UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO
FAKULTETA ZA MATEMATIKO IN FIZIKO

Lan Žagar

PREDSTAVITEV PODATKOV IN ODKRIVANJE ZNANJ S TEHNIKO VEČNIVOJSKIH MREŽ

Diplomska naloga
na univerzitetnem študiju

Mentor: izr. prof. dr. Blaž Zupan

Ljubljana, 2008

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Namesto te strani **vstavite** original izdane teme diplomskega dela s podpisom mentorja in dekana ter žigom fakultete, ki ga diplomant dvigne v študentskem referatu, preden odda izdelek v vezavo!

Zahvala

Zahvaljujem se mentorju izr. prof. dr. Blažu Zupanu za pomoč in usmerjanje pri izdelavi diplomske naloge.

Prav tako bi se zahvalil dr. Tomažu Curku, Mihi Štajdoharju, Gregorju Rotu, Lanu Umeku in vsem ostalim članom Laboratorija za umetno inteligenco za pomoč pri delu.

Zahvala pa gre tudi moji družini, ki me je med študijem podpirala in mi stala ob strani.

Kazalo

Seznam uporabljenih oznak in definicij

$N = (V, E)$ - večnivojska mreža z množico vozlišč V in množico povezav E

$N_i = (V_i, E_i)$ - mreža i-tega nivoja

$N_{i,j} = (V_i \cup V_j, E_{i,j})$ - mreža vmesnega nivoja

$V = \bigcup_i V_i$ - množica vseh točk v večnivojski mrežni predstavitvi

V_i - množica točk i-tega nivoja

$E = \bigcup_i E_i \cup \bigcup_{i,j} E_{i,j}$ - množica vseh povezav v večnivojski mrežni predstavitvi

$E_i = \{\{u, v\} \mid \{u, v\} \in E, u \in V_i,\ v \in V_i\}$ - množica povezav mreže i-tega nivoja

$E_{i,j} = \{\{u, v\} \mid \{u, v\} \in E, u \in V_i,\ v \in V_j\}$ - množica povezav med mrežama na nivoju i in j

P_N - predstavitev (projekcija) mreže N v dvodimenzionalni ravnini

$\|e\|_{P_N}$ - dolžina povezave e (evklidska razdalja med krajišči) v projekciji P_N

$m_e(X)$ - mediana vrednosti v množici X

Povzetek

V okviru diplomskega dela je bil razvit nov način vizualizacije podatkov in odkrivanja znanj, ki uporablja mreže. Vizualizacija z mrežami se je standardno uporabljala za prikaz ene vrste relacij. Pogosto pa imamo na voljo še dodatne nabore podatkov in relacije, ki jih povezujejo. Več različnih vrst podatkov v takih relacijskih bazah smo želeli hkrati prikazati v eni sami vizualizaciji. Problema smo se lotili s prikazovanjem več med seboj povezanih nivojev mrež. Omejili smo se na dva nivoja, s katerima smo predstavili dva nabora podatkov o različnih entitetah. Predlagali smo postopek, ki optimizira predstavitev večnivojskih mrež v ravnini, kar nam omogoča, da jih všečno narišemo. Podali smo tudi nekaj objektivnih kriterijev za ocenjevanje dobljenih vizualizacij. Preizkusi na sintetičnih podatkih so pokazali, da hkratna optimizacija smiselno uredi točke tako, da upošteva zakonitosti obeh nivojev. Preverili smo tudi možnost konkretne uporabe na podatkih s področja bioinformatike. Mrežo genov smo uspešno dopolnili z mrežo oznak MeSH na drugem nivoju. Te oznake so se po optimizaciji značilno umestile v okolice genskih skupin in jih s tem dodatno razložile.

Ključne besede:

večnivojske mreže, relacijske podatkovne baze, odkrivanje znanj, optimizacija na osnovi sil, genske mreže

Abstract

We present a new technique for network visualization and network-based data mining. Standard network visualization techniques most often focus on a single-type relations and are used for visualization of a single data set. In practical problem solving, however, additional data sets and relations that relate them are available. Our specific goal in this thesis was to address the problem of visualization of multiple datasets from a relational database. Our proposed approach is based on multi-layer networks. In this study we use only two layers representing two different datasets. A method for optimizing the layout of a multi-layer network was proposed. Several objective criteria for evaluation of network visualizations were also developed. Simulations on synthetic data sets showed that the proposed optimization technique performs well in simultaneous optimization of two-layered network with respect to the structure of both layers. We have also studied the performance of the technique in bioinformatical application, where a gene network was successfully complemented with a network of MeSH terms resulting in an informative two-layer network. After the optimization step, several MeSH terms were placed near related gene clusters and thus provided additional insight into the identified gene sets.

Keywords:

multi-layer networks, relational databases, data mining, force-based optimization, gene networks

Poglavje 1

Uvod

Količina podatkov, ki nam je na voljo, se je v zadnjih desetletjih zelo povečala. Medtem ko je bil včasih glavni problem pridobiti podatke, je sedaj, ko jih je na voljo ogromno, težava predvsem, kako jih obdelati in uporabiti. Tudi v znanosti in raziskovanju, še posebej pa pri raziskavah iz naravoslovnih področij, kot so kemija, biologija, medicina, ... se vedno bolj pogosto uporabljajo tehnike avtomatskega izvajanja eksperimentov in zajemanje njihovih rezultatov, tako da tudi tu količina podatkov v zadnjem času silovito narašča.

Po pridobitvi podatkov je naslednji smiselni korak njihova analiza ter iskanje zakonitosti. Analizo lahko ročno izvajajo domenski strokovnjaki, pri večjih količinah podatkov pa morajo seveda tudi ti poseči po specializiranih orodjih. Na poprej omenjenem področju raziskav so se v zadnjih nekaj letih intenzivno pričela uporabljati orodja strojnega učenja in odkrivanja znanj iz podatkov, ki lahko pomagajo analitikom odkrivati v podatkih skrite povezave in zakonitosti. Ključni element teh tehnik je predstavitev odkritih povezav v pregledni, informativni in po možnosti interaktivni obliki. Te pogoje v praksi največkrat izpolnjujejo različne vizualizacije ali kombinacije grafičnih elementov z dodatnimi podatki v drugih oblikah.

V zadnjem desetletju izjemno popularna vizualizacijska tehnika je vizualizacija mrež. Z njimi lahko množico elementov in njihove relacije predstavimo

kot točke in povezave med njimi. Ker točke in povezave predstavljajo entitete oziroma koncepte iz realnega sveta, jih navadno lahko spremljajo dodatne informacije in podatki. Te lastnosti lahko predstavimo z različnimi barvami točk, njihovo velikostjo, obliko, ..., povezave pa so lahko usmerjene, imajo različno debelino, barvo ...

Mreže pogosto uporabljajo raziskovalci iz naravoslovnih področij, kot so biologija, biomedicina, in kemija, ter raziskovalci iz družboslovnih ved. Tam imajo namreč pogosto opravka z večjimi količinami podatkov, ki se jih da z mrežami dobro prikazati, na primer baze podobnih genov, kemikalij, procesov, ljudi in organizacij.

Običajno se pri standardnih tehnikah odkrivanja zakonitosti v podatkih uporablja en sam nabor podatkov oziroma eno samo množico primerov, kjer vsak primerek predstavlja neko entiteto, opisano z izbrano množico atributov. Predvsem pri analizi podatkov iz kompleksnih področij pa se pogosto zgodi, da imamo dodaten nabor podatkov o istih entitetah, dobljen pri spremenjenih pogojih ali opisan z atributi iz popolnoma drugačne domene. Lahko so nam na voljo podatki o kakšnem izmed atributov ali pa o čisto drugih entitetah, ki pa se jih da s pomočjo tretjega nabora povezati s prvimi podatki.

S stališča shranjevanja podatkov sodijo tovrstni podatki na področje relacijskih podatkovnih baz. V diplomski nalogi obravnavamo področje vizualizacije relacij v relacijskih podatkovnih bazah ter izdelave orodij, ki nam preko vizualizacije lahko pomagajo pri odkrivanju v podatkih skritih vzorcev. Vizualizacija, s katero smo se lotili problema, je prikaz več nivojev mrež. V nalogi smo se omejili na dva nivoja, s katerima smo predstavili dva nabora podatkov o različnih entitetah in njihovih mednivojskih povezavah. V taki vizualizaciji so podobnostne povezave v prvem naboru podatkov predstavljene z eno mrežo, povezave med podatki v drugem naboru z drugo mrežo v vzporedni ravnini, relacijo, ki povezuje oba nabora podatkov, pa predstavljajo povezave med točkami različnih ravnin.

Cilj pričujoče naloge je bil razviti algoritem za odkrivanje razporeditve točk

po obeh nivojih mrež na način, ki bi nam omogočil prepoznati zakonitosti v podatkih, ki bi jih bilo v kakšni drugi predstavitvi težje opaziti. Tak algoritem mora poiskati primeren prikaz podatkov, ki je namenjen preprostemu in udobnemu pregledovanju ter predstavitvi večjih količin podatkov z znanimi ali neznanimi lastnostmi. Dodatni cilj nalogi je bil razviti mere za oceno kvalitete vizualizacij obravnavanih mrež ter mere za oceno stabilnosti posameznih točk, ki jih v teh mrežah vizualiziramo.

Diplomska naloga je sestavljena iz petih poglavij. V uvodu smo se seznanili z obravnavano tematiko in problemom, ki ga rešujemo, ter opisali naš pristop. V naslednjem poglavju bomo na kratko predstavili dve področji: mreže in odkrivanje znanj iz relacijskih podatkovnih baz. Nato bomo v tretjem poglavju bolj natančno opisali metode, ki smo jih razvili za reševanje našega problema. Poleg postopka za optimizacijo razporeditve točk v večnivojski mreži bomo tu spoznali tudi kriterije za ocenjevanje elementov vizualne predstavitve podatkov in orodja, uporabljena pri razvoju omenjenih metod. O eksperimentih, ki smo jih izvedli na sintetičnih podatkih in podatkih s področja bioinformatike ter njihovih rezultatih, poročamo v četrtem poglavju. Delo zaključuje kratek opis rezultatov naloge in opis možnosti za nadaljnje delo ter nadgradnje.

Poglavje 2

Pregled sorodnega dela

V diplomski nalogi bomo problematiko predstavitve relacijskih podatkovnih baz in odkrivanja znanj iz njih reševali s tehniko večnivojskih mrežnih vizualizacij. Smiselno je, da se poprej seznanimo tako s področjem problema in sorodnimi rešitvami kot tudi z osnovami, na katerih temelji naša rešitev.

2.1 Odkrivanje znanj iz relacijskih podatkovnih baz

V Uvodu smo omenili, da standardne tehnike za odkrivanje znanj iz podatkov delujejo na podatkih, zapisanih v eni tabeli, ki primerke opisuje z nekim naborom atributov. Seveda pa so se ob razširitvi relacijskih baz podatkov, kjer imamo več med seboj povezanih tabel, pojavile tudi metode, ki znajo uporabiti take podatke. Poglejmo si nekaj primerov iz različnih tipov strojnega učenja.

Nadzorovano učenje (ang. *supervised learning*). Najbolj razširjena in obdelana vrsta učenja. V primeru več tabel s podatki iz različnih domen lahko zgradimo več klasifikatorjev in nato njihove rezultate kombiniramo. Druga možnost je, da že klasifikator priredimo tako, da zna delati z različnimi podatki. V [6] so uporabili metodo podpornih vektorjev, ki

je uporabljala linearno kombinacijo večih jeder za delo s podatki iz več domen.

Delno nadzorovano učenje (ang. *semi-supervised learning*). V situaciji, kjer imamo nekaj primerov označenih in nekaj neoznačenih, nam več neodvisnih opisov podatkov lahko zelo pomaga. Znana metoda, ki to izkorišča, je *co-training* [1].

