slabo posnema lastnosti omrežij iz realnih podatkov. Model je preprost: v omrežju z n vozlišči med vsak par vozlišč dodamo povezavo z verjetnostjo p neodvisno od drugih parov. Namesto izbire parametra p lahko fiksiramo število povezav na e. Nato uporabimo prvih e povezav iz enakomerno naključne permutacije povezav. Tako generirana omrežja bomo označili z $G_{\text{ER}}(n, e)$.

Barabási-Albertov model sestavi naključna *brezlestvična* (angl. *scale-free*) omrežja, tj. omrežja s potenčno porazdelitvijo stopenj vozlišč. Ta lastnost je bila namreč opažena pri velikih realnih omrežjih. Algoritem začne s povezanim grafom (npr. potjo) na m_0 vozliščih, nato pa dodaja vozlišča vse dokler jih omrežje ne vsebuje n. Vsako vozlišče je, ko je dodano, povezano z $m \leqslant m_0$ naključno izbranimi dotlej obstoječimi ciljnimi vozlišči. Verjetnost izbire vsakega vozlišča za ciljno je premo sorazmerna z njegovo takratno stopnjo; to imenujemo mehanizem *prednostne povezanosti* (angl. *preferential attachment*) ali tudi *bogati bogatijo* (angl. *the rich get richer*). Omrežja, generirana s tem modelom, bomo označevali z $G_{\text{BA}}(n, m)$ in privzeli $m_0 = m$.

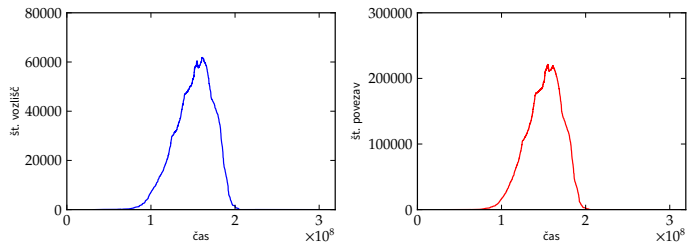
Geometrijski model interpretira točke v geometrijskem prostoru kot vozlišča omrežja, ki so povezana, če in samo če so si pri izbrani metriki dovolj blizu. Pogosto je uporabljen kar d-dimenzionalen Evklidski prostor z Evklidsko (ℓ^2) normo in koordinatami, omejenimi na interval $[0, 1)$. Algoritem torej generira n enakomerno naključnih točk v prostoru, ki ustrezajo vozliščem omrežja. Parameter, ki omejuje razdaljo povezanih vozlišč, lahko nastavimo tudi po generiranju točk in s tem poskrbimo, da bo v omrežju natanko e povezav. Take grafe bomo označevali z $G_{\text{G}}(n, m, d)$.

Realna omrežja

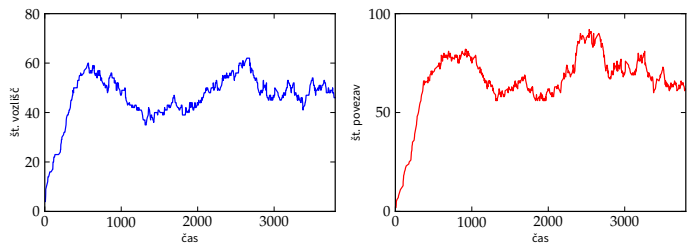
V analizo smo zajeli tudi 4 realna omrežja: *AS*, *Enron*, *India* in *Levant*. Trendi števil vozlišč (stopnje vsaj 1) in povezav skozi čas so za vsa omrežja prikazani na sliki 4.7.



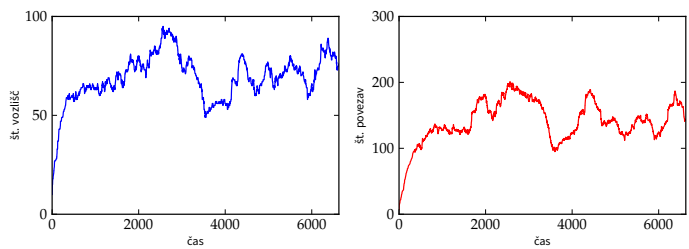
(a) AS



(b) Enron



(c) India



(d) Levant

Slika 4.7: Število vozlišč (levo) in število povezav (desno) v posamezni seriji omrežij skozi čas.

AS (*Autonomous Systems*) je serija 733 omrežij internetnih usmerjevalnikov [8]. Povezave opisujejo sosednost usmerjevalnikov v smislu neposredne medsebojne komunikacije za obdobje dobrih dveh let (785 dni) med koncem leta 1997 in začetkom leta 2000. Časovna enota je dan.

Enron je zbirka elektronskih sporočil, ki so jih poslali ali prejeli zaposleni v istoimenskem propadlem Teksaškem energetskem podjetju, poznanem po škandalu ob odkritju večletnega prirejanja poslovnih izidov. Zaporedje omrežij smo sestavili tako, da vozlišča predstavljajo elektronske naslove, povezave med vozlišči pa so dodane v trenutku prvega izmenjanega sporočila. Po enem letu brez izmenjanega sporočila povezava izgine in se kasneje lahko zopet pojavi. Časovna enota je sekunda.

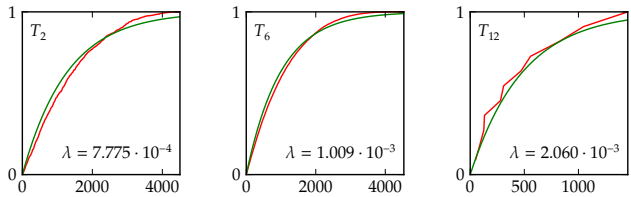
Omrežji *India* in *Levant* smo zgradili iz dnevnika dogodkov v odnosih med državami. Kot povesta imeni zbirk, prva vsebuje dogodke, povezane z Indijo, druga pa z državami tako imenovanega bližnjega vzhoda [72]. V dnevniku so zabeleženi dogodki različnih vrst (več kot 100 izidov diplomatskih srečanj, uradnih sporočil za javnost in dejanj); mi smo se omejili na dogodke uporabe vojaške sile (predpona 22 po klasifikaciji WEIS). Vozlišča predstavljajo države, pomembne osebe (predsednike, teroriste, …), regije, etnične skupine ipd., povezave pa se med njimi pojavijo v trenutku, ko skupaj nastopijo v zabeleženem dogodku opazovane vrste. Podobno kot pri zbirki *Enron* smo povezave po enem letu od zadnje interakcije odstranili in po potrebi ponovno dodali. Časovna enota je dan.

4.3.3 Rezultati

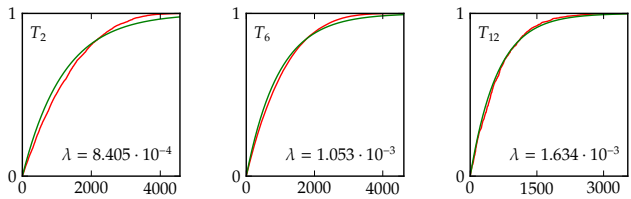
Analizirali smo opisana realna omrežja ter sintetična omrežja z naslednjimi parametri:

- $G_{\mathrm{ER}}(500, 5000)$,
- $G_{\mathrm{BA}}(500, 10)$,
- $G_{\mathrm{G}}(500, 5000, 2)$ in
- $G_{\mathrm{G}}(500, 5000, 4)$.

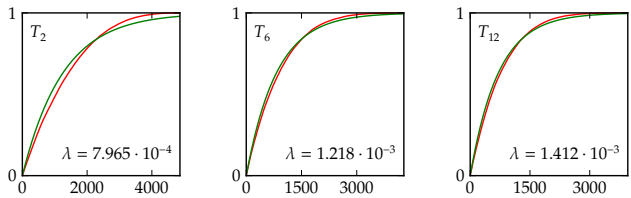
Opazovali smo vrednosti slučajnih spremenljivk T_o, kot smo jih definirali z izrazom 4.7. Vseh spremenljivk je 68 za vsako omrežje, zato na slikah 4.8 in 4.9 prikazujemo kumulativne porazdelitve le za izbrane.



