UNIVERZITET U BEOGRADU

MATEMATIČKI FAKULTET

Milana Grbić

RAČUNARSKE METODE PARTICIONISANJA I GRUPISANJA U BIOLOŠKIM MREŽAMA

doktorska disertacija

Beograd, 2020

UNIVERSITY OF BELGRADE
FACULTY OF MATHEMATICS

Milana Grbić

COMPUTATIONAL METHODS FOR PARTITIONING AND GROUPING IN BIOLOGICAL NETWORKS

Doctoral Dissertation

Belgrade, 2020

Podaci o mentoru i članovima komisije

Mentor

dr Gordana Pavlović-Lažetić, redovni profesor, Matematički fakultet, Univerzitet u Beogradu

Članovi komisije

dr Vladimir Filipović, redovni profesor, Matematički fakultet, Univerzitet u Beogradu

dr Aleksandar Kartelj, docent, Matematički fakultet, Univerzitet u Beogradu

dr Dragan Matić, vanredni profesor, Prirodno - matematički fakultet, Univerzitet u Banjoj Luci

dr Branislava Gemović, naučni saradnik, Institut za nuklearne nauke Vinča, Univerzitet u Beogradu

Datum odbrane:

Podaci o doktorskoj disertaciji

Naslov doktorske disertacije
Računarske metode particionisanja i grupisanja u biološkim mrežama

Rezime
U ovoj disertaciji se istražuju aktuelni problemi bioinformatike i računarske biologije i metode za njihovo rješavanje. Razmatrane su metode za rješavanje sljedećih problema: particionisanje rijetkih bioloških mreža na *k-plex* podmreže, predviđanje uloge metabolita u metaboličkim reakcijama, particionisanje bioloških mreža na visoko povezane komponente i problem identifikacije značajnih grupa proteina dodavanjem novih grana u težinsku mrežu proteinskih interakcija. Navedeni problemi imaju teorijski značaj u oblastima mašinskog učenja i optimizacije, ali i praktičnu primjenu u biološkim istraživanjima. Stoga se, pored rješavanja navedenih problema sa računarskog aspekta, u disertaciji istražuje dalja primjena dobijenih rezultata u oblastima biologije i biohemije, kao i integracija rezultata unutar postojećih bioinformatičkih alata.

Problem predviđanja uloge metabolita u metaboličkim reakcijama je riješen prediktivnom metodom mašinskog učenja zasnovanom na uslovnim slučajnim poljima, dok su za rješavanje preostala tri problema razvijeni algoritmi zasnovani na metodi promjenljivih okolina. Za rješavanje problema identifikacije značajnih grupa proteina dodavanjem novih grana u težinsku mrežu proteinskih interakcija, metoda promjenljivih okolina predstavlja samo prvu fazu predloženog rješenja, a u drugoj i trećoj fazi metode vršena je integracija sa dodatnim biološkim informacijama i bioinformatičkim alatima.

Predložene računarske metode particionisanja i grupisanja u biološkim mrežama na nov način potvrđuju postojeće i dovode do otkrivanja novih informacija o biološkim elementima i vezama između njih. Rješavanjem navedenih problema i tumačenjem dobijenih rješenja u ovom radu dat je naučni doprinos naučnoj oblasti računarstva i informatike, a posebno užim naučnim oblastima bioinformatike i računarske biologije.

Ključne riječi
kombinatorna optimizacija, metoda promjenljivih okolina, uslovna slučajna polja, biološke mreže, protein-protein interakcije, *k-plex*, visoko povezane komponente

Naučna oblast
Računarstvo

Uža naučna oblast
Bioinformatika

UDK broj
(519.179.2+004.9):577(043.3)

Dissertation Data

Doctoral dissertation title
Computational methods for partitioning and grouping in biological networks

Abstract
In this dissertation some actual problems of bioinformatics and computational biology are explored, together with the methods for solving them. The following problems are considered: partitioning of sparse biological networks into *k-plex* subnetworks, prediction of the role of metabolites in metabolic reactions, partitioning of biological networks into highly connected components and the problem of identification of significant groups of proteins by adding new edges to the weighted protein interactions network. The aforementioned problems have theoretical importance in areas of machine learning and optimization, and practical application in biological research. In addition to solving the aforementioned problems from the computational aspect, the dissertation explores further application of the obtained results in the fields of biology and biochemistry, as well as the integration of results within existing bioinformatics tools.

The problem of predicting the role of metabolites in metabolic reactions is solved by a predictive machine learning method based on the conditional random fields, while for the remaining three problems the algorithams based on variable neighbourhood search are developed. For solving the problem of identification of significant groups of proteins by adding new edges to the weighted protein interactions network, the variable neighbourhood search is only the first phase of the proposed solution, while in the second and the third phase of the proposed method, the integration with additional biological information and bioinformatics tools are performed.

The proposed computational methods of partitioning and grouping in biological networks confirm existing findings in a new manner and lead to new discoveries about biological elements and the connections between them. By solving these problems and by interpreting the obtained results in this dissertation, a scientific contribution was made to the scientific field of computer science, particularly to the scientific disciplines of bioinformatics and computational biology.