Nenadzorovano učenje (ang. *unsupervised learning*). Tukaj imamo opravka samo z neoznačenimi primeri, kar ponavadi velja tudi za podatke, ki jih prikazujemo z mrežami. Klasičen primer nenadzorovanega učenja je razvrščanje (ang. *clustering*), kjer primerke, opisane z vektorji glede na njihovo podobnost, delimo v podskupine. Posplošitev, ki v nekem smislu dela z več vrstami podatkov, je dvorazvrščanje (ang. *biclustering* ali *co-clustering*) [7, 11]. Tukaj ne iščemo samo značilnih skupin primerkov (vrstic v tabeli), ampak kar značilne skupine primerkov in atributov hkrati (podmatrik v tabeli). Če imamo na voljo sorodne podatkovne baze (na primer o atributih), jih lahko uporabimo pri računanju podobnosti.

Pogojno bi v ta del lahko vključili tudi metode za transformacijo in prikazovanje podatkov, kot je na primer korespondenčna analiza (ang. *correspondence analysis*) [4]. Ta je zelo sorodna vizualizacijam z večnivojskimi mrežami, saj pri njej tudi prikazujemo z eno vizualizacijo entitete dveh vrst. Podatki so pri tej tehniki podani v obliki kontingenčne tabele.

2.2 Mreže in grafi

Za razliko od večine zgornjih primerov pri vizualizacijah z mrežami ne delamo s podatki v atributnem zapisu. Če take podatke vseeno dobimo, jih lahko pretvorimo v obliko primerno za grajenje mrež tako, da s pomočjo atributov definiramo mero podobnosti med primerki. To je nato s pomočjo pragovne funkcije lahko prevesti v relacijo povezanosti.

Za opis in definicijo mrež moramo najprej vedeti nekaj o grafih in teoriji, ki jih preučuje. Teorija grafov je prisotna že kar nekaj časa, za njen začetek pa mnogi smatrajo objavo Leonharda Eulerja o problemu sedmih mostov Königsberga. Teorija preučuje abstraktne objekte — grafe, ki so definirani kot par množice točk V in množice povezav E (definicija 2.1).

$$G = (V, E) \tag{2.1}$$

Povezave so lahko neurejeni pari točk $\{u, v\}$ ali urejeni pari (u, v). V prvem primeru govorimo o neusmerjenih, v drugem pa o usmerjenih povezavah (množico usmerjenih povezav pogosto označujemo s črko A namesto E). Poznamo več vrst grafov, najpogosteje pa se ukvarjamo z enostavnimi grafi, to je neusmerjenimi grafi brez vzporednih povezav in zank.

V zadnjih letih se v povezavi z grafi pogosto uporablja izraz mreža [8] (definicija 2.2):

Mreža je graf z dodatnimi informacijami o točkah ali povezavah (2.2)

Omeniti velja, da dodatne informacije mreže ne vplivajo na njeno strukturo, ki je določena s točkami in povezavami. Ko bomo v nadaljevanju te naloge opisovali metode, dodatnih informacij zato pogosto ne bomo neposredno podajali, saj bo definicija strukture zadostovala.

Mreže so uporabno orodje za abstraktno predstavitev objektov in pojavov iz našega sveta. Potrebno se je le odločiti, kaj bodo točke predstavljale, in definirati relacijo povezanosti. Slednje lahko naredimo tako, da iz znanih podatkov o objektih izpeljemo mero podobnosti in z uporabo pragovne funkcije določimo, katere točko bodo povezane in katere ne.

Vendar pa nam abstrakten opis grafov in mrež včasih preprosto ne zadostuje. Veliko lažje si jih je predstavljati, če jih vidimo grafično upodobljene — npr. narisane v ravnini. Za dobro predstavitev mreže v ravnini pa je zelo pomembna razporeditev točk. Struktura kompleksnejše mreže je namreč ob

nepazljivi razporeditvi lahko na prvi pogled popolnoma neprepoznavna. Da dosežemo ugodno predstavitev, je potrebna neka vrsta optimizacije, v katere namene je bilo razvitih že več različnih metod in pristopov. Pri nekaterih omejimo možne pozicije točk na manjši del celotne ravnine (npr. na krožnico ali križišča pravokotne mreže) in nato med manj rešitvami iščemo čim boljšo. Druga vrsta so spektralne metode [5], ki delujejo na osnovi računanja lastnih vrednosti in vektorjev. Zanimive so, ker lahko optimalno rešijo določen minimizacijski problem in s tem najdejo najboljšo možno predstavitev grafa glede na nek ocenjevalni kriterij (kar pa ne pomeni nujno, da bo ta predstavitev tudi ljudem najbolj všeč). Uporabne so predvsem za risanje grafov z večjo stopnjo simetrije, saj le-to optimalne predstavitve ponavadi obdržijo.

V zadnjem času pa so verjetno najbolj popularne metode optimizacij, ki delujejo na osnovi privlačnih in odbojnih sil. Pri takem pristopu na točke gledamo kot na fizikalne delce, na katere delujejo sile, odvisne od razdalje med točkami. Odbojne sile delujejo med vsemi pari točk, privlačne pa samo med povezanimi. Cilj algoritma je minimizirati energijo sistema, kar pa je zelo težko rešiti optimalno. Metode zato z iterativnimi postopki iščejo čim boljše približke. Eno takih metod sta predlagala T.M.J. Fruchterman in E.M. Reingold [3]. Je preprosta za razumevanje in v praksi daje dobre rezultate, zato se jo pogosto uporablja. Tudi metode za optimizacijo predstavitev večnivojskih mrež, razvite v tej nalogi, so osnovane na algoritmu Fruchterman–Reingold. Psevdokoda postopka, kot je bila zapisana v članku, je priložena v dodatku A.

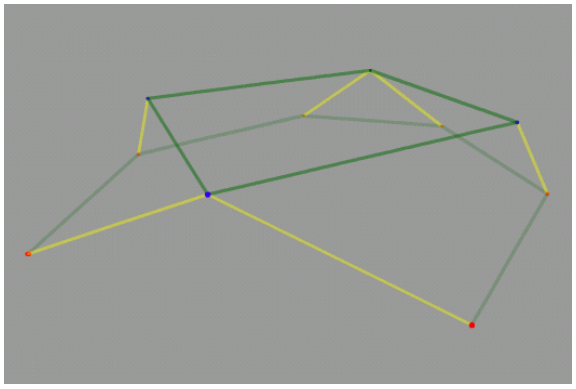
Poglavje 3

Razvite metode in pristopi

V prejšnjih dveh poglavjih smo v kratkem pregledu vizualizacij z mrežami spoznali, kako lahko prikažemo en sam nabor podatkov. V nalogi smo to predstavitev želeli razširiti tako, da bi bilo možno z eno vizualizacijo hkrati prikazati več med seboj povezanih množic podatkov. Oglejmo si ta problem na primeru podatkov "Pot in cikel" (PC), ki so sestavljeni iz poti na 6 točkah (P_6) na prvem nivoju in cikla s 4 točkami (C_4) na drugem. Formalno so podatki opisani z enačbami 3.1 in prikazani na sliki 3.1.

$$\begin{array}{lrcl} N_1: & V_1 & = & \{v_{1,1}, v_{1,2}, \ldots, v_{1,6}\} \\ & E_1 & = & \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_1\}\} \\ N_2: & V_2 & = & \{v_{2,1}, v_{2,2}, v_{2,3}, v_{2,4}\} \\ & E_2 & = & \{\{v_{2,i}, v_{2,i+1}\},\ i = 1, \ldots, 5\} \\ N_{12}: & E_{12} & = & \{\{v_{1,1}, v_{2,1}\}, \{v_{1,2}, v_{2,2}\}, \{v_{1,3}, v_{2,3}\}, \\ & & & \{v_{1,4}, v_{2,3}\}, \{v_{1,5}, v_{2,4}\}, \{v_{1,6}, v_{2,1}\}\} \end{array} \tag{3.1}$$

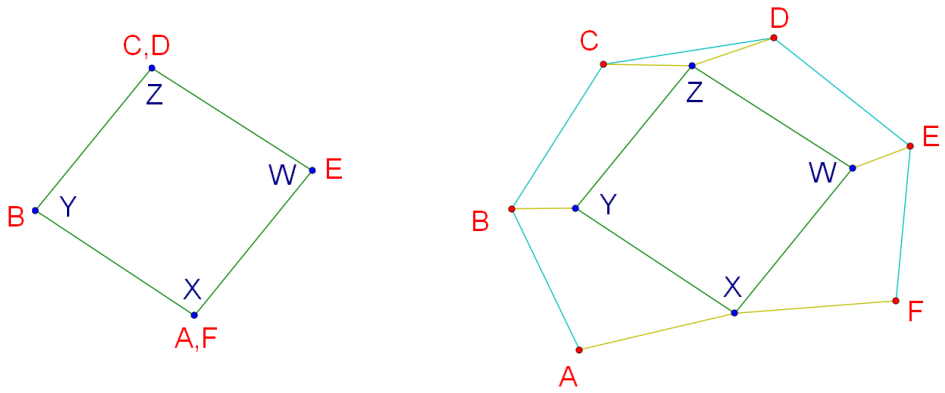
Tu je vredno omeniti, da podatki, ki jih navadno obdelujemo z metodami odkrivanja znanj, originalno niso dostopni v obliki grafov in mrež, marveč v obliki tabel, kjer je posamezen primer opisan z naborom atributov. Za predstavitve, kot jih uporabljamo v tej diplomski nalogi, take primere postavimo v mrežo tako, da izberemo določeno funkcijo podobnosti ter z določitvijo praga



Slika 3.1: 3D prikaz podatkov “Pot in cikel” (PC)

podobnosti podobnostno matriko prevedemo v nabor relacij. Relacija med dvema elementoma tako pove, da je njuna podobnost večja od izbranega praga. Kot bomo to predstavili na primerih iz področja bioinformatike, je ta pristop moč uporabiti tako za pridobivanje mrež na posameznih nivojih kot tudi za določanje povezav med mrežami.

Eden od možnih pristopov predstavitve podatkov PC je, da najprej izrišemo mrežo enega nabora podatkov in nato podatke iz drugega nabora poskusimo dodati tej vizualizaciji. To lahko naredimo tako, da uporabimo vrednosti iz drugih podatkov za določanje lastnosti točk (na primer barve, velikosti ipd.) ali pa jim dodamo spremno besedilo ali oznake (v primeru na sliki 3.2a so to kar črke). Vendar pa je to smiselno samo za nekatere najbolj preproste primere. Dodatni podatki, ki so v relacijskih bazah podani v drugih tabelah, so večinoma bolj kompleksni in imajo tudi svojo strukturo, ki jo želimo prikazati. Zato smo v diplomski nalogi razvili alternativni pristop, ki podatke iz različnih tabel uredi v mrežah na različnih nivojih, povezave med podatki v tabelah pa prikaže kot povezave med mrežami (slika 3.2b).



(a) (b)

Slika 3.2: Enonivojska (a) in dvonivojska (b) vizualizacija podatkov PC

3.1 Optimizacija več nivojev mrež

Postopna optimizacija

Kadar so dodatne informacije strukturirane tako, da lahko tudi iz njih sestavimo mrežo, tako kot smo to predstavili v uvodnem primeru, lahko le-to projiciramo v isto ravnino. Pri optimizaciji razporeditve točk druge mreže upoštevamo predhodno izračunane koordinate točk prve mreže. Zaradi povezav med točkami obeh mrež upamo, da se bodo točke druge mreže razporedile tako, da bodo blizu sorodnih točk prve mreže. Postopek je zelo preprost in ga lahko povzamemo z dvema korakoma:

1. Optimiziramo P_{N_1} z izbranim algoritmom (npr. Fruchterman–Reingold)
2. V isto ravnino razporedimo še točke V_2, pri čemer upoštevamo povezave E_2 in E_{12} v kombinaciji z že izračunanimi koordinatami točk V_1

S tako vizualizacijo je možno prikazati dve sorodni mreži in za skupino točk v eni mreži najti ustrezne značilne točke v drugi. Vendar pa ima ta pristop pomanjkljivosti, saj prvo mrežo optimizira samostojno, brez uporabe informacij iz ostalih podatkov. Zaradi tega se lahko zgodi, da drugega koraka

optimizacije ni mogoče dobro izvesti. Če pa bi prvi del optimizirali drugače, bi se mogoče tudi drugi del dalo dobro optimizirati. Končni rezultat bi tako lahko bil boljši v zameno za le malenkost slabšo ali kar enako dobro optimizacijo P_{N_1}. Kljub omenjenim pomanjkljivostim smo postopek implementirali, da bi ga lahko primerjali z boljšo različico, opisano v nadaljevanju.