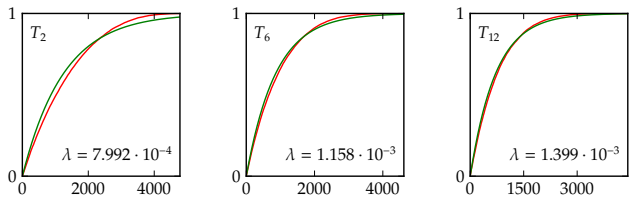
(a) $G_{\mathrm{ER}}(500, 5000)$



(b) $G_{\mathrm{BA}}(500, 10)$

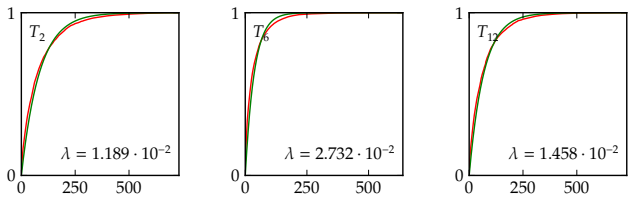


(c) $G_{\mathrm{G}}(500, 5000, 2)$

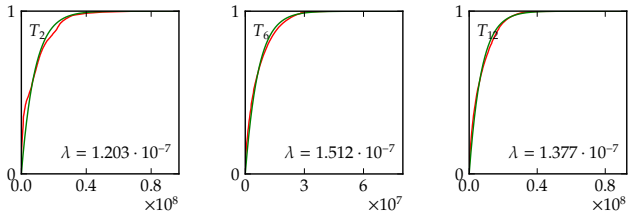


(d) $G_{\mathrm{G}}(500, 5000, 4)$

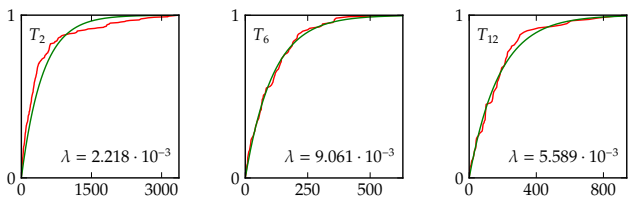
Slika 4.8: Porazdelitve slučajnih spremenljivk T_o (rdeče) in ocena eksponentne porazdelitve po metodi največjega verjetja (zeleno) za sintetična omrežja.



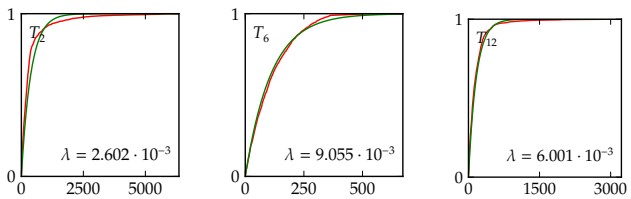
(a) *AS*



(b) *Enron*



(c) *India*



(d) *Levant*

Slika 4.9: Porazdelitve slučajnih spremenljivk T_o (rdeče) in ocena eksponentne porazdelitve po metodi največjega verjetja (zeleno) za realna omrežja.

Opazimo lahko, da so pri vseh omrežjih, tudi pri različnih sintetičnih modelih, porazdelitve slučajnih spremenljivk T_o približno eksponentne, zato bomo odslej predpostavljali $T_o \sim \mathrm{Exp}(\lambda_o)$. Ob vsakem izrisu porazdelitve je izpisan pripadajoč parameter $\lambda_o = \frac{1}{\mathrm{E}[T_o]}$, ocenjen iz podatkov z uporabo metode največjega verjetja.

Slika 4.10 za vsa omrežja grafično prikazuje vrednosti $E[T_o] = \frac{1}{\lambda_o}$ vseh orbit povezav v grafkih velikosti do 4 vozlišč. Številke nad podatki predstavljajo zaporedne številke orbit (o).

Interpretacija rezultatov

$E[T_1]$ je pričakovani čas, ki v določenem omrežju preteče od pojavitve povezave uv do pojavitve povezave vw, ko u in w (še) nista povezani. $E[T_2]$ je pričakovani čas zapiranja trikotnikov, torej čas med pojavitvijo druge in tretje, zadnje, povezave med določenimi tremi točkami. (Pri interpretaciji pomena indeksov slučajnih spremenljivk si pomagamo s sliko 4.4 ali 4.6.)

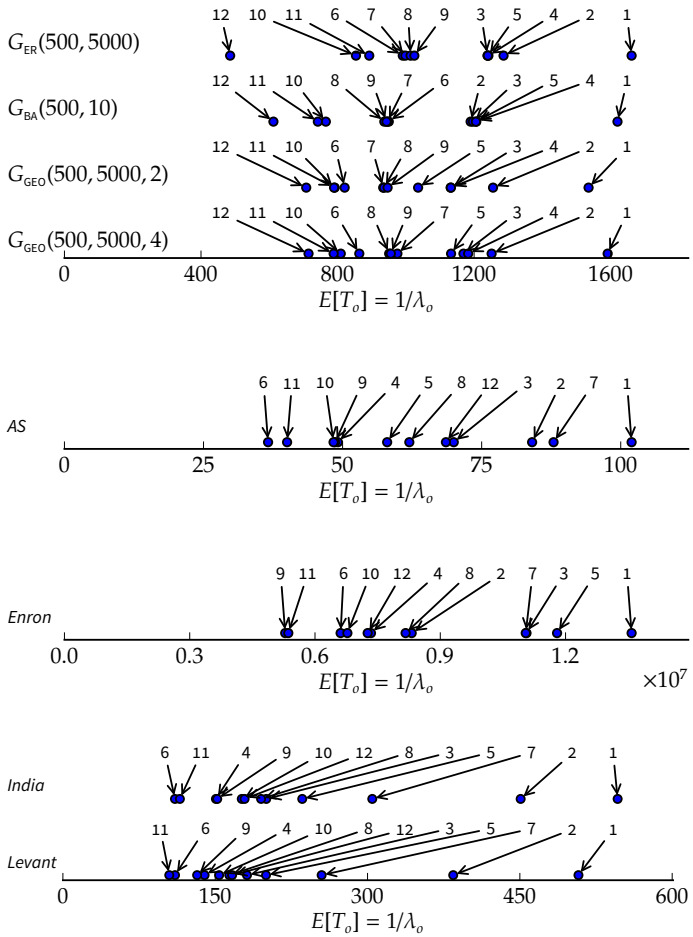
V prav vseh opazovanih omrežjih v povprečju mine manj časa do zapiranja trikotnika kot do pojavitve druge povezave v trojici točk. Vzroke za to gre iskati tudi v številu povezav v grafku pred dogodkom. Tudi pri popolnoma naključnem vrstnem redu pojavitve povezav, ki smo mu priča pri uporabljenih naključno zgrajenih omrežjih, je namreč *čas najkasneje dodane povezave* (odštevanec v izrazu 4.7) pričakovano večji pri večjem številu povezav. To sledi iz neenačbe

$$\begin{aligned} P\left[\max\{C_1, C_2\} \geqslant c\right] &= 1 - P\left[C_1 < c, C_2 < c\right] = \\ &= 1 - P\left[C_1 < c\right] \cdot P\left[C_2 < c \mid C_1 < c\right] \geqslant \\ &\geqslant 1 - P\left[C_1 < c\right] = P\left[C_1 \geqslant c\right], \end{aligned} \tag{4.10}$$

kjer smo s C_1 in C_2 označili časa pojavitve dveh povezav.

Vrednosti $E[T_o]$ torej niso primerljive med tistimi orbitami povezav, ki nastopajo v grafkih z različnim številom povezav. Lahko pa primerjamo vrednosti za orbiti 2 in 3 (sprememba poti dolžine 2 v trikotnik ali pot dolžine 3), za orbite od 6 do 9 ter za orbiti 10 in 11.

Na sliki 4.10 vidimo, da je pri naključnem vrstnem redu dodajanja povezav v naključnih omrežjih pričakovani čas do zapiranja trikotnika enak ali nekoliko *daljši* kot čas do dodajanja tretje povezave poti. V realnih omrežjih je razlika med dogodkoma večja. V edinem družbenem omrežju, *Enron*, se mnogo hitreje zapirajo trikotniki, pri



Slika 4.10: Pričakovane vrednosti T_o ($E[T_o] = 1/\lambda_o$) po posameznih omrežjih. Številke nad podatki predstavljajo zaporedne številke orbit (o).

ostalih treh realnih omrežjih pa se hitreje daljšajo poti. Slednje se zdi razumljivo, saj trikotnik v omrežju povezljivosti internetnih usmerjevalnikov ni zelo uporabna struktura (le obstoječo pot dolžine 2 skrajša za eno povezavo), povezave v omrežjih *India* in *Levant* pa simbolizirajo negativne odnose in interpretacija trikotnika (sovražnik mojega sovražnika je moj sovražnik) nasprotuje intuiciji in rezultatom raziskav v omrežjih s predznačenimi (torej pozitivnimi in negativnimi) povezavami [73].