Hkratna optimizacija

Zgornji postopek optimizacije lahko izboljšamo tako, da podatke predstavimo z dvonivojsko mrežo in oba nivoja uredimo hkrati. Za osnovo smo podobno kot zgoraj uporabili algoritem Fruchterman–Reingold in ga ustrezno prilagodili. Točke iz različnih nivojev smo obravnavali ločeno, kar si lahko predstavljamo tako, kot da bi bili mreži N_1 in N_2 vsaka v svoji, vzporedni ravnini. Pri dejanskem računanju v resnici uporabljamo samo koordinati x in y, torej projekcije točk na skupno ravnino.

Privlačne sile računamo med vsemi povezanimi pari točk, odbojne pa samo med točkami istega nivoja. Slednje omogoča, da se nad skupino točk prvega nivoja postavi sorodna točka drugega nivoja, ne da bi jo te "odrinile" stran.

Zaradi različnih vrst povezav želimo ločiti tudi privlačne sile, ki jih te povezave povzročijo. To smo dosegli z novim parametrom λ, ki predstavlja moč mednivojskih povezav. Podamo ga ob klicanju funkcije za optimizacijo oz. se uporabi privzeta vrednost $\lambda = 1$, če tega ne storimo. Parameter nam pove, kolikokrat manjše so sile, ki so posledica povezav iz E_{12} v primerjavi s silami znotraj istega nivoja (povezave E_1 in E_2). Pri vrednosti $\lambda = 1$ se povezave obravnavajo enako. Pri $\lambda = 2$ se (v projekciji) enako oddaljeni točki privlačita dvakrat manj in pri $\lambda = \frac{1}{2}$ dvakrat bolj, če sta iz različnih nivojev, kot če sta iz istega.

V grobem si lahko mislimo, da parameter λ določa stopnjo interakcije med nivojema. Z njim lahko nastavimo, ali naj se vsaka zase bolje optimizirata predstavitvi mrež P_{N_1} ter P_{N_2} ali predstavitev mednivojske mreže $P_{N_{12}}$.

Sledi povzetek značilnosti, bolj natančen opis s psevdokodo pa je v do-

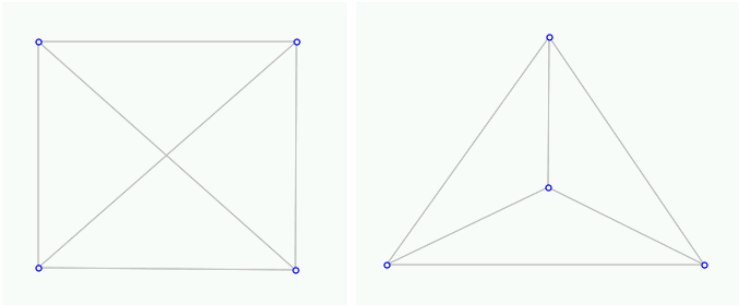
datku B.

1. Optimiziramo projekcijo P_N celotne mreže naenkrat
2. Privlačijo se točke, ki so med sabo povezane:
   u in v se privlačita $\iff \{u, v\} \in E$
3. Odbijajo se samo točke iz istega nivoja:
   u in v se odbijata $\iff u \in V_i \wedge v \in V_i$
4. Dodaten parameter λ uravnava moč mednivojskih povezav in s tem prioriteto optimizacije posameznih projekcij nivojev (P_{N_1}, P_{N_2}) oz. projekcije vmesnega nivoja $(P_{N_{12}})$

3.2 Ocenjevanje elementov vizualne predstavitve

Vizualizacije podatkov so namenjene ljudem in so zato tudi podvržene človeški presoji o njihovi kvaliteti. To predstavlja problem pri poizkusu objektivnega ocenjevanja različno optimiziranih prikazov in dobljenih rezultatov. Nekaj osnovnih kriterijev pa je lahko skupnih vsem uporabnikom. Vendar bomo kmalu naleteli na elemente, ki imajo tako prednosti kot slabosti, zaradi česar ne moremo reči, da je kateri izmed načinov boljši od drugih.

Preprost primer je recimo risanje grafa K_4. Dve simetrični in človeškemu očesu prijazni razporeditvi sta kvadrat z diagonalama (slika 3.3a) in trikotnik s sredinsko točko, povezano z vsemi oglišči (slika 3.3b). Prvi prikaz je malo bolj pogost in verjetno tisti, na katerega večina ljudi najprej pomisli ob omembi K_4. Razlog bi lahko bil v pogosti vizualizaciji polnih grafov na način, pri katerem točke enakomerno razporedimo po namišljeni krožnici in jih nato med sabo povežemo. Vendar pa ima druga razporeditev manj križanj, kar je pri predstavitvah grafov zelo zaželena lastnost in jo ponavadi pri ocenjevanju upoštevamo.



(a) (b)

Slika 3.3: Dve predstavitvi grafa K_4

Dokler se te problematike zavedamo, je vseeno uporabno definirati objektivne, po možnosti numerične načine ocenjevanja. To nam omogoča hiter pregled večje količine ocenjevanih objektov in avtomatsko primerjavo ali izbiro najboljšega. V nadaljevanju bomo zato predstavili nekaj možnih načinov ocenjevanja mrež, posameznih točk in kompatibilnosti nivojev, s katerimi smo si pri svojem delu pomagali.

3.2.1 Ocenjevanje kvalitete mrežnih vizualizacij

Kot je bilo že omenjeno, na izgled mreže kot celote vpliva več kriterijev. Vendar pa pri ocenjevanju navadno ne uporabljamo sestavljenih ocen, ki bi poizkušale zajeti vse lastnosti mreže. Raje se odločimo za en osnovni kriterij, ki se nam zdi v določenem kontekstu najbolj relevanten. Če je potrebno, pa lahko navedemo tudi več samostojnih ocen, ki opisujejo različne pomembne lastnosti.

V praksi, ko se ukvarjamo s podatki iz realnega sveta, mreže ponavadi niso simetrične, prav tako se jih ne da vložiti v ravnino brez križanj povezav. Ravno nasprotno — klike in polni dvodelni grafi so pogosti motivi in se pojavljajo kot podgrafi, zaradi česar se velikemu številu križanj niti ne moremo izogniti. Podatek o njihovem točnem številu zato ni med najbolj zanimivimi in ga ne bi izbrali kot glavni kriterij. Pogosto pa se izkaže, da imajo po nekem drugem izbranem kriteriju bolje ocenjene mreže tudi manjše

število križanj. Prepoznavanje podgrafov, ki izkazujejo določeno simetrijo, in posebna optimizacija takih posameznih komponent tudi nista enostavni in časovno učinkoviti. V večjih in kompleksnejših mrežah iz realnih domen (biologije, medicine, družboslovja, ...) ti pristopi ponavadi niso praktični.

Bolj splošno pravilo za dobre razporeditve je, da naj bodo med sabo povezane točke čim bolj skupaj. Seveda moramo razdalje nekako normirati, da ne bi velikost enot vplivala na oceno sicer enakih mrež. Taki zahtevi se ujemata tudi z osnovnima principoma metode Fruchterman–Reingold, ki smo jo uporabili kot osnovo pri svojem delu. Zato smo tudi za ocenjevanje izbrali funkcijo, ki sledi tema vodiloma. Bližino povezanih točk izmerimo kot mediano vseh razdalj povezanih parov, oceno pa normaliziramo z mediano razdalj nepovezanih parov.

$$q(P_N) = \frac{m_e(\{\|e\|_{P_N} \mid e \in E(N)\})}{m_e(\{\|e\|_{P_N} \mid e \notin E(N)\})} \tag{3.2}$$

Namesto mediane bi lahko izbrali povprečje, kar bi pomenilo slabše ocene za mreže z nekaj zelo dolgimi (slabo optimiziranimi) povezavami. Take povezave so sicer pri risanju mrež res nezaželene, a smo se v našem primeru odločili, da naj nekaj osamelcev ne pokvari celotne ocene. Zaradi poizkusa optimizacije več nivojev se namreč pogosto ne moremo izogniti kakšni slabi povezavi, s čimer pa smo se pripravljeni sprijazniti, če to pomeni sicer boljši končni rezultat.

S kriterijsko funkcijo (3.2) lahko ločeno ocenimo predstavitev prvega (P_{N_1}) ali drugega (P_{N_2}) nivoja kot tudi njuno prepletanje ($P_{N_{12}}$). Te tri ocene so povsem samosvoje in se, kot bomo videli kasneje, lahko v procesu optimizacije obnašajo zelo različno. Zato se nam je zdela bolj smiselna uporaba in navajanje vseh treh ocen ločeno, kot računanje ene skupne ocene. Pri združevanju se namreč ne bi mogli izogniti izgubi informacije.

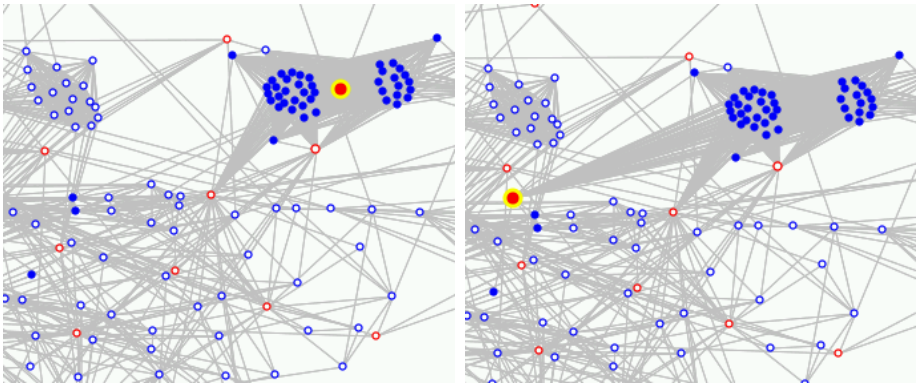
3.2.2 Ocenjevanje točk

Poleg ocenjevanja celotne mreže je smiselno ocenjevati tudi specifične dele mreže oz. posamezne točke. Ker se ukvarjamo s predstavitvijo več plasti podatkov, nas najbolj zanima iskanje in prepoznavanje zanimivih točk in soseščin z vidika mednivojskih povezav. Če se pri optimizaciji uspejo sosede opazovane točke razporediti v njeno bližino, se s tem oblikuje skupina, ki morda razkriva neko strukturo v podatkih, ki bi jo bilo sicer težko opaziti. Primer dobro in slabo postavljene točke je prikazan na sliki 3.4.

Točke s takimi soseščinami si želimo imeti poudarjene in hitreje opazne, kar lahko dosežemo tako, da jih narišemo večje. Za določanje velikosti vsake točke pa potrebujemo objektivno oceno njene kvalitete.

$$f_P(v) = m_e(\{\|e\|_P \mid e \in E_{12} \wedge v \in e\}) \tag{3.3}$$

S pomočjo ocene $f_P(v)$, izračunane po formuli (3.3), ocenimo, ali se točka v v predstavitvi P nahaja blizu večine svojih sosed iz drugega nivoja. Da bi izvedeli, ali je rezultat tudi statistično značilen, lahko uporabimo permutacijski test.



(a) (b)

Slika 3.4: Primer dobre (a) in slabe (b) postavitve točke

Permutacijski test [9] je vrsta statističnega testa značilnosti, kjer referenčno porazdelitev pridobimo iz večjega števila naključno spremenjenih podatkov. Iz dobljene porazdelitve nato ocenimo, kolikšen delež le-teh je večji od našega rezultata (ocena p-vrednosti). Če je le-ta manjša od vnaprej predpisane stopnje tveganja (tipično $0,05$), zavrnemo domnevo, da je naš rezultat plod naključja.