Podobno lahko interpretiramo razlike v pričakovanih časih pojavitev povezav v orbitah od 6 do 9. V naključnih omrežjih so vrednosti za vse štiri orbite praktično enake, le geometrična omrežja v krajšem času tvorijo štirikotnike (grafek G_5) kot pa trikotnike z repom (grafek G_6). V realnih omrežjih so rezultati precej drugačni. Zopet izstopa *Enron*, ki iz poti dolžine 3 (grafek G_3) prej tvori trikotnike (nova povezava v orbiti 9) kot kvadrate (orbita 6). Od vseh štirih dogodkov se najpočasneje zgodi dodajanje repa trikotniku (orbita 7).

Razumljiv se zdi tudi rezultat, da v realnih omrežjih prej nastaneta dva trikotnika iz kvadrata (z dodajanjem povezave v orbiti 11) kot pa dodajanje še enega trikotnika v trikotnik z repom (z dodajanjem povezave v orbiti 7 v grafek G_6).

4.4 Napovedovanje nadaljnjega razvoja

Tako informacijo o časovni porazdelitvi lokalnega razvoja lahko v principu izkoristimo za napovedovanje verjetnosti določenih sprememb v omrežju skozi čas. Učni del postopka je podan s prejšnjim razdelkom: omrežje opazujemo do nekega trenutka $G_{t_L} = \left(V, E_{t_L}\right)$, na primer sedanjosti, in za videne podatke izračunamo količine $\lambda_1, \lambda_2, \ldots, \lambda_{68}$ pri predpostavki $T_o \sim \mathrm{Exp}(\lambda_o)$. To predstavlja model, torej približek opisa mehanizma, ki naj bi generiral videne podatke.

Napovedni del postopka na vhodu dobi model $M = (\lambda_1, \lambda_2, \ldots, \lambda_{68})$, stanje omrežja G v nekem trenutku t_T, nepovezan par vozlišč $\{u, v\}$, za katerega nas zanima verjetnost pojavitve povezave, ter časovno obdobje Δt, za katerega nas omenjena verjetnost zanima. Pri predpostavljenem modelu M želimo torej izračunati

$$P\left[uv \in E_{t_T+\Delta t} \,\middle|\, uv \notin E_{t_T}\right]. \tag{4.11}$$

Naj bo $\mathscr{S} = \left\{S \subseteq V \,\middle|\, |S| \leqslant 5, \{u, v\} \subset S, \text{graf}\,(G_{t_T} + uv)[S] \text{ je povezan}\right\}$ družina podmnožic vozlišč omrežja G, ki tvorijo grafke okrog povezave uv v omrežju $G_{t_T} + uv$. Označimo orbito povezave uv v grafku $(G_{t_T} + uv)[S]$ z o_S.

Preprosta napovedna metoda se ponuja z neposredno uporabo modela M. Najprej preštejemo pojavitve povezave uv, če bi ta v grafu G_{t_T} obstajala, v posameznih orbitah. S tem dobimo vektor $\eta(uv) \coloneqq \text{edge-}GDV(uv) = \left[\left|\left\{S \in \mathscr{S} \;\middle|\; o_S = o\right\}\right|\right]_o$. Označimo z A_S dogodek iz realnega sveta, ki potencialno sproži pojavitev povezave uv v grafku $(G_{t_T} + uv)[S]$. Pri tem se sklicujemo na nek stohastičen mehanizem v naravnem sistemu, opisovanem z od časa odvisnim omrežjem G, ki proži morebitno povezovanje vozlišč v takem grafku. Povedano drugače, dogodek A_S je v nekakšni pozitivni korelaciji z dogodkom pojavitve povezave uv v grafku na vozliščih S, a ni nujno, da en dogodek implicira drugega. Verjetnost, da se v časovnem oknu dolžine Δt zgodi dogodek A_S ne znamo oceniti, zato predpostavimo, da je verjetnost kar enaka ocenjeni verjetnosti dogodka, da se v tem času pojavi povezava uv v grafku na vozliščih S:

$$P[A_S] = P\left[T_{o_S} \leqslant \Delta t\right] = 1 - e^{-\lambda_{o_S}\Delta t}. \tag{4.12}$$

Ena izmed možnih ekstremnih predpostavk v takem sistemu je, da se povezava pojavi le, če se v naravnem sistemu zgodijo *vsi* dogodki A_S za $S \in \mathscr{S}$. Do pojavitve torej pride natanko takrat, ko se pojavijo vse poti dolžine 2, v katerih nastopa ta povezava (takih poti je $\eta(uv)_1$, saj povezava uv v njih nastopa v orbiti 1) in se zaprejo vsi trikotniki, ki jo vsebujejo (takih trikotnikov je $\eta(uv)_2$ itd. Zaradi preprostosti metode zanemarimo odvisnosti med temi dogodki in predpostavimo njihovo pogojno neodvisnost, zato je verjetnost, da se zgodijo vsi, enaka produktu verjetnosti posameznih dogodkov. Potem vrednost izraza 4.11 izračunamo kot

$$P_{\text{all}}\left[uv \in E_{t_T+\Delta t} \;\middle|\; uv \notin E_{t_T}\right] = P\left[\bigwedge_{S\in\mathscr{S}} A_S\right] = \tag{4.13}$$

$$= \prod_{S\in\mathscr{S}} P[A_S] = \prod_{o=1}^{68}(1 - e^{-\lambda_o\Delta t})^{\eta(uv)_o}$$

in s tem dobimo sorazmerno preprosto napovedno metodo.

V učnem delu postopka smo vrednosti slučajne spremenljivke T_o definirali kot čas, ki je do pojavitve povezave v orbiti o v podmnožici točk S pretekel od zadnje spremembe med katerimakoli dvema vozliščema v S (izraz 4.7). Napovedna metoda ne upošteva časa, ki je minil od zadnje spremembe v podmnožici točk S ($c_{t_T}(S)$) do časa začetka

napovedovanja (t_T), in torej predpostavlja

$$P\left[uv \in E_{t_T+\Delta t} \;\middle|\; uv \notin E_{t_T}\right] = P\left[uv \in E_{t_T+\Delta t+\left(t_T-c_{t_T}(S)\right)} \;\middle|\; uv \notin E_{t_T}\right] \tag{4.14}$$

oziroma ekvivalentno

$$P\left[T_o \leqslant \Delta t\right] = P\left[T_o \leqslant \Delta t + \left(t_T - c_{t_T}(S)\right) \;\middle|\; T_o > t_T - c_{t_T}(S)\right]. \tag{4.15}$$

To je poseben primer enačbe

$$P\left[X \leqslant x\right] = P\left[X \leqslant x + x_0 \;\middle|\; X > x_0\right]. \tag{4.16}$$

Porazdelitvam, za katere velja ta enačba, pravimo, da so *brez spomina* (angl. *memoryless*). Eksponentna porazdelitev ima to lastnost, zato je predpostavka iz enačbe 4.14 izpolnjena.

Izraz 4.14 je en od ekstremnih pogledov na povode za spremembo v omrežju. Čeprav je dejstvo, da se pri eni dodani povezavi zgodi mnogo ($|\mathscr{S}|$) dogodkov na nivoju grafkov, pa si lahko mislimo, da so zgolj nekateri od teh povod za dodajanje povezave, ostali pa so kolateralni. To pomeni, da smo pri zanemarjanju odvisnosti med dogodki storili hudo napako: čeprav bo pod določenimi pogoji do kolateralnih dogodkov zagotovo prišlo, pa je njihova brezpogojna verjetnost lahko nizka. V gostem delu omrežja je takih dogodkov veliko in ko pod predpostavko neodvisnosti zmnožimo njihove verjetnosti, pridemo do zelo nizke vrednosti. To nasprotuje že omenjenemu principu *prednostne povezanosti*, ki pravi, da bi ravno v gostejšem delu omrežja morala biti verjetnost za nastanek nove povezave večja.