Naključno spreminjanje (slučajenje) mreže moramo izvesti previdno. Če obdržimo samo točke in jih povežemo naključno, lahko dobimo povsem drugačno strukturo mreže. Primerjava optimizacije med izvirno mrežo in takimi naključnimi mrežami ni preveč smiselna. Točka mora zato obdržati enako stopnjo, da lahko bližino sosed primerjamo s tisto v izvirni mreži. Spreminjanje mreže, ki nadomesti izvirne mednivojske povezave z naključnimi, pri tem pa ohrani stopnje točk, lahko dosežemo s prevezovanjem povezav. Postopek je sledeč:

1. Naključno izberemo povezavi $\{u, v\}$, $\{x, y\}$ z različnimi krajišči
2. Povezavi odstranimo in ju nadomestimo s povezavama $\{u, y\}$, $\{x, v\}$
3. Točki 1. in 2. ponovimo $(8 \times m)$-krat, kjer je m število vseh povezav[1]

3.2.3 Kompatibilnost nivojev

Mnogokrat bomo imeli opravka s kompatibilni podatki, to je takimi, ki se jih da lepo prikazati z več nivoji mrež in kjer bo na vsakem nivoju ter na povezavah vidna določena struktura. Možno pa je, da se ne glede na razporeditev točk, hkrati ne da dobro optimizirati vseh projekcij nivojev P_{N_1}, P_{N_2} in $P_{N_{12}}$. V takem primeru vizualizacije ne bodo zelo pregledne in všečne. Toda tudi taka informacija pove nekaj o uporabljenih naborih podatkov in bi nas lahko zanimala.

[1]Faktor 8 v točki 3. je bil empirično določen. Testi so celo pokazali, da je zadostno tudi že manjše število ponovitev in je rezultat dobra aproksimacija povsem naključnih povezav.

V namen opazovanja tega fenomena smo definirali mero kompatibilnosti, ki jo izračuna sledeči postopek:

1. Samostojno optimiziramo projekcije vseh nivojev P_{N_i}
2. Izračunamo $q_i = q(P_{N_i})$ oceno projekcije nivoja N_i
3. Hkratno optimiziramo projekcijo celotne mreže P_N
4. Izračunamo $q_i' = q(P_{N_i})$ oceno projekcije nivoja N_i
5. Izračunamo $k_i = \frac{q_i}{q_i'}$

Kvocienta k_1 in k_2 povesta, koliko hkratna optimizacija obeh nivojev poslabša oceno posamezne projekcije nivoja (P_{N_1} oz. P_{N_2}). Pri kompatibilnih podatkih se oceni ne bosta bistveno spremenili, zato bosta k_1 in k_2 blizu 1. Medtem ko pri nekompatibilnih podatkih hkratna optimizacija lahko zelo pokvari ocene posameznih nivojev in sta kvocienta manjša.

3.3 Uporabljena orodja

Večina kode, razvite v namene te diplomske naloge, je bila napisana v programskem jeziku Python (`www.python.org`). Ker je to skriptni jezik, ni zelo učinkovit kar se tiče hitrosti, a je bil za naše potrebe primeren predvsem zaradi možnosti hitrega pisanja in preverjanja posameznih delov razvite kode. To je odtehtalo malo počasnejše izvajanje, še posebej, ker pri edinem počasnejšem delu — optimizaciji mrež, ni bil pogoj izvajanje v realnem času. Za izračun razporeditve točk je skrbela samostojna funkcija, rezultate pa smo lahko shranili in jih kasneje po potrebi hitro naložili brez ponovnega računanja.

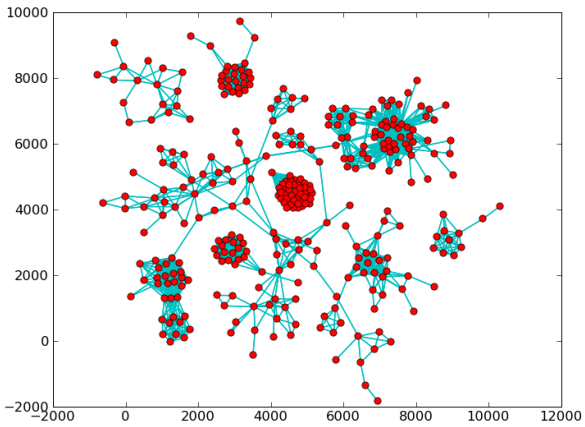
Pri obdelavi podatkov, prikazovanju vizualizacij in nekaterih pomožnih funkcijah, nam je bilo v veliko pomoč ogrodje za strojno učenje in odkrivanje znanj Orange [2]. Zasnovan je modularno in omogoča integracijo s pythonom,

kar pomeni preprosto in učinkovito kombiniranje uporabnikovih funkcij z že obstoječimi.

Za končni rezultat je seveda zelo pomemben način prikaza dobljenih mrež in možnost njihovega interaktivnega pregledovanja. Vendar pa so bili naši cilji predvsem priprava podatkov in vizualizacij ter raziskava mehanike njihovega delovanja. Za grafični prikaz in uporabniški vmesnik smo zato uporabili že obstoječe rešitve.

V nalogi smo uporabili različne načine predstavitve večnivojskih mrež, ki jih opisujemo spodaj.

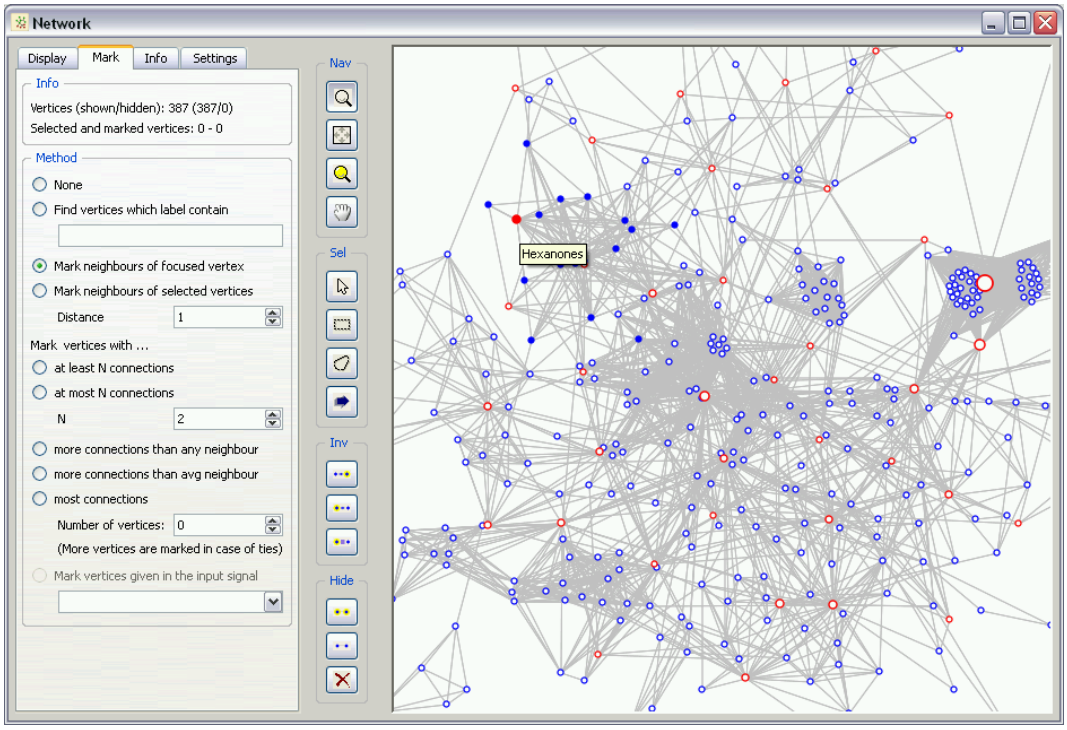
Slike . Najbolj preprost način prikaza mrež je, da točke in povezave narišemo na sliko. Tak način smo uporabljali predvsem za sprotno preverjanje rezultatov in risanje bolj enostavnih mrež sintetičnih podatkov. Neposredno iz pythona smo slike risali s pomočjo knjižnice Matplotlib (`http://matplotlib.sourceforge.net`). Primer prikazuje slika 3.5.



Slika 3.5: Slika mreže narisana s pomočjo knjižnice Matplotlib

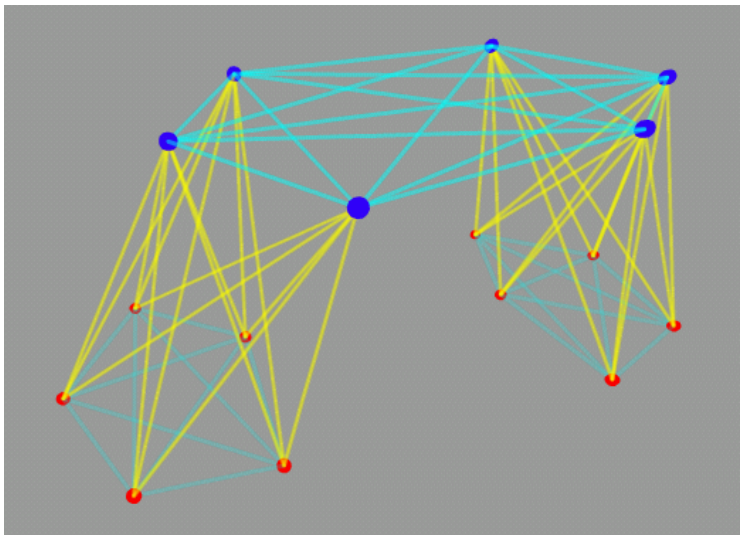
Interaktivni 2D prikaz . Za končnega uporabnika, ki želi vizualizacije čim bolje izkoristiti za preiskovanje podatkov, odkrivanje novih znanj in zanimivih lastnosti, je interaktiven prikaz bolj primeren od statičnih slik.

Priprava orodja, ki to omogoča, je zahtevno opravilo, saj je potrebno uravnotežiti preprostost in intuitivnost uporabe z dovolj velikim naborom funkcij in uporabnostjo. Na srečo smo v ta namen lahko uporabili že razviti modul Network (opisan v [10]) iz ogrodja Orange. Mreže je bilo potrebno le shraniti v ustrezni obliki, nakar jih v omenjenem modulu naložimo in nadalje raziskujemo (slika 3.6).



Slika 3.6: Orangeov gradnik Network omogoča interaktivno pregledovanje mrež

Interaktivni 3D prikaz . Sama narava obravnavanih vizualizacij — večnivojskih mrež, namiguje na prostorsko predstavitev. V treh dimenzijah lahko mreže postavimo v vzporedne ravnine in tako bolj jasno ločimo posamezne plasti. S tako predstavitvijo se nismo pretirano ukvarjali. Nekaj testnih vizualizacij je bilo pripravljenih samo v namen prikaza tudi te možnosti (slika 3.7).



Slika 3.7: 3D prikaz mreže v Flashu

Poglavje 4

Eksperimenti in rezultati

V diplomski nalogi razvite tehnike smo preverjali in ovrednotili v eksperimentih, ki smo jih izvedli na sintetičnih domenah in na podatkih s področja bioinformatike.

4.1 Sintetične eksperimentalne domene

Najprej na podlagi testov na sintetičnih podatkih spoznajmo značilnosti in prednosti opisanih metod ter preverimo njihovo delovanje.

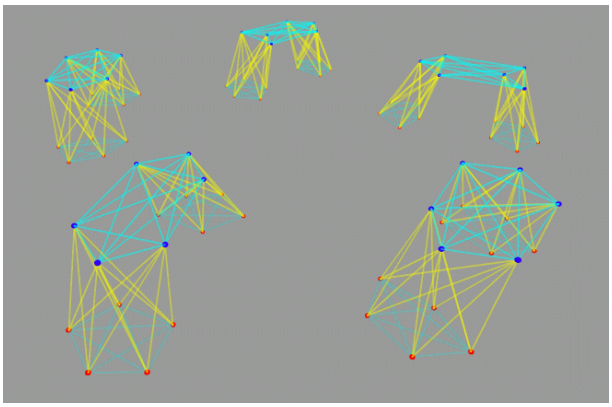
4.1.1 Podatki

Sestavili smo nekaj umetnih naborov podatkov, ki za razliko od dejanskih podatkov iz pravega sveta niso zelo kompleksni. Namenoma smo jih poizkušali sestaviti tako, da bi se na njih že lahko opazile lastnosti in zakonitosti, ki jih preverjamo, a hkrati ostanejo kar se da preprosti.