Nasprotna ekstremna interpretacija sistema je, da se povezava pojavi, ko se zgodi *katerikoli* dogodek (en ali več) izmed $A_S, S \in \mathscr{S}$. Vse ostale spremembe grafkov so morda kolateralne. Verjetnost pojavitve povezave izrazimo kot od 1 odšteto verjetnost, da se ne zgodi noben dogodek, ki implicira pojavitev povezave:

$$P_1\left[uv \in E_{t_T+\Delta t} \;\middle|\; uv \notin E_{t_T}\right] = 1 - \bigwedge_{S\in\mathscr{S}} \overline{A_S} = \tag{4.17}$$

$$= \quad 1 - \prod_{S\in\mathscr{S}} \left(1 - P\left[A_S\right]\right) = 1 - \prod_{o=1}^{68} \left(1 - \left(1 - e^{-\lambda_o \Delta t}\right)\right)^{\eta(uv)_o} =$$

$$= \quad 1 - \prod_{o=1}^{68} e^{-\lambda_o \Delta t \cdot \eta(uv)_o} = 1 - e^{-\Delta t \cdot \sum_{o=1}^{68} \lambda_o \eta(uv)_o}$$

V tem primeru smo predpostavili pogojno neodvisnost nasprotnih dogodkov sprememb v grafkih okrog povezave uv. Očitna posledica so višje vrednosti v gostejših delih grafa, kar ustreza principu *prednostne povezanosti*.

Izraza P_{all} in P_1 zastopata ekstremne interpretacije: bodisi je za pojavitev povezave potrebno, da se zgodijo vsi dogodki $A_S, S \in \mathscr{S}$, bodisi je dovolj zgolj en. Drugo možnost je mogoče posplošiti v zahtevo, da se zgodi najmanj neko fiksno število (k) dogodkov in tako definirati verjetnost P_k pojavitve nove povezave.

$$
\begin{aligned}
& P_k\left[uv \in E_{t_T+\Delta t} \;\middle|\; uv \notin E_{t_T}\right] = \qquad (4.18)\\
= \; & P\left[\bigvee_{\substack{\mathscr{S}_+ \subseteq \mathscr{S} \\ |\mathscr{S}_+| \geqslant k}} \left(\bigwedge_{S \in \mathscr{S}_+} A_S \wedge \bigwedge_{S \in \mathscr{S} \setminus \mathscr{S}_+} \overline{A_S}\right)\right] = \\
= \; & \sum_{\substack{\mathscr{S}_+ \subseteq \mathscr{S} \\ |\mathscr{S}_+| \geqslant k}} \left(\prod_{S \in \mathscr{S}_+} P\left[A_S\right] \cdot \prod_{S \in \mathscr{S} \setminus \mathscr{S}_+} P\left[\overline{A_S}\right]\right)
\end{aligned}
$$

Izidi v disjunkciji so disjunktni (tj. se ne morejo zgoditi hkrati), zato je njihova verjetnost kar vsota verjetnosti posameznih izidov. Vseh členov v vsoti je toliko, kot je podmnožic množice $\mathscr{S}$ z vsaj k elementi, kar z velikostjo $\mathscr{S}$ raste eksponentno hitro. Neposreden izračun verjetnosti s podanim izrazom torej ni možen zaradi velike časovne zahtevnosti.

Ker opazujemo $N = |\mathscr{S}|$ po predpostavki neodvisnih dogodkov, pri čemer i-ti dogodek uspe z verjetnostjo $p_i = P\left[A_{S_i}\right]$, rečemo, da izvajamo zaporedje N *Poissonovih poskusov*[1]. Slučajna spremenljivka K, katere vrednost je enaka številu uspelih poskusov, je porazdeljena po *Poissonovi binomski porazdelitvi*. Izraz 4.18 je enak $P[K \geqslant k] = 1 - P[K \leqslant k-1]$. Za izračun njegove vrednosti torej potrebujemo vrednost kumulativne porazdelitve slučajne spremenljivke K v točki $k-1$.

Chen, Dempster in Liu [74] so razvili postopek za izračun gostote verjetnosti v konkretni točki, torej $P[K = k]$, s časovno zahtevnostjo $\mathscr{O}(Nk + k^2)$. Vmesni rezultati pri tem izračunu vsebujejo tudi $P[K = k']$ za vse $0 \leqslant k' < k$, v asimptotično enakem času

[1] Poissonov poskus in Bernoullijev poskus sta izraza, ki označujeta poskus z dvema izidoma (uspeh in neuspeh), od katerih se prvi zgodi z verjetnostjo p. Pri zaporedju takih pogojno neodvisnih poskusov govorimo o Bernoullijevih poskusih, če je verjetnost za uspeh pri vseh poskusih enaka, sicer pa o Poissonovih poskusih.

lahko izračunamo tudi vrednost kumulativne porazdelitve slučajne spremenljivke v tej točki. Očitno je tudi, da pri izračunu P_k nimamo opravka s čisto splošnim zaporedjem Poissonovih poskusov, saj se kljub velikemu številu (N) poskusov lahko pojavi zgolj do 68 različnih verjetnosti uspeha le-teh: $P[A_S]$ je odvisen od λ_o (teh je 68) in Δt (ta je za en izračun fiksno izbran). Brez dokaza navajamo enostavno preverljivo dejstvo, da ima v takem primeru Chen-Dempster-Liujev postopek časovno zahtevnost $\mathscr{O}(k^2)$, če predpostavimo, da se operacija potenciranja s celoštevilskim eksponentom od 1 do N izvede v konstantnem času.

Z izrazi 4.14, 4.17 in 4.18 smo definirali družino napovednih funkcij, ki na podlagi orbit, v katerih bi trenutno neobstoječa povezava nastopala, če bi se pojavila, oceni verjetnost dejanske pojavitve povezave v vnaprej določenem časovnem obdobju Δt. Prva dva izraza sta le posebna primera tretjega: vrednost P_1 je enaka ne glede na to, po katerem od izrazov jo izračunamo, P_{all} pa je enak $P_N = P_{|\mathscr{S}|}$.

Če je cilj uporabe napovedne metode zgolj rangiranje parov vozlišč po naraščajoči verjetnosti pojavitve povezave, je pri P_1 vrednost Δt nepomembna. Ocena verjetnosti nastanka povezave med vozliščema u in v je namreč enaka vrednosti kumulativne porazdelitveni funkciji eksponentne porazdelitve s parametrom $\lambda_{uv} \coloneqq \sum_{o=1}^{68} \lambda_o \eta(uv)_o$ v točki Δt. Ta vrednost (pri poljubnem fiksnem $\Delta t > 0$) monotono narašča z vrednostjo parametra λ_{uv}, torej je rangiranje parov vozlišč odvisno le od pripadajočih vrednosti tega parametra. Za funkcije $P_k, k > 1$ in P_{all} to v splošnem ne velja in je vrednost Δt pomembna.

4.4.1 *Ocena uspešnosti*

Uspešnost predlagane družine napovednih funkcij P_k smo ocenjevali na časovnih zaporedjih omrežij, ki smo jih opisali v razdelku 4.3.2. Na vsaki podatkovni množici smo iz seznama dogodkov (pojavitev in izginotij povezav) izbrali testne intervale tako, da smo zadnjih 75 % seznama enakomerno (po številu dogodkov) razbili na d delov. Pri večini množic smo uporabili $d = 10$, znižali smo ga le pri zelo majhnem omrežju *India*, saj so sicer intervali vsebovali prenizko število testnih povezav. Nato smo izvedli oceno uspešnosti posamezne napovedne funkcije na določenem omrežju s sledečim postopkom.

- Učni del metode smo sprožili na prvih 25 % seznama dogodkov, ki ni bil razdeljen na testne intervale.

- Iz prvega testnega intervala smo izbrali $\frac{f}{2}$ *pozitivnih testnih povezav*, tj. povezav, ki se v tem intervalu pojavijo, ter prav toliko *negativnih testnih povezav*, tj. parov vozlišč, ki v tem intervalu ostanejo nepovezana.

- Z ocenjevano napovedno funkcijo smo ocenili verjetnosti pojavitve pozitivnih in negativnih povezav ter izračunali delež narobe razvrščenih parov, tj. parov ene pozitivne in ene negativne povezave, kjer je bila verjetnost pojavitve negativne povezave ocenjena višje kot verjetnost pojavitve pozitivne povezave. S tem smo dobili oceno za vrednost mere AUC (ploščine pod krivuljo ROC).