"Pot in cikel" (PC) je izredno majhna mreža, ki smo jo spoznali že v prejšnjem poglavju. Sestavljena je iz samo 10 točk. Eno plast predstavlja pot s 6. točkami (P_6), na drugi pa se nahaja cikel s 4. točkami (C_4). Med nivojema

je 6 povezav. Formalno so podatki opisani z enačbami (3.1) in prikazani na sliki 3.1.

"Več komponent" (KOMP0) je preprosta mreža, sestavljena iz večjega števila manjših komponent. Na prvem nivoju je 10 nepovezanih podgrafov K_5, na drugem nivoju 5 nepovezanih podgrafov K_6, vmes pa 150 povezav. Natančna struktura je opisana z enačbami (4.1) in grafično prikazana na sliki 4.1.



Slika 4.1: 3D prikaz podatkov "Več komponent" (KOMP0)

$$\begin{array}{lrcl} N_1: & V_1 & = & \{v_{1,1}, v_{1,2}, \ldots, v_{1,50}\} \\ & E_1 & = & \{\{v_{1,i}, v_{1,j}\} \mid i \neq j,\ \lceil \frac{i}{5} \rceil = \lceil \frac{j}{5} \rceil\} \\ N_2: & V_2 & = & \{v_{2,1}, v_{2,2}, \ldots, v_{2,30}\} \\ & E_2 & = & \{\{v_{2,i}, v_{2,j}\} \mid i \neq j,\ \lceil \frac{i}{6} \rceil = \lceil \frac{j}{6} \rceil\} \\ N_{12}: & E_{12} & = & \{\{v_{1,i}, v_{2,j}\} \mid \lceil \frac{i}{5} \rceil = \lceil \frac{j}{3} \rceil\} \end{array} \tag{4.1}$$

Zgornji podatki so zelo lepo sestavljeni, zato smo jih za uporabo v nekaterih testih morali malo "pokvariti". To smo naredili na dva načina in tako dobili podatke KOMP1 ter KOMP2.

KOMP1 Povezave med nivoji (E_{12}) naključno prevežemo s postopkom, opisanim v 3.2.2. Taki podatki so veliko bolj nekompatibilni, saj se medni-

vojske povezave ne ujemajo več s strukturama nivojev N_1 in N_2. Vmesni nivo N_{12} v tem primeru strukture sploh nima, saj je popolnoma neurejen.

KOMP2 Povezave med nivoji definiramo drugače. Tudi v tem primeru poslabšamo kompatibilnost podatkov, vendar pa tukaj nivo N_{12} ni neurejen. Ima zelo izrazito strukturo, ki pa je sorodna samo ureditvi nivoja N_2 in popolno nasprotje strukture v N_1. Definicija množice E_{12} je podana z enačbo (4.2).

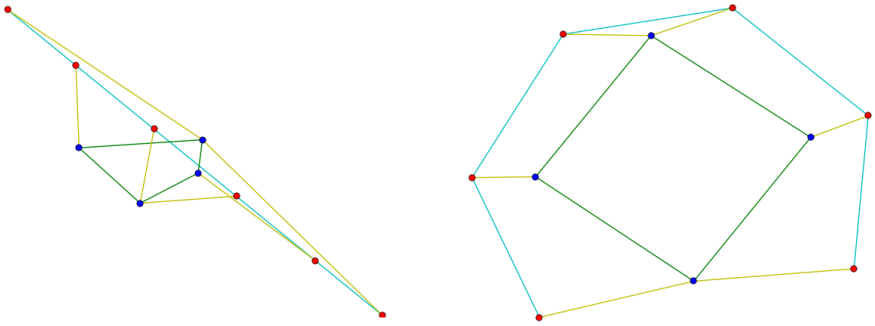
$$E_{12} = \{\{v_{1,i}, v_{2,j}\} \mid i \bmod 5 = \lceil \frac{j}{6} \rceil\} \tag{4.2}$$

4.1.2 Primerjava postopne in hkratne optimizacije

V poglavju 3.1 smo spoznali postopka za postopno in hkratno optimizacijo. Predpostavka je bila, da ima postopna optimizacija pomanjkljivosti, ki jih hkratna optimizacija odpravi in zato pričakujemo, da dela bolje. To predpostavko smo tudi v praksi preverili na dveh vrstah podatkov, pripravljenih posebej v ta namen.

1. Podatki PC prikazujejo preprost primer, kjer zaradi interakcije med nivojema hkratni optimizaciji razporeditev točk uspe veliko bolje. Prvi in drugi nivo sama po sebi nista zelo zanimiva in ju lahko dobro prikažemo s skoraj kakršnokoli metodo. Vendar pa se zaradi mednivojskih relacij da odkriti dodatno podobnost med prvo ($v_{1,1}$) in zadnjo ($v_{1,6}$) točko poti (prvi nivo). Obe sta namreč povezani z isto točko iz drugega nivoja ($v_{2,1}$). Take informacije želimo prepoznati tudi vizualno.

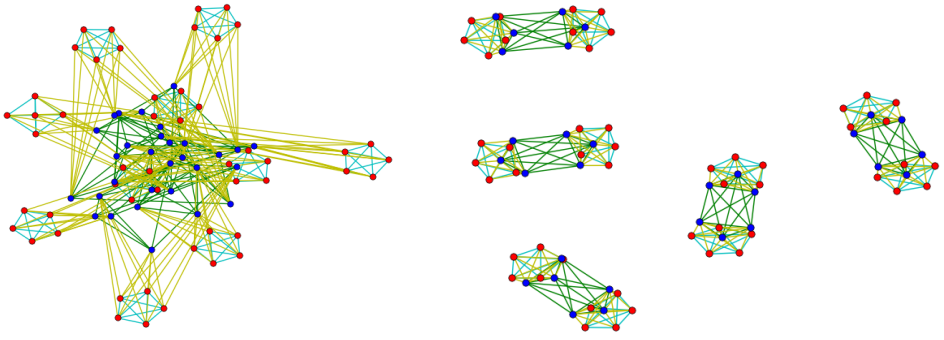
   Kot vidimo na sliki 4.2, to postopni optimizaciji ne uspe, saj točk drugega nivoja ni možno razporediti tako, da bi se $v_{2,1}$ znašla v okolici obeh svojih sosed. Hkratna optimizacija pa vpliv drugega nivoja izkoristi, da "zvije" pot. Sorodne točke so si tako lahko bližje in vizualizacija zaradi tega boljša.



(a) Postopna (b) Hkratna

Slika 4.2: Dva načina optimizacije podatkov PC

2. Podatki KOMP0 so uporabljeni za potrditev intuitivne teze, da postopna optimizacija ni uporabna pri podatkih z več nepovezanimi komponentami. V takem primeru se med prvo fazo postopne optimizacije posamezne komponente razporedijo po ravnini naključno oz. odvisno od začetnih položajev točk. Ko se začne druga faza, je zato že prepozno in dobra optimizacija ni več možna. Hkratna optimizacija problem reši, saj že med postopkom povleče ustrezne komponente skupaj. Rezultata obeh postopkov sta prikazana na sliki 4.3.



(a) Postopna (b) Hkratna

Slika 4.3: Dva načina optimizacije podatkov KOMP0

V obeh primerih je bila razlika med postopkoma očitna in lahko rečemo, da so se naši sumi izkazali za upravičene. Postopna optimizacija pri nekaterih mrežah res odpove in ker so motivi iz testnih podatkov pogosti pri skoraj vseh večjih mrežah, se na ta postopek ne gre zanašati. Hkratna optimizacija se je po drugi strani izkazala v obeh primerih, kar nam da upanje, da bo uporabna tudi pri optimizaciji večjih in kompleksnejših mrež.

4.1.3 Ocene nivojev v odvisnosti od moči mednivojskih povezav

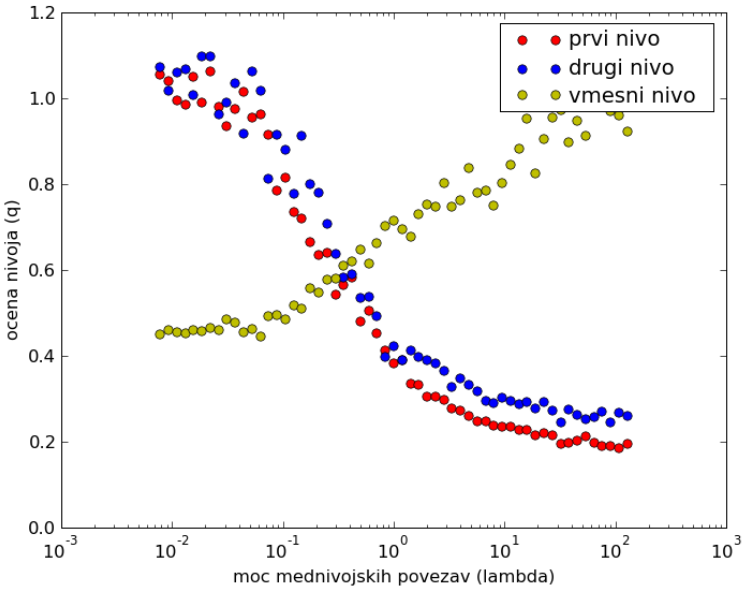
Pri postopku hkratne optimizacije, opisanem v 3.1, imamo parameter λ, ki vpliva na moč mednivojskih povezav in s tem uravnava prioriteto optimizacije posameznih nivojev ali vmesnega nivoja. Pokazati smo želeli, da z njim tudi dejansko lahko izbiramo med optimizacijo, ki ima poudarek na posameznih nivojih in tako, kjer so bolj pomembne mednivojske povezave.

Grafa na sliki 4.4 prikazujeta ocene posameznih nivojev v odvisnosti od λ za podatke KOMP1 in KOMP2, ki so zaradi manjše kompatibilnosti na vrednost parametra bolj občutljivi. Na slikah 4.5 in 4.6 pa so posledice različnih vrednosti λ tudi grafično prikazane za podatke PC in KOMP2.

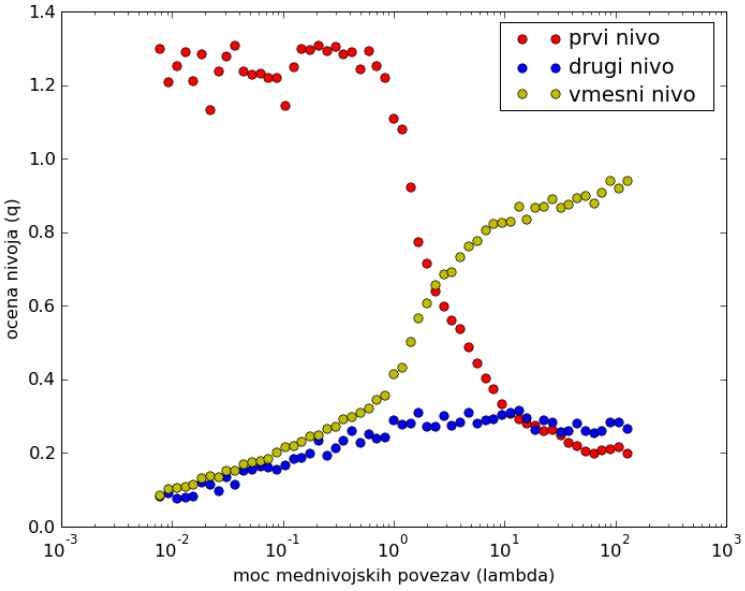
4.1.4 Kompatibilnost mrež v praksi

Že pri drugih preizkusih smo opažali, da se nekatere mreže bolje hkratno optimizirajo, nekatere slabše. V prejšnjem razdelku, kjer smo preučevali vpliv moči mednivojskih povezav (λ) na ocene posameznih nivojev, smo opazili razlike v grafih, ki prikazujejo to odvisnost za različne podatke.