- Uporabljen testni interval smo dodali učnim podatkom in postopek ponovili z naslednjim testnim intervalom dogodkov, dokler niso bili uporabljeni vsi razpoložljivi podatki.

Izjema je omrežje *Enron*, pri katerem smo na 10 testnih intervalov razdelili drugih 25 % dogodkov, zadnjih 50 % dogodkov pa smo ignorirali. Večina teh je namreč odstranjevanje povezav, zato omrežje za napovedovanje novih povezav v tem časovnem obdobju ni zanimivo.

Zaradi časovne zahtevnosti postopka smo pri omrežjih *AS* in *Enron* za model domene uporabili le grafke velikosti do 4 vozlišč. Tudi pri ostalih omrežjih smo poskusili uporabiti tak model in rezultati se niso bistveno razlikovali od teh pri uporabi vseh grafkov. Vseeno v teh primerih objavljamo rezultate pri uporabi grafkov s 5 vozlišči.

Ocenili smo uspešnost napovednih funkcij P_1, P_2, P_5, P_{10}, P_{100}, P_{1000} in P_{all}. Vrednost parametra Δt smo nastavili na dolžino intervala, za katerega ocenjujemo verjetnosti pojavitve povezav. To je v teoretičnem smislu edina vrednost, ki napovednim metodam da pravo informacijo o problemu, ki ga z njimi rešujemo. Empirično vpliva tega parametra na uspešnost napovednih metod nismo opazovali.

Kot referenčno metodo za primerjavo uspešnosti smo uporabili sodobno metodo za dopolnjevanje velikih omrežij, metodo *naključnih sprehodov s ponovnimi začetki* (angl. *Random Walk with Restarts, RWR* [48]). Uporabili smo enak postopek ocenjevanja uspešnosti kot pri družini napovednih funkcij, ki jih predlagamo. Delovanje metode RWR smo opisali že v prejšnjem poglavju v razdelku 3.4.3. Za parameter α smo uporabili vrednosti iz množice $\{0; 0{,}01; 0{,}1; 0{,}5; 0{,}9; 0{,}99\}$. Na podatkih *Enron* se izračun pri vrednostih $\alpha = 0$ in $\alpha = 0{,}01$ ni zaključil po 14 dneh neprestanega računanja, zato smo poskus prekinili. Na vseh podatkih smo preizkusili še dodatne vrednosti α

okrog dotedaj najboljšega doseženega AUC, a s tem nismo dosegli izboljšanja rezultata. Konkretno smo uporabili:

- na množicah $G_{\mathrm{ER}}(500, 5000)$ in $G_{\mathrm{BA}}(500, 10)$ vrednosti $\alpha \in \{0{,}001, 0{,}0001\}$,
- na $G_{\mathrm{G}}(500, 5000, 2)$ in $G_{\mathrm{G}}(500, 5000, 4)$ vrednosti $\alpha \in \{0{,}4, 0{,}45, 0{,}55, 0{,}6\}$,
- na AS in Enron vrednosti $\alpha \in \{0{,}85, 0{,}875, 0{,}925, 0{,}95\}$ ter
- na podatkovnih množicah India in Levant vrednosti $\alpha \in \{0{,}999, 0{,}9999\}$.

Rezultate smo poskusili izboljšati še z uporabo meta-učenja, natančneje s skladanjem klasifikatorjev (angl. *stacking*). Verjetnostne napovedi osnovnih klasifikatorjev, tj. v tem delu definiranih napovednih funkcij ter metode naključnih sprehodov s ponovnimi zaključki (pri vseh uporabljenih vrednostih parametrov), za posamezno povezavo smo skupaj z njenimi orbitami uporabili kot vhodne podatke za klasifikator. S tem drugonivojskim klasifikatorjem nismo mogli izdelati napovedi za prvi testni interval, saj še ni bila izdelana nobena verjetnostna napoved za učenje. Pri nadaljnjih testnih intervalih smo za učenje uporabili verjetnostne napovedi osnovnih klasifikatorjev na vseh predhodnih testnih intervalih.

Kot drugonivojski klasifikator smo preizkusili klasifikacijska drevesa, naivni Bayesov klasifikator (z uporabo metode LOESS za ocenjevanje pogojnih porazdelitev zveznih atributov), metodo podpornih vektorjev (epsilon-SVR) ter metodo naključnih dreves (angl. *Random Forest*). Slednja je konsistentno dajala najboljše rezultate, zato bomo v tem delu zaradi preglednosti objavili le rezultate le-te.

Z namenom ugotovitve koristnosti osnovnih klasifikatorjev ter informacije o orbitah povezav smo drugonivojski klasifikator učili in uporabljali na več različnih množicah značilk. Označimo:

- z $\mathscr{F}_P$ označimo značilke z verjetnostnimi napovedmi funkcij P_1, P_2, P_5, P_{10}, P_{100}, P_{1000} in P_{all},
- z $\mathscr{F}_R$ označimo značilke z verjetnostnimi napovedmi metode naključnih sprehodov pri vseh vrednostih α, ki so bile pri posamezni podatkovni množici uporabljene,
- z $\mathscr{F}_O$ označimo značilke s števili pojavitev povezave v orbitah.

Drugonivojski klasifikator smo preizkusili na množicah značilk $\mathcal{F}_P$, $\mathcal{F}_R$, $\mathcal{F}_{PR} = \mathcal{F}_P \cup \mathcal{F}_R$, $\mathcal{F}_{OP} = \mathcal{F}_O \cup \mathcal{F}_P$, $\mathcal{F}_{OR} = \mathcal{F}_O \cup \mathcal{F}_R$ in $\mathcal{F}_{OPR} = \mathcal{F}_O \cup \mathcal{F}_P \cup \mathcal{F}_R$.

V tabeli 4.2 so izpisane ploščine pod ROC krivuljo (AUC) za naštete kombinacije metod in parametrov za vse uporabljene množice podatkov. Zaradi prostorske omejitve širine tabele smo izpustili rezultate za P_2 in P_5, saj so zelo podobnim rezultatom, ki jih dosega P_1.

Opazimo lahko, da P_{all} in P_{1000} dajeta neuporabne rezultate. Vse izračunane verjetnosti pojavitev povezav so namreč tako majhne, da jih pri zapisu v plavajoči vejici pri dvakratni natančnosti ni več možno razločiti. Napovedne funkcije P_k pri preostalih uporabljenih vrednostih k dajejo podobne rezultate, najvišjo uspešnost pa na različnih podatkih dosežejo pri različnih vrednostih parametra.

Pri metodi naključnih sprehodov je za doseganje optimalnih rezultatov potrebno pri različnih podatkovnih množicah uporabiti različne vrednosti parametra α. Za doseganje najvišje možne uspešnosti zgolj z uporabo te metode bi v praksi uporabili notranje prečno preverjanje (angl. *internal cross-validation*) na učni množici. Za korektno neposredno primerjavo parametriziranih napovednih metod je potrebna uporaba takega preverjanja pri ocenjevanju njihove uspešnosti. Mi ga nismo izvedli zaradi visoke časovne potratnosti postopka. Odločitev utemeljujemo s tem, da na nekaterih množicah metoda pri nobeni od uporabljenih vrednosti parametra ne uspe izdelati dobrih napovedi, na drugih pa deluje dobro ne glede na izbrano vrednost parametra (le 0 in vrednosti zelo blizu 1 niso dobre za vsaj katero izmed teh množic).

Pri zadani napovedni nalogi metoda RWR deluje bolje kljub neupoštevanju informacij o preteklem časovnem razvoju. Na vseh množicah, kjer je vsaj ena od metod dosegla omembe vredne rezultate (torej vrednost AUC krepko nad 0,5), je rezultat metode RWR pri večini vrednosti parametra α vseeno boljši od najboljšega rezultata napovednih funkcij P_o, z izjemo množice Enron, kjer sta metodi primerljivi.