Te razlike smo pripisali različnim stopnjam kompatibilnosti, ki so bile zaradi preprostosti podatkov razvidne že iz njihovih definicij (v ta namen so bili tudi tako sestavljeni). V splošnih mrežah, kjer strukturo težko razumemo, pa bi bila taka utemeljitev veliko težja. Zato želimo preprostejšo mero, ki izmeri kompatibilnost podatkov in jo poda numerično. Taka ocenjevalna funkcija je

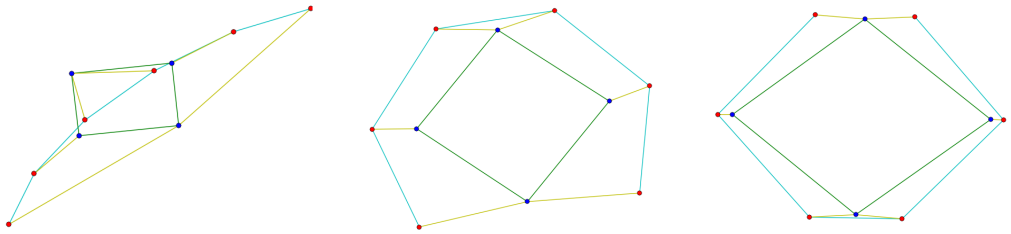


(a) KOMP1



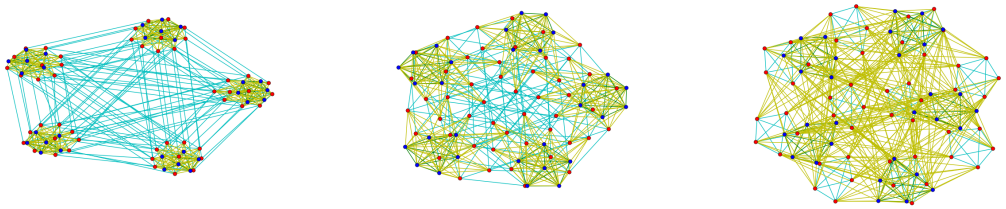
(b) KOMP2

Slika 4.4: Ocene nivojev v odvisnosti od λ za podatke KOMP1 in KOMP2



(a) $\lambda = \frac{1}{20}$ (b) $\lambda = 1$ (c) $\lambda = 20$

Slika 4.5: Hkratna optimizacija podatkov PC pri različnih vrednostih λ



(a) $\lambda = \frac{1}{20}$ (b) $\lambda = 1$ (c) $\lambda = 20$

Slika 4.6: Hkratna optimizacija podatkov KOMP2 pri različnih vrednostih λ

bila podana v 3.2.3, na tem mestu pa jo želimo v praksi preveriti in ugotoviti, ali se njeni rezultati ujemajo s pričakovanimi.

Testirali smo mreže podatkov KOMP0, KOMP1 in KOMP2. Ker so rezultati optimizacije v neki meri odvisni od naključja (zaradi naključne začetne razporeditve), smo vse poizkuse ponovili 100-krat in podali povprečno vrednost rezultata in interval zaupanja velikosti $1,96$ standardnega odklona.

	KOMP0	KOMP1	KOMP2
k_1	1.241 ± 0.178	0.475 ± 0.070	0.174 ± 0.034
k_2	1.152 ± 0.253	0.589 ± 0.124	0.950 ± 0.218

Tabela 4.1: Ocene kompatibilnosti mrež KOMP0, KOMP1 in KOMP2

Dobljeni rezultati so prikazani na tabeli 4.1. Zelo kompatibilni podatki

KOMP0 se tako izkažejo tudi v našem testu. Naključno spremenjeni podatki KOMP1 so veliko manj kompatibilni in sicer z ozirom na prvi in drugi nivo približno enako. Podatki KOMP2 pa izkazujejo izrazito asimetrijo — prvi nivo je zelo nekompatibilen s celotno mrežo, medtem ko se drugi nivo z njo dobro ujema.

4.2 Genske mreže

V nalogi razvite tehnike smo preverili tudi na realnih (nesintetičnih) podatkih s področja bioinformatike. Tam bi tako orodje lahko bistveno pomagalo strokovnjakom — biologom pri iskanju novih funkcijskih povezav med geni. Na področju bioinformatike je na voljo namreč veliko različnih baz podatkov, ki opisujejo različne entitete in kjer je moč podatke v različnih bazah na izbran način med sabo povezati. Uporabnost razvitih tehnik smo prikazali tako, da smo iz prosto dostopnih podatkov, ob definiciji smiselnih relacij, sestavili dve večnivojski mreži. Ti smo nato optimizirali in pripravili za interaktivno pregledovanje z orodjem Orange.

4.2.1 Opis domene

Dictyostelium discoideum

Ameba *D. discoideum* je socialni enoceličar, ki živi v prsti, a se ob pomanjkanju hrane združi v mnogocelično združbo, ki tvori steblo in spore. Za biologe in genetike je zanimiva predvsem s stališča opazovanja procesa razvoja in diferenciacije celic, kot tudi s stališča medcelične komunikacije. Njen genski zapis je sestavljen iz okoli 12,000 genov. V svetu je okoli 200 laboratorijev, ki to amebo in njene biogenetične ter biokemične procese intenzivno proučujejo. V namene testiranja v pričujoči diplomski nalogi smo iz osnovne spletne strani o tem organizmu pridobili listo genov in z njimi povezanih člankov, nato iz baze podatkov PubMed pridobili oznake MeSH za te članke, ter v naših eksperimen-

tih skušali povezati podobnostno mrežo genov ter podobnostno mrežo oznak MeSH. Podobnost med geni smo ocenili s podobnostjo vektorjev oznak ali pa na podlagi časovnega profila genske ekspresije za nespremenjeni organizem (angl. *wild type*).

Ontologija MeSH

MeSH (Medical Subject Headings) je ontologija oznak in pojmov s področja medicine in sorodnih ved, ki jo nadzira in ureja ameriška nacionalna medicinska knjižnica (NLM). Ker jo vodi skupina strokovnjakov, so vnosi skrbno izbrani in organizirani v celoto.

Baza biomedicinskih člankov PubMed

PubMed je indeksna baza v glavnem revijskih publikacij s področja biomedicine in naravoslovnih ved. Za delo, ki ga predstavljamo tu, je pomembno, da kuratorji vsakemu članku iz baze PubMed priredijo okoli 10-15 oznak, ki čim bolje opišejo obravnavano tematiko. Skrbno dodeljevanje in standardiziranost oznak omogočata učinkovito indeksiranje baze objav in iskanje po njej.

4.2.2 Podatki o objavah povezanih z geni *D. discoideum*

Na spletni strani o amebi *D. discoideum* (`http://dictybase.org`) je na voljo baza genov organizma *D. discoideum* s podatki o objavah, v katerih so bili geni omenjeni. Informacije v njej so za raziskovalce zanimive in uporabne pri pregledovanju obstoječega dela. Z njihovo pomočjo je možno tudi odkriti še ne dovolj obdelane teme in si s tem pomagati pri načrtovanju novih eksperimentov. Ker nas navadno pri delu s članki najbolj zanimajo teme, o katerih govorijo, smo si kot dodatno bazo podatkov izbrali ontologijo MeSH. S pomočjo baze člankov PubMed smo za vsak članek pridobili nekaj značilnih oznak MeSH in tako oba nabora podatkov tudi povezali.

Za predstavitev podatkov z mrežami je bilo potrebno najprej izbrati, kaj bodo predstavljale točke in povezave. Odločili smo se za mrežo genov na prvem nivoju in oznak MeSH na drugem. Sorodnosti med oznakami MeSH smo definirali s pomočjo Jaccardovega indeksa (enačba 4.3). Vsaki oznaki smo priredili množico objav, ki so bile z njim opisane. Če sta si bili dve množici podobni (vrednost $J(A, B)$ večja od določenega praga), sta bili ustrezni MeSH oznaki povezani.

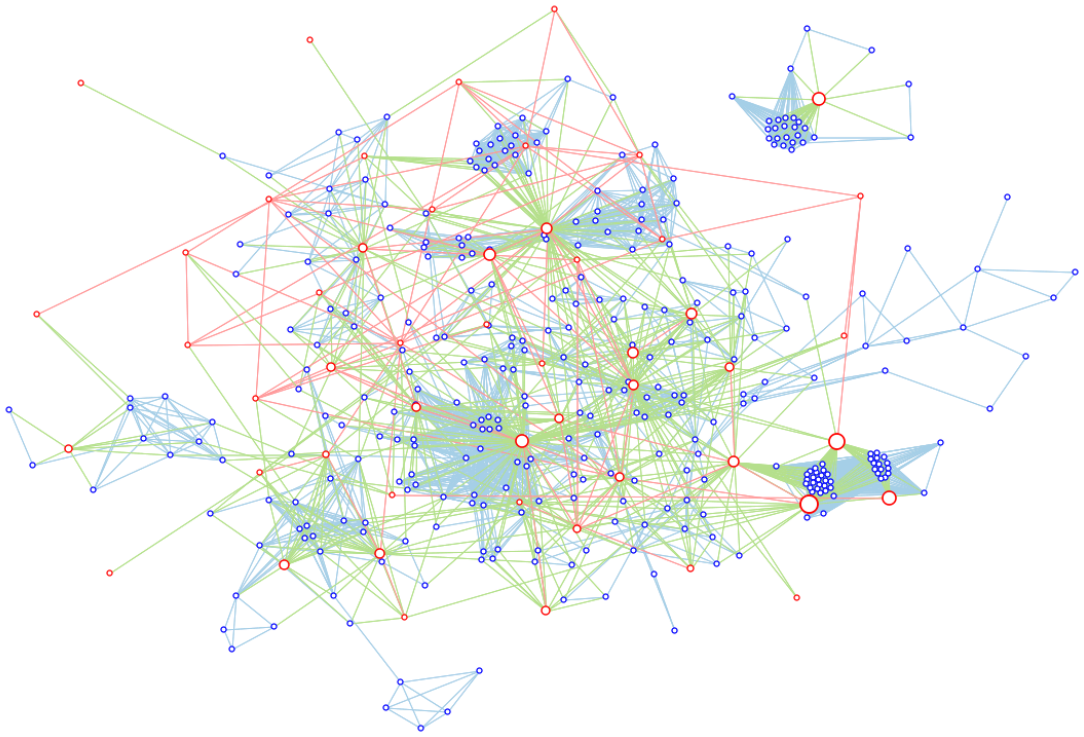
$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{4.3}$$

Pri povezovanju genov med sabo in genov z oznakami MeSH smo si pomagali s tehnikami iz področja odkrivanja znanj iz besedil (ang. *text mining*) [13]. Vsak gen smo obravnavali kot besedilo, sestavljeno iz množice oznak MeSH (besed), ki opisujejo članke, v katerih se gen pojavlja. Izračunali smo numerične uteži tf-idf (term frequency — inverse document frequency) za vsak par gen–oznaka po enačbah (4.4). Te smo normalizirali in uporabili za opis genov z vektorji ter kot osnovo za relacijo med nivojema. Povezave med geni pa so temeljile na kosinusni podobnosti (ang. *cosine similarity*).

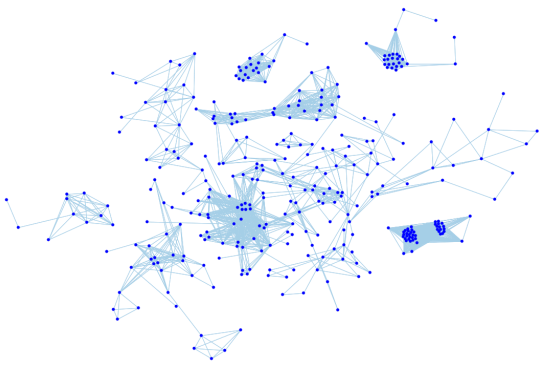
$$\begin{aligned} tf_{i,j} &= \text{število pojavitev izraza } i \text{ v besedilu } j \\ df_i &= \text{število besedil, ki vsebujejo izraz } i \\ idf_i &= \log_2(\tfrac{N}{df_i}) \quad \text{, kjer je } N \text{ število vseh besedil} \\ tf\text{-}idf_{i,j} &= tf_{i,j} * idf_i \end{aligned} \tag{4.4}$$

Na sliki 4.7 je prikazana dobljena mreža in posamezni mreži prvega in drugega nivoja. Opazimo lahko, da prvi nivo sestavlja 7 nepovezanih komponent. Tudi na drugem nivoju je veliko samostojnih točk in parov. Celotna večnivojska mreža pa je veliko bolj povezana in jo sestavljata samo 2 komponenti. Iz nje lahko razberemo informacije, ki jih iz posameznih mrež ne bi mogli.