Največji uspeh smo dosegli z zlaganjem klasifikatorjev z metodo naključnih gozdov. Pri uporabi vseh razpoložljivih informacij ($\mathcal{F}_{OPR}$) je metoda dosegla zavidljive rezultate na vseh uporabljenih podatkovnih množicah z izjemo sintetičnih podatkov po modelu Erdős–Rényi. Uporaba različnih množic značilk nam daje vpogled v koristnost posameznih informacij. Vidimo, da že zgolj zlaganje napovedi metode naključnih sprehodov daje dobre rezultate. Na nekaterih podatkovnih množicah je koristno dodati še informacijo o orbitah, v katerih nastopajo napovedovane povezave. Uporaba verjetnostnih napovedi v tem poglavju definiranih napovednih funkcij opazno izboljša

Tabela 4.2: Vrednosti AUC.

Metoda	P_k: k = ...				
Podatki	1	10	100	1000	all
$G_{ER}(500, 5000)$	0,511	0,504	0,504	0,500	0,500
$G_{BA}(500, 10)$	0,511	0,511	0,509	0,500	0,500
$G_{G}(500, 5000, 2)$	0,934	0,934	0,901	0,500	0,500
$G_{G}(500, 5000, 4)$	0,851	0,851	0,854	0,500	0,500
AS	0,402	0,402	0,536	0,500	0,500
Enron	0,693	0,693	0,617	0,500	0,500
India	0,392	0,467	0,500	0,500	0,500
Levant	0,463	0,473	0,472	0,500	0,500

	RWR: α = ...					
	0	0,01	0,1	0,5	0,9	0,99
$G_{ER}(500, 5000)$	0,517	0,516	0,515	0,506	0,488	0,500
$G_{BA}(500, 10)$	0,505	0,486	0,483	0,477	0,480	0,500
$G_{G}(500, 5000, 2)$	0,967	0,969	0,978	0,979	0,949	0,500
$G_{G}(500, 5000, 4)$	0,881	0,888	0,918	0,939	0,907	0,500
AS	0,489	0, 641	0, 821	0, 864	0, 890	0, 875
Enron			0,527	0,644	0,692	0,535
India	0,333	0,343	0,366	0,369	0,371	0,446
Levant	0,449	0,471	0,495	0,504	0,540	0,547

	Zlaganje klasifikatorjev z metodo naključnih gozdov					
	$\mathscr{F}_P$	$\mathscr{F}_R$	$\mathscr{F}_{PR}$	$\mathscr{F}_{OP}$	$\mathscr{F}_{OR}$	$\mathscr{F}_{OPR}$
$G_{ER}(500, 5000)$	0,507	0,563	0,590	0,531	0,590	0,585
$G_{BA}(500, 10)$	0,495	0,761	0,699	0,550	0,699	0,687
$G_{G}(500, 5000, 2)$	0,934	0,976	0,983	0,955	0,983	0,984
$G_{G}(500, 5000, 4)$	0,863	0,934	0,955	0,928	0,955	0,956
AS	0,608	0, 986	0, 985	0, 686	0, 985	0, 984
Enron	0,693	0,827	0,833	0,691	0,833	0,842
India	0,525	0,888	0,882	0,674	0,882	0,883
Levant	0,483	0,703	0,741	0,551	0,769	0,810

rezultat le pri množici Levant.

Zastavljen problem je vsaj na nekaterih množicah zelo težak. Tudi uveljavljena metoda za dopolnjevanje omrežij je namreč pogosto dosegla ploščino pod krivuljo, nižjo od 0,5, kar je slabše od naključnega uvrščanja. To pomeni, da bi boljše rezultate dosegli z zamenjavo napovedovanih vrednosti: kot verjetnost, da se neka povezava pojavi, bi upoštevali verjetnost, ki jo metoda napove za dogodek, da se povezava *ne bo* pojavila, in obratno. S tako rešitvijo ne moremo biti zadovoljni, saj pri nobeni od dveh uporabljenih metod napovedovani vrednosti nista arbitrarno določeni, temveč sta metodi zasnovani na teoretičnih temeljih, ki pojasnjujejo, zakaj so vrednosti, ki jih dajeta metodi, ravno verjetnosti, da določene povezave *se* pojavijo. Po drugi strani pa napovedi vseeno imajo praktično vrednost, saj jih z uporabo zlaganja klasifikatorjev lahko koristno uporabimo.

Predlagana družina napovednih funkcij kot napovedni model za časovni razvoj omrežja ne deluje prepričljivo. Po drugi strani pa rezultati zlaganja napovedi metode naključnega sprehoda, po možnosti pri uporabi dodatnih atributov o povezavi, kot so na primer orbite, dajejo zanimivo iztočnico za nadaljnje raziskovalno delo.

5

Zaključek

V tem delu smo postavili temelje novim načinom analize omrežij. Poudarek smo namenili predvsem spremembam v omrežjih na lokalnem nivoju. Temo smo obravnavali z dveh zornih kotov: kot opazovano enoto smo v enem primeru uporabili pogoste vzorce v označenih omrežjih, v drugem pa grafke.

Dela na tem področju nismo izčrpali v širino: časovnega razvoja tretje vrste podomrežij, v razdelku 2.2.1 omenjenih motivov, nismo opazovali. Menimo, da so v tem oziru manj zanimivi od grafkov, saj z dodajanjem nove povezave v motiv ne dobimo nujno spet motiva, torej ni možno opazovati vpliva časa na *razvoj* motivov, temveč zgolj na njihovo pojavljanje in izginjanje. Poleg tega se v različnih omrežjih pojavljajo različni motivi, kar je bilo s pridom uporabljeno za statično primerjavo omrežij, onemogoča pa primerjavo časovnega razvoja različnih omrežij.

Tudi v globino raziskovalnega področja zaradi njegove obsežnosti nismo izčrpali. V zaključku zato poleg povzetka rezultatov objavljamo tudi razpravo o možnostih za nadaljnje delo na tem področju.

5.1 Dopolnjevanje majhnih omrežij

V kontekstu analize omrežij smo odprli novo smer – dopolnjevanje majhnih omrežij. Njeno uporabnost smo utemeljili z opisi praktičnih aplikativnih primerov in v ta namen zajeli konkretne podatke, na katerih smo tudi sami izvajali poskuse. Problem smo formalno definirali in predlagali metodo Hyspan, ki ga rešuje. Slednja temelji na iskanju delnih vpetij vzorcev, ki se v domeni grafov pogosto pojavljajo. Algoritem za reševanje tega podproblema smo zasnovali v okviru razvoja metode. Za odkrivanje pogostih vzorcev smo uporabili enega izmed obstoječih algoritmov, gSpan.

Ker gre za novo področje, smo določili tudi način merjenja uspešnosti metode ter pripadajočo mero uspešnosti. Ciljne vrednosti, ki naj bi jih metoda dosegla, se navadno postavljajo primerjalno z drugimi metodami. Ker drugih metod za reševanje tega problema še ni, smo razvili preprosto in hitro referenčno metodo. Da postavimo svoje delo v širši kontekst, smo ob prilagoditvi načina merjenja uspešnosti rezultate svoje metode primerjali še z znanimi metodami za reševanje sorodnega problema napovedovanja povezav v velikih neoznačenih omrežjih.

Rezultati kažejo, da je razvita metoda uspešnejša od preproste referenčne, hkrati pa daje tudi boljše napovedi od metod za napovedovanje povezav v velikih neoznačenih grafih. Slednje kaže na to, da se problem, ki smo ga zastavili, bistveno razlikuje od tistega, ki ga rešujejo te metode, zato je prav, da mu posvečamo posebno pozornost.

Ob tem je pomembno tudi, da se na problem dopolnjevanja majhnih omrežij prevedejo konkretne aplikativne naloge. Čeprav so nas na problem napeljali uporabniški vmesniki za vizualno programiranje, majhna označena omrežja lahko srečamo v raznih vejah znanosti in tehnologije.

S poskusi smo pokazali, da s povečevanjem števila povezav v omrežjih naša metoda začne izgubljati svoje prednosti pred splošnejšimi metodami. Njena časovna zahtevnost preprečuje analizo velikih omrežij, prav tako pa začne padati njena napovedna uspešnost. Z nadaljnjim delom na tem področju bo potrebno pokazati, ali je to pomanjkljivost metode, ali gre za občo lastnost omrežij vsaj v določenih domenah.

Tudi metoda Hyspan še ponuja možnosti za nadaljnje raziskovalno delo. Točkovalne funkcije razlikujejo vzorce le na podlagi velikosti in pogostosti, predvsem pa so zanemarjene tako strukturne podobnosti med njimi kot tudi korelacije med njihovimi pojavitvami v učni množici. Z modeliranjem slednjih na primer z uporabo metod strojnega učenja bi bilo morda moč doseči višje napovedni uspešnosti v kompleksnejših množicah podatkov.

gSpan ni edina možna izbira algoritma za odkrivanje pogostih vzorcev. Še več, poleg množice *pogostih* vzorcev obstajajo še drugi koncepti množic vzorcev, ki so značilni za neko domeno omrežij. Zanimiv primer so na primer *zaprti pogosti vzorci* (angl. *closed frequent graph patterns* [29]). Izbira drugega koncepta za model in s tem učnega algoritma bi spremenila tako časovno zahtevnost kot tudi napovedno uspešnost metode.

Pri dopolnjevanju majhnih omrežij se nismo omejili na omrežja z dodano časovno komponento, smo pa predpostavili, da imamo kot učne podatke na voljo množico majhnih omrežij (in ne enega velikega omrežja). Posledično kot model ni možno uporabiti metode, ki v od časa odvisnih omrežjih išče pogoste vzorce časovnega razvoja – GERM [52]. Vzorci časovnega razvoja so tudi zelo nefleksibilni glede časov pojavitve posameznih povezav v vzorcu, kar njihovo uporabnost omeji na zelo ozek krog domen.

5.2 Časovni razvoj lokalnih struktur

Časovni razvoj velikih omrežij je bil doslej obravnavan predvsem na globalni ravni [8]. Leskovec in sodelavci [54] so raziskovali tudi razvoj na, kot so poimenovali v raziskavi, mikroskopskem nivoju. Osredotočili so se na medprihodne čase vozlišč in povezav pri posameznem vozlišču ter na mehanizme, ki določajo *zapiranje trikotnikov*, torej povezovanje vozlišč na razdalji 2.

V tem delu smo postavili teoretične temelje za opazovanje razvoja grafkov, torej

lokalnih vzorcev velikosti do 5 vozlišč, v omrežju skozi čas. Izpeljali smo tudi matematična orodja, ki omogočajo tovrstno analizo z minimalno spremembo obstoječih algoritmov za štetje orbit vozlišč. Pri empirični analizi več zelo različnih omrežij smo prišli do zaključka, da čas, ki poteče od nastanka grafka do njegovega razvoja v gostejši grafek, lahko obravnavamo kot eksponentno porazdeljeno slučajno spremenljivko. Ta porazdelitev ima zgolj en parameter, zato lahko vrednosti parametra za različne orbite medsebojno primerjamo. Različna razmerja parametrov porazdelitev pri različnih omrežjih lahko interpretiramo kot razlike v mehanizmu, ki povzroča rast posameznih omrežij.

Teoretično smo utemeljili družino metod za napovedovanje novih povezav na osnovi analize časovnega razvoja grafkov ter izpeljali matematične izraze, ki omogočajo časovno učinkovito izvedbo. Empirična ocena učinkovitosti na različnih vrstah omrežij in primerjava z učinkovitostjo uveljavljene metode naključnih sprehodov pokažeta, da predlagane napovedne funkcije pogosto niso učinkovite in dosegajo slabše rezultate od metode, ki časovne komponente omrežij sploh ne upošteva. Tudi v kombinaciji z drugimi napovednimi metodami ne prinašajo bistvene dodane vrednosti. Vprašanje, če nam časovni razvoj grafkov lahko pomaga pri lokalnem napovedovanju časovnega razvoja omrežja, tako puščamo odprto. Glede na rezultate, ki smo jih dosegli s skladanjem napovedi ene najboljših metod za dopolnjevanje velikih omrežij, RWR, pri različnih vrednostih parametra α, pa bi bilo zanimivo ta pristop uporabiti tudi izven konteksta časovnega razvoja na omrežjih, na katerih je znano, da RWR dosega dobre rezultate.

LITERATURA

[1] Koji Tsuda in Hiroto Saigo. Graph classification. V *Managing and Mining Graph Data*, str. 337–363. Springer, 2010.

[2] Stephen P Borgatti, Ajay Mehra, Daniel J Brass in Giuseppe Labianca. Network analysis in the social sciences. *science*, 323(5916):892–895, 2009.

[3] Alain Degenne in Michel Forse. *Introducing Social Networks*. Sage Pubn Inc, 1999. ISBN 0761956042.

[4] C. Borgelt. Graph mining: An overview. V *Proc. 19th GMA/GI Workshop Computational Intelligence*, str. 189–203, 2009.

[5] J. Travers in S. Milgram. An experimental study of the small world problem. *Sociometry*, str. 425–443, 1969.

[6] Albert-László Barabási in Réka Albert. Emergence of scaling in random networks. *science*, 286(5439): 509–512, 1999.

[7] D. S. Price. A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science*, 27(5):292–306, 1976.

[8] J. Leskovec, J. Kleinberg in C. Faloutsos. Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations. V *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, str. 177–187. ACM, 2005.

[9] Mary McGlohon, Leman Akoglu in Christos Faloutsos. Weighted Graphs and Disconnected Components: Patterns and a Generator. V *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, str. 524–532, New York, NY, USA, 2008. ACM.

[10] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii in U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824, 2002.

[11] Nataša Pržulj, Derek G. Corneil in Igor Jurisica. Modeling interactome: scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 2004.

[12] M. Kuramochi in G. Karypis. Frequent subgraph discovery. V *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, str. 313–320. IEEE, 2001.

[13] B. Bringmann in S. Nijssen. What is frequent in a single graph? *Advances in Knowledge Discovery and Data Mining*, str. 858–863, 2008.

[14] N. Kashtan, S. Itzkovitz, R. Milo in U. Alon. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, 20 (11):1746–1758, 2004.

[15] S. Wernicke. Efficient detection of network motifs. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 3(4):347–359, 2006.

[16] Y. Kabutoya, K. Nishida in K. Fujimura. Dynamic network motifs: evolutionary patterns of substructures in complex networks. *Web Technologies and Applications*, str. 321–326, 2011.

[17] Tong Ihn Lee, Nicola J Rinaldi, François Robert, Duncan T Odom, Ziv Bar-Joseph, Georg K Gerber, Nancy M Hannett, Christopher T Harbison, Craig M Thompson, Itamar Simon, et al. Transcriptional regulatory networks in saccharomyces cerevisiae. *Science*, 298 (5594):799–804, 2002.

[18] Ron Milo, Shalev Itzkovitz, Nadav Kashtan, Reuven Levitt, Shai Shen-Orr, Inbal Ayzenshtat, Michal Sheffer in Uri Alon. Superfamilies of evolved and designed networks. *Science*, 303(5663):1538–1542, 2004.

[19] Piers J Ingram, Michael PH Stumpf in Jaroslav Stark. Network motifs: structure does not determine function. *BMC genomics*, 7(1):108, 2006.

[20] N. Pržulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2): e177–e183, 2007.

[21] Ryan W. Solava, Ryan P. Michaels in T. Milenković. Graphlet-based edge clustering reveals pathogen-interacting proteins. *Bioinformatics*, 28(18):i480–i486, 2012.

[22] A. M. Bhuiyan, M. Rahman, M. Rahman, M. Al Hasan, et al. Guise: Uniform sampling of graphlets for large graph analysis. V *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012.

[23] Tomaž Hočevar in Janez Demšar. A combinatorial approach to graphlet counting. *Bioinformatics*, str. btt717, 2013.

[24] Akihiro Inokuchi, Takashi Washio in Hiroshi Motoda. An apriori-based algorithm for mining frequent substructures from graph data. V Djamel Zighed, Jan Komorowski in Jan Zytkow, editors, *Principles of Data Mining and Knowledge Discovery*, zvezek 1910 *Lecture Notes in Computer Science*, str. 13–23. Springer Berlin / Heidelberg, 2000. ISBN 978-3-540-41066-9.

[25] R. Agrawal in R. Srikant. Fast algorithms for mining association rules. V *Proc 20th Int Conf Very Large Data Bases VLDB*, zvezek 1215, str. 487–499. Citeseer, 1994.

[26] X. Yan in J. Han. gSpan: Graph-based substructure pattern mining. V *Data Mining, 2002. ICDM 2002. Proceedings. 2002 IEEE International Conference on*, str. 721–724. IEEE, 2002.

[27] T. Werth, A. Dreweke, M. Wörlein, I. Fischer in M. Philippsen. *DAGMA: Mining Directed Acyclic Graphs*. Bibliothek der Universität Konstanz, 2008.