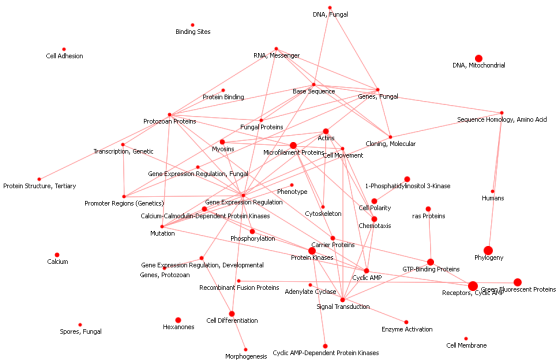
Nekaj zanimivih delov mreže je dodatno izpostavljenih na sliki 4.8. Navadno gre za soseščine večje narisanih oznak MeSH, kar dodatno osmisli naš



(a)



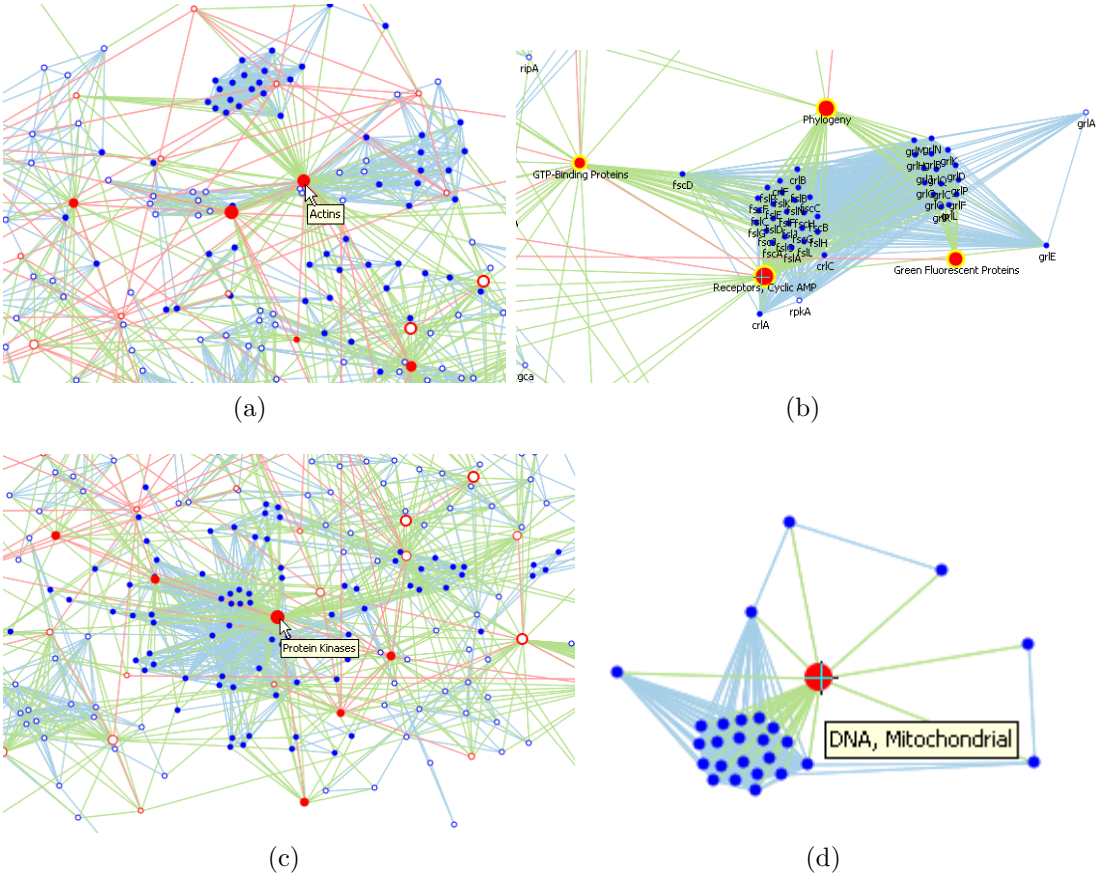
(b)



(c)

Slika 4.7: Mreža na osnovi podatkov o objavah (a) in posamezno izrisana prvi (b) in drugi (c) nivo

način ocenjevanja točk, ki je bil podlaga za določitev velikosti. Poleg grafične potrditve smo statistično značilnost preizkusili tudi s permutacijskim testom. Ta je hipotezo o pomembnosti točk potrdil, saj se je izkazalo, da imajo vse bolje ocenjene in zanimive točke p-vrednost 0 (dobra ocena ni bila plod naključja, ampak zakonitosti v podatkih).



(a) (b)

(c) (d)

Slika 4.8: Zanimivi deli mreže iz slike 4.7a

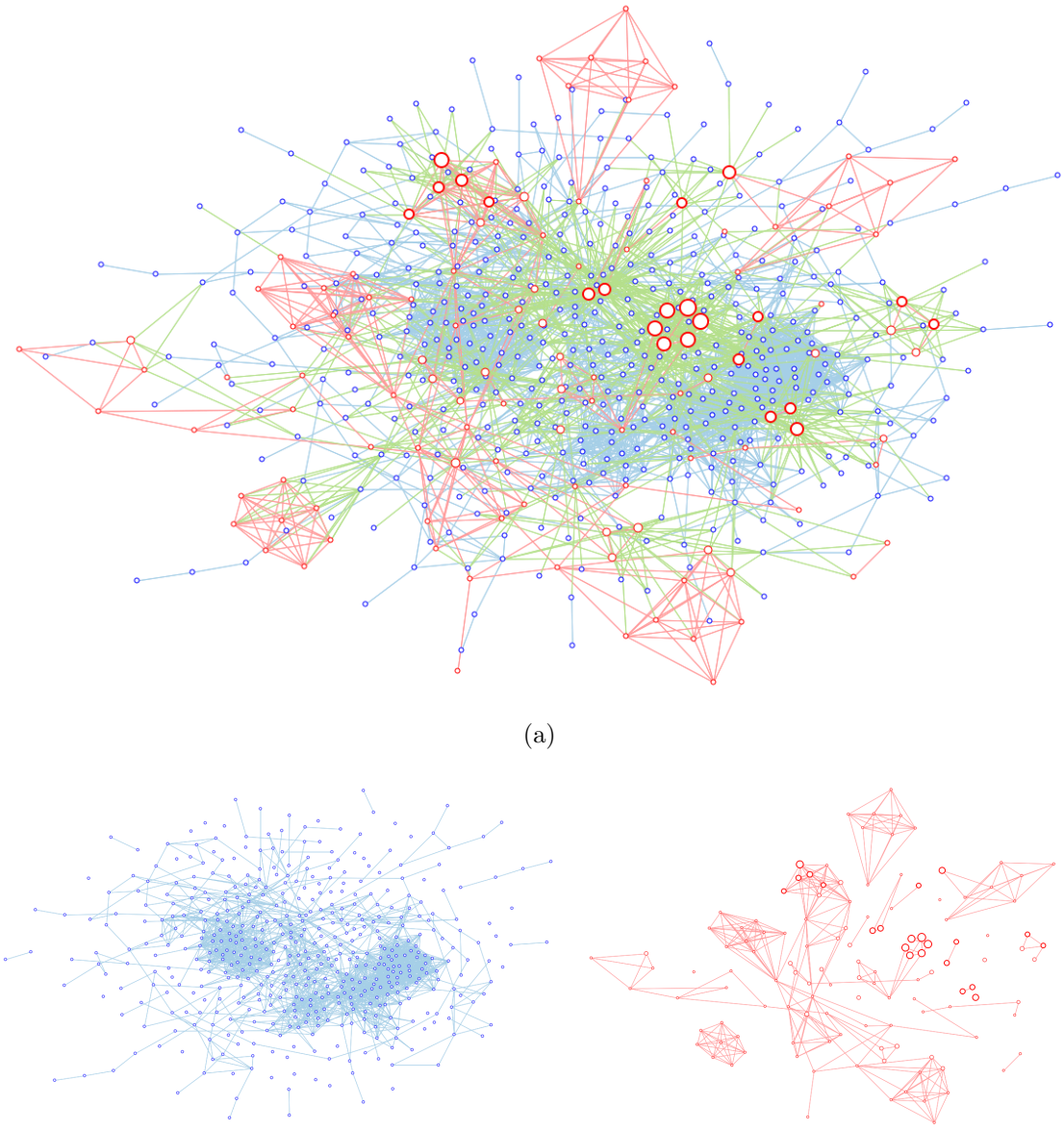
4.2.3 Soizrazne mreže

Druga vrsta večnivojske mreže je bila tudi sestavljena iz mreže genov na prvem nivoju in mreže oznak MeSH na drugem, vendar pa so bile relacije definirane drugače. V prejšnjem primeru smo se posvečali bolj metapodatkom in gene

preiskovali v kontekstu njihove citiranosti. Lahko pa seveda iščemo znanja tudi v novih eksperimentalnih rezultatih in tako neposredno pomagamo pri postavljanju hipotez in njihovem preverjanju.

V našem primeru smo uporabili podatke o izraženosti genov [12], izmerjene s skrbno načrtovanimi laboratorijskimi poizkusi. Take podatke se pogosto predstavlja z mrežami sočasno izraženih genov (ang. *co-expression networks*). Povezave med točkami so določene glede na korelacijo profilov izraženosti. Soizrazni mreži, ki je predstavljala prvi nivo, smo v našem preizkusu dodali še mrežo oznak MeSH, ki je predstavljala drugi nivo.