[28] T. Calders, J. Ramon in D. Van Dyck. Anti-monotonic overlap-graph support measures. V *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, str. 73–82. IEEE, 2008.

[29] X. Yan in J. Han. Closegraph: mining closed frequent graph patterns. V *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, str. 286–295. ACM, 2003.

[30] Jun Huan, Wei Wang, Jan Prins in Jiong Yang. Spin: mining maximal frequent subgraphs from graph databases. V *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, str. 581–586. ACM, 2004.

[31] Xifeng Yan, Hong Cheng, Jiawei Han in Philip S Yu. Mining significant graph patterns by leap search. V *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, str. 433–444. ACM, 2008.

[32] M. Berlingerio, F. Bonchi, B. Bringmann in A. Gionis. Mining graph evolution rules. *Machine Learning and Knowledge Discovery in Databases*, str. 115–130, 2009.

[33] Deepayan Chakrabarti, Christos Faloutsos in Mary McGlohon. Graph mining: Laws and generators. V *Managing and Mining Graph Data*, str. 69–123. Springer, 2010.

[34] P. Erdős in A. Rényi. On the evolution of random graphs. V *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, str. 17–61. Akad. Kiadó, 1960.

[35] W. Aiello, F. Chung in L. Lu. A random graph model for massive graphs. V *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, str. 171–180. Acm, 2000.

[36] J. M. Carlson in J. Doyle. Highly optimized tolerance: A mechanism for power laws in designed systems. *Physical Review E*, 60(2):1412, 1999.

[37] A. Fabrikant, E. Koutsoupias in C. Papadimitriou. Heuristically optimized trade-offs: A new paradigm for power laws in the internet. *Automata, languages and programming*, str. 781–781, 2002.

[38] D. Chakrabarti, Y. Zhan in C. Faloutsos. R-MAT: A recursive model for graph mining. V *Fourth SIAM International Conference on Data Mining*, str. 541, 2004.

[39] J. Leskovec, D. Chakrabarti, J. Kleinberg in C. Faloutsos. Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. *Knowledge Discovery in Databases: PKDD 2005*, str. 133–145, 2005.

[40] Leman Akoglu, Mary McGlohon in Christos Faloutsos. Rtm: Laws and a recursive generator for weighted time-evolving graphs. V *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, str. 701–706. IEEE, 2008.

[41] K. Börner, J. T. Maru in R. L. Goldstone. The simultaneous evolution of author and paper networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5266, 2004.

[42] Usha Nandini Raghavan, Réka Albert in Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76 (3):036106, 2007.

[43] D. Zhou, O. Bousquet, T. N. Lal, J. Weston in B. Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16:321–328, 2004.

[44] K. Tsuda, H. J. Shin in B. Schölkopf. Fast protein classification with multiple networks. *Bioinformatics*, 21(suppl 2):ii59, 2005.

[45] D. Liben-Nowell in J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7): 1019–1031, 2007.

[46] Linyuan Lü in Tao Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150 – 1170, 2011. ISSN 0378-4371.

[47] Tao Zhou, Linyuan Lü in Yi-Cheng Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630, 2009.

[48] Ming-Sheng Shang, Linyuan Lü, Wei Zeng, Yi-Cheng Zhang in Tao Zhou. Relevance is more significant than correlation: Information filtering on sparse data. *EPL (Europhysics Letters)*, 88(6):68008, 2009.

[49] L. Backstrom in J. Leskovec. Supervised random walks: predicting and recommending links in social networks. V *Proceedings of the fourth ACM international conference on Web search and data mining*, str. 635–644. ACM, 2011.

[50] M. Kim in J. Leskovec. The network completion problem: Inferring missing nodes and edges in networks. *SDM'11*, 2011.

[51] B. Bringmann, M. Berlingerio, F. Bonchi in A. Gionis. Learning and predicting the evolution of social networks. *Intelligent Systems, IEEE*, 25(4):26–35, 2010.

[52] M. Lahiri in T. Y. Berger-Wolf. Structure prediction in temporal networks using frequent subgraphs. V *Computational Intelligence and Data Mining, 2007. CIDM 2007. IEEE Symposium on*, str. 35–42. IEEE, 2007.

[53] J. Leskovec, D. Huttenlocher in J. Kleinberg. Predicting positive and negative links in online social networks. V *Proceedings of the 19th international conference on World wide web*, str. 641–650. ACM, 2010.

[54] J. Leskovec, L. Backstrom, R. Kumar in A. Tomkins. Microscopic evolution of social networks. V *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, str. 462–470. ACM, 2008.

[55] Katherine Wolstencroft, Robert Haines, Donal Fellows, Alan Williams, David Withers, Stuart Owen, Stian Soiland-Reyes, Ian Dunlop, Aleksandra Nenadic, Paul Fisher, et al. The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. *Nucleic acids research*, 2013.

[56] J. Demšar, B. Zupan, G. Leban in T. Curk. Orange: From experimental machine learning to interactive data mining. *Knowledge discovery in databases: PKDD 2004*, str. 537–539, 2004.

[57] Miller McPherson, Lynn Smith-Lovin in James M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

[58] Tijana Milenković, Weng Leong Ng, Wayne Hayes in Nataša Pržulj. Optimal network alignment with graphlet degree vectors. *Cancer informatics*, 9(8): 121–137, 2010.

[59] Xifeng Yan in Jiawei Han. gSpan: Graph-based substructure pattern mining, expanded version. *Urbana-Champaign: UIUC Technical Report*, 2002.

[60] Stephen A. Cook. The complexity of theorem-proving procedures. V *Proceedings of the third annual ACM symposium on Theory of computing*, str. 151–158. ACM, 1971.

[61] M. Kuramochi in G. Karypis. Finding frequent patterns in a large sparse graph. *Data mining and knowledge discovery*, 11(3):243–271, 2005.

[62] M. Fiedler in C. Borgelt. Support computation for mining frequent subgraphs in a single graph. V *Proc. 5th Int. Workshop on Mining and Learning with Graphs (MLG 2007, Florence, Italy), Florence, Italy*, str. 25–30. Citeseer, 2007.

[63] M. Wörlein. Extension and parallelization of a graph-mining-algorithm. *Master's thesis, Friedrich-Alexander-Universität, Erlangen-Nürnberg*, 2006.

[64] Matija Polajnar in Janez Demšar. Small network completion using frequent subnetworks. *Intelligent Data Analysis*, 2014.

[65] Lovro Šubelj. *Odkrivanje skupin vozlišč v velikih realnih omrežjih na osnovi izmenjave oznak*. PhD thesis, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2013.

[66] Wei-Ke Xiao, Jie Ren, Feng Qi, Zhi-Wei Song, Meng-Xiao Zhu, Hong-Feng Yang, Hui-Yu Jin, Bing-Hong Wang in Tao Zhou. Empirical study on clique-degree distribution of networks. *Physical Review E*, 76(3): 037102, 2007.

[67] H. A. Jung. Zu einem Isomorphiesatz von H. Whitney für Graphen. *Mathematische Annalen*, 164(3):270–271, 1966.

[68] Xiaolin Shi, Lada A. Adamic in Martin J. Strauss. Networks of strong ties. *Physica A: Statistical Mechanics and its Applications*, 378(1):33–47, 2007.

[69] Frank Kose, Wolfram Weckwerth, Thomas Linke in Oliver Fiehn. Visualizing plant metabolomic correlation networks using clique–metabolite matrices. *Bioinformatics*, 17(12):1198–1208, 2001.

[70] Gergely Palla, Imre Derényi, Illés Farkas in Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[71] Josep Díaz, Mathew D Penrose, Jordi Petit in María Serna. Convergence theorems for some layout measures on random lattice and random geometric graphs. *Combinatorics, Probability and Computing*, 9(06):489–511, 2000.

[72] Event data sets. http://eventdata.psu.edu/data.html, 2013. Pridobljeno 12. 12. 2013.

[73] J. Leskovec, D. Huttenlocher in J. Kleinberg. Signed networks in social media. V *Proceedings of the 28th international conference on Human factors in computing systems*, str. 1361–1370. ACM, 2010.

[74] Sean X Chen in Jun S Liu. Statistical applications of the poisson-binomial and conditional bernoulli distributions. *Statistica Sinica*, 7(4):875–892, 1997.