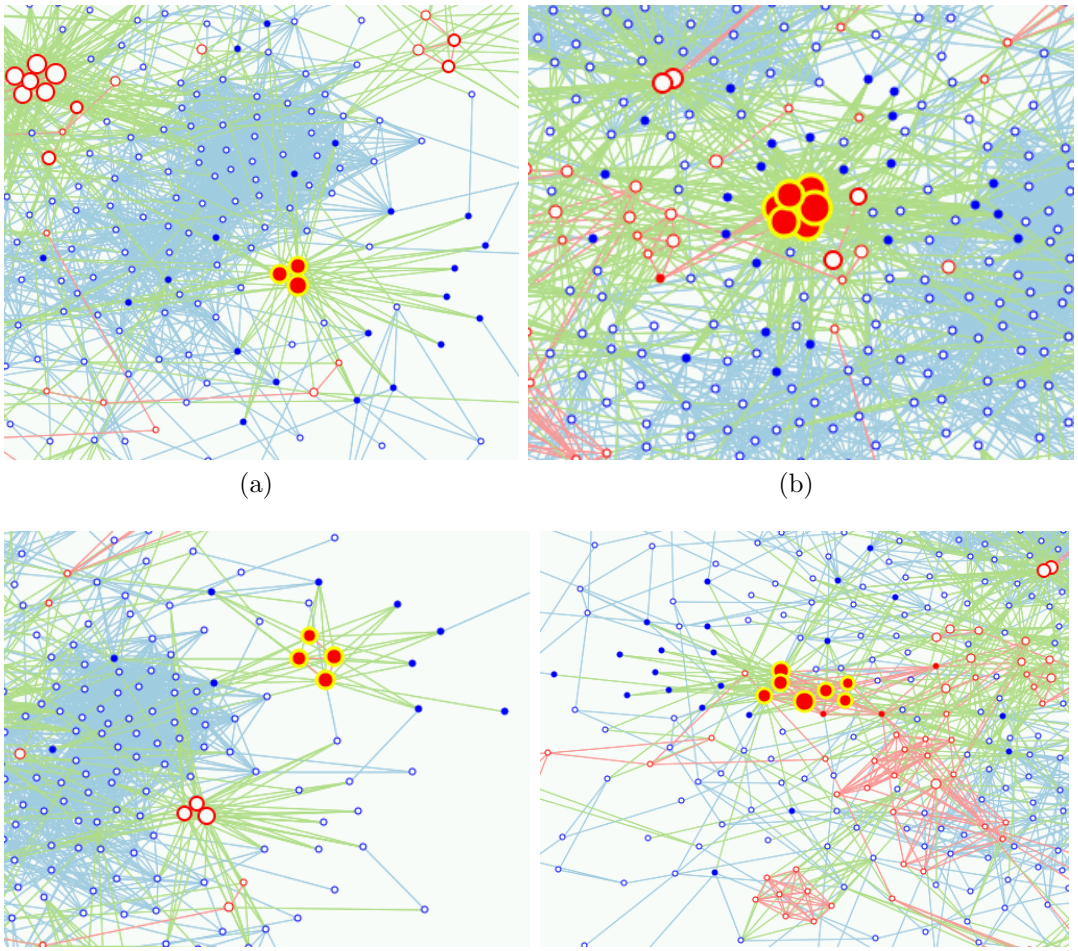
Dobljena večnivojska mreža je bila sestavljena iz velike povezane komponente in večjega števila nepovezanih točk in parov točk. Takemu rezultatu se je bilo težko izogniti, saj bi v primeru nižjega praga za določitev povezav mreža sicer imela manj komponent, a bi imela preveč povezav. Zato smo se raje odločili za večjo preglednost ob manjšem številu povezav in prikazali samo veliko komponento mreže (slika 4.9). V primerjavi s prejšnjo mrežo s podatki o objavah je tukaj manj skupin genov s skupno značilno oznako MeSH, kar nakazuje manjše ujemanje obeh vrst podatkov. To ni bilo nepričakovano, saj sta podlagi za strukturo obeh nivojev (sočasna izraženost genov in sorodnost oznak MeSH) zelo različni. Kljub temu bi lahko nekdo, ki razume prikazane podatke opazil v mreži informativne motive. Nekaj zanimivejših delov, ki jih prepozna tudi laik, je prikazano na sliki 4.10.



(a)

(b) (c)

Slika 4.9: Soizrazna mreža (a) in posamezno izrisana prvi (b) in drugi (c) nivo



(a) (b)

(c) (d)

Slika 4.10: Zanimivi deli mreže iz slike 4.9a

Poglavje 5

Zaključek

Mreže so uporabno orodje za vizualizacijo podatkov. Teorija, ki jih preučuje, se je v zadnjem času zelo razvila, mrežne vizualizacije pa so se tudi v praksi uveljavile na številnih področjih. V bližnji prihodnosti ni pričakovati, da bi se njihova uporaba opustila, kvečjemu nasprotno — našli se bodo novi načini uporabe in predlogi, kako jih v te namene prilagoditi in nadgraditi.

V tem delu smo preučili možnost uporabe večnivojskih mrež za predstavitev relacijskih podatkovnih baz. Te baze vsebujejo bolj ali manj sorodne podatke, zapisane v več tabelah in povezane z različnimi relacijami. Primerke različnih tabel smo predstavili kot točke različnih nivojev in jih med sabo povezali glede na izbrano podobnostno relacijo. S tem smo dobili abstrakten opis večnivojske mreže. Predlagali in opisali smo postopek, ki optimizira predstavitev (projekcijo) takih mrež, kar nam omogoča, da jih všečno narišemo in prikažemo. Prav tako smo podali nekaj objektivnih kriterijev za ocenjevanje dobljenih vizualizacij. Razvite metode smo tudi v praksi preizkusili s simulacijami na sintetičnih podatkih in preverili možnost konkretne uporabe na podatkih s področja bioinformatike. Simulacije so potrdile želene lastnosti, primera dveh mrež na osnovi genskih podatkov pa sta se izkazala za praktični vizualizaciji in opravičila tako predstavitev podatkov.

V prihodnosti bi bilo mogoče predstavljene tehnike še dodatno nadgraditi

in preizkusiti. Tukaj smo se omejili na dvonivojske mreže, kar se seveda da posplošiti na več nivojev. Uporabljen algoritem za razporeditev točk deluje na principu privlačnih in odbojnih sil. Veljalo bi preizkusiti tudi druge pristope in preveriti, ali se jih da prilagoditi za delo z večnivojskimi mrežami. Obdelane metode lahko, namesto kot tehnike vizualizacije, obravnavamo tudi s stališča odkrivanja znanj. V tem okviru bi bilo zanimivo oceniti napovedno moč metod in jih primerjati z ostalimi pristopi odkrivanja znanj iz relacijskih podatkovnih baz. Nenazadnje pa lahko za tako vrsto vizualizacije poiščemo tudi nove možnosti uporabe. Namesto da prikazujemo samo podatke, lahko recimo v isto vizualizacijo dodamo rezultate metod strojnega učenja v obliki drugače označenih točk in povezav.

Dodatek A

Algoritem Fruchterman–Reingold

Izvirna psevdokoda algoritma Fruchterman–Reingold, kot je bila zapisana v [3].

```
area := W * L; { W and L are the width and length of the frame}
G := (V, E); { the vertices are assigned random initial positions}
k := sqrt( area / |V| );

function f_a(z) := begin return z^2/k end ;
function f_r(z) := begin return k^2/z end ;

for i := 1 to iterations do begin
   { calculate repulsive forces}
   for v in V do begin
      { each vertex has two vectors: .pos and .disp }
      v.disp := 0;
      for u in V do
         if (u != v) then begin
            { d is short hand for the difference}
            { vector between the positions of the two vertices )
            d  := v.pos - u.pos;
            v.disp := v.disp + ( d /|d|) * f_r(|d|)
```

```
      end
   end
   { calculate attractive forces }
   for e in E do begin
      { each edge is an ordered pair of vertices .v and .u }
      d  := e.v.pos { e.u.pos
      e.v.disp := e.v.disp - ( d/|d|) * f_a(|d|);
      e.u. disp := e.u.disp + ( d /|d|) * f_a(|d|)
   end
   { limit the maximum displacement to the temperature t }
   { and then prevent from being displaced outside frame}
   for v in V do begin
      v.pos := v.pos + (v. disp/ |v.disp|) * min (v.disp, t);
      v.pos.x := min(W/2, max(-W/2, v.pos.x));
      v.pos.y := min(L/2, max({L/2, v.pos.y))
   end
   { reduce the temperature as the layout}
   { approaches a better configuration }
   t := cool(t)
end
```

Dodatek B

Algoritem za hkratno optimizacijo

Spremenjen algoritem za hkratno optimizacijo dveh nivojev mrež.

```
area := W * L { W and L are the width and length of the frame}
G := (V, E) { the vertices are assigned random initial positions}
k1 := sqrt( area / |V1| )
k2 := sqrt( area / |V2| )
k12 := lambda * (k1+k2)/2 {lambda regulates the inter-layer forces}

function f_a(z,x) := begin return z^2/x end ;
function f_r(z,x) := begin return x^2/z end ;

for i := 1 to iterations do begin
   { repulsive forces on layer 1 }
   for v in V1 do begin
      v.disp := 0
      for u in V1 do
         if (u != v) then begin
            d  := v.pos - u.pos
            v.disp := v.disp + ( d /|d|) * f_r(|d|,k1)
         end
   end
```

```
   { repulsive forces on layer 2 }
   for v in V2 do begin
      v.disp := 0
      for u in V2 do
         if (u != v) then begin
            d  := v.pos - u.pos
            v.disp := v.disp + ( d /|d|) * f_r(|d|,k2)
         end
   end
   { attractive forces on layer 1}
   for e in E do begin
      d  := e.v.pos { e.u.pos
      e.v.disp := e.v.disp - ( d/|d|) * f_a(|d|,k1)
      e.u. disp := e.u.disp + ( d /|d|) * f_a(|d|,k1)
   end
   { attractive forces on layer 2}
   for e in E do begin
      d  := e.v.pos { e.u.pos
      e.v.disp := e.v.disp - ( d/|d|) * f_a(|d|,k2)
      e.u. disp := e.u.disp + ( d /|d|) * f_a(|d|,k2)
   end
   { attractive inter-layer forces }
   for e in E do begin
      d  := e.v.pos { e.u.pos
      e.v.disp := e.v.disp - ( d/|d|) * f_a(|d|,k12)
      e.u. disp := e.u.disp + ( d /|d|) * f_a(|d|,k12)
   end
   { limit the displacement with temperature and frame size}
   for v in V do begin
      v.pos := v.pos + (v. disp/ |v.disp|) * min (v.disp, t)
      v.pos.x := min(W/2, max(-W/2, v.pos.x))
      v.pos.y := min(L/2, max({L/2, v.pos.y))
   end
   { reduce the temperature }
   t := cool(t)
end
```

Slike

Literatura

[1] A. Blum in T. Mitchell: *Combining labeled and unlabeled data with co-training.* V *COLT' 98: Proceedings of the eleventh annual conference on Computational learning theory*, str. 92–100, New York, NY, USA, 1998. ACM.

[2] J. Demšar, B. Zupan in G. Leban: *Orange: From experimental machine learning to interactive data mining.* White paper (`www.ailab.si/orange`), Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, Ljubljana, 2004.

[3] T.M.J. Fruchterman in E.M. Reingold: *Graph drawing by force-directed placement.* Softw. Pract. Exper., 21(11):1129–1164, 1991.

[4] M. Greenacre: *Correspondence analysis in practice.* Chapman & Hall/CRC, 2. izd., 2007.

[5] Y. Koren: *On spectral graph drawing.* V *COCOON 03*, del 2697 iz LNCS, str. 496–508. Springer-Verlag, 2003.

[6] D.P. Lewis, T. Jebara in W.S. Noble: *Support vector machine learning from heterogeneous data: an empirical analysis using protein sequence and structure.* Bioinformatics, 22(22):2753–2760, 2006.

[7] S.C. Madeira in A.L. Oliveira: *Biclustering algorithms for biological data analysis: A survey.* IEEE/ACM Trans. Comput. Biol. Bioinformatics, 1(1):24–45, 2004.

[8] W. de Nooy, A. Mrvar in V. Batagelj: *Exploratory Social Network Analysis with Pajek*. Cambridge University Press, New York, NY, USA, 2005.

[9] P. Sprent in N.C. Smeeton: *Applied Nonparametric Statistical Methods, Fourth Edition*, pogl. 2, str. 23–44. Chapman & Hall/CRC, 2006.

[10] M. Štajdohar: *Odkrivanje zakonitosti iz mrež s pomočjo vizualizacije*. Diplomsko delo, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, Ljubljana, 2007.

[11] A. Tanay, R. Sharan in R. Shamir: *Handbook of Computational Molecular Biology (Chapman & All/Crc Computer and Information Science Series)*, pogl. 26, str. 26.1–26.17. Chapman & Hall/CRC, 2005.

[12] N. Van Driessche, C. Shaw, M. Katoh, T. Morio, R. Sucgang, M. Ibarra, H. Kuwayama, T. Saito, H. Urushihara, M. Maeda, I. Takeuchi, H. Ochiai, W. Eaton, J. Tollett, J. Halter, A. Kuspa, Y. Tanaka in G. Shaulsky: *A transcriptional profile of multicellular development in Dictyostelium discoideum*. Development, 129(7):1543–1552, 2002.

[13] S. Weiss, N. Indurkhya, T. Zhang in F. Damerau: *Text Mining: Predictive Methods for Analyzing Unstructured Information*. SpringerVerlag, 2004.

Izjava

Izjavljam, da sem diplomsko nalogo izdelal samostojno pod vodstvom mentorja izr. prof. dr. Blaža Zupana. Izkazano pomoč drugih sodelavcev sem v celoti navedel v zahvali.

Ljubljana, 8. september 2008 Lan Žagar