Keywords
combinatorial optimization, variable neighborhood search, conditional random fields, biological networks, protein-protein interaction *k-plex*, highly connected components

Scientific field
Computer Science

Scientific subfield
Bioinformatics

UDC number
(519.179.2+004.9):577(043.3)

Zahvalnica

Posebno bih da se zahvalim mom mentoru prof. dr Gordani Pavlović-Lažetić, na nesebičnom angažovanju, podršci i korisnim savjetima tokom studija, a naročito pri izradi ove disertacije. Veliko hvala na saradnji i korisnim sugestijama prof. dr Vladimiru Filipoviću, prof. dr Draganu Matiću, doc. dr Aleksandru Kartelju i dr Branislavi Gemović. Posebno hvala prof. dr Nenadu Mitiću, čija podrška mi je mnogo značila tokom osnovih, master i doktorskih studija. Takođe, želim da izrazim veliku zahvalnost profesorima, kolegama i upravi Matematičkog fakulteta, Univerziteta u Beogradu i Prirodno-matematičkog fakulteta, Univerziteta u Banjoj Luci na podršci tokom studija i izradi disertacije.

Hvala i kolegama iz Labaratorije za bioinformatiku i računarsku hemiju, Instituta za nuklearne nauke Vinča, Univerziteta u Beogradu na saradnji. Izuzetnu zahvalnost dugujem mojoj koleginici i drugarici Savki Vračević, asistentu SP Hemija, Prirodno-matematičkog fakulteta, Univerziteta u Banjoj Luci na našoj dugogodišnjoj saradnji i velikom strpljenju za sva moja pitanja.

Sve ovo ne bi bilo moguće bez ogromne podrške moje porodice i prijatelja, posebno mojih roditelja - mame Nade i tate Milana. Beskrajno im hvala na razumijevanju, podršci, ljubavi i vjeri u moj uspjeh.

Sadržaj

Glava 1

Uvod

Enorman rast količine podataka koji potiču iz bioloških istraživanja, koji se svakodnevno pronalaze i čuvaju u raznim bazama podataka, kao i specifičnost i kompleksnost samih bioloških podataka, doveli su do potrebe razvoja sofisticiranih računarskih metoda pomoću kojih bi se ti podaci strukturirali, pohranjivali, analizirali i tumačili. Stoga je u naučnim oblastima bioinformatike i računarske biologije posljednjih godina fokus stavljen na razvoj matematičkih i računarskih modela i metoda kojim bi se zadaci obrade bioloških podataka rješavali na što efikasniji način. Potreba za sistematskim pristupom izučavanja određenih bioloških struktura (kao što su proteini ili metaboliti), kao i veza između njih, dovela je do formiranja bioloških mreža, koje se dalje analiziraju u cilju dobijanja korisnih bioloških informacija. Predstavljanje bioloških procesa u vidu mreže omogućava jasnije sagledavanje kompletnog procesa i konteksta u kojem se dešava, kao i lakše izvođenje novih saznanja i zaključaka.

Biološke mreže se mogu predstaviti kao grafovi, a različiti problemi definisani nad biološkim mrežama se mogu posmatrati kao optimizacioni problemi ili problemi mašinskog učenja. U ovom poglavlju su prvo formalno definisane biološke mreže i dat je osvrt na dostupnost podataka o biološkim mrežama (Odjeljak 1.1), zatim je dat pregled nekih aktuelnih problema koji su definisani nad biološkim mrežama i izdvojeni su problemi koji će biti razmatrani u ovoj disertaciji (Odjeljak 1.2). U Odjeljaku 1.3 dat je pregled metoda koje će se koristiti, dok su u posljednjem odjeljku ovog poglavlja (Odjeljak 1.4) prikazani bioinformatički alati i resursi korišteni za potrebe istraživanja ove disertacije.

1.1 Biološke mreže

Biološke mreže se definišu kao apstraktan prikaz bioloških sistema, koji sadrži većinu važnih karakteristika samog sistema [6]. Čvorovima u mreži odgovaraju komponente sistema, a grane između čvorova predstavljaju veze ili zavisnosti između komponenti. Informacije dobijene iz različitih istraživanja se mogu koristiti za formiranje biološke mreže čija topološka struktura sadrži važne biološke osobine. S obzirom da veze između pojedinačnih komponenti mogu biti različite jačine (značajnosti, pouzdanosti), mogu se posmatrati i težinske biološke mreže u kojima težina grane predstavlja jačinu veze [6]. Postoji nekoliko vrsta bioloških mreža, u [169] je data sljedeća podjela:

- mreže vezivanja transkripcionog faktora (engl. Transcription factor-binding networks);
- mreže proteinskih interakcija (engl. Proteinprotein interaction networks PPI mreže);

- mreže proteinskih fosforilacija (engl. Protein phosphorylation networks);
- mreže metaboličkih interakcija (engl. Metabolic interaction networks);
- mreže genskih interakcija (engl. Genetic and small molecule interaction networks);
- ostale biološke mreže.

Kao što je već napomenuto, čvorovi u navedenim mrežama su biološki elementi, najčešće proteini, geni ili metaboliti, a grana između dva čvora postoji ako između njih postoji fizička interakcija ili neka druga vrsta povezanosti. PPI mreže i mreže genskih interakcija su obično predstavljene neusmjerenim grafovima (mrežama), dok sa druge strane mreže vezivanja transkripcionog faktora, mreže metaboličkih interakcija i mreže proteinskih fosforilacija su uglavnom usmjereni grafovi (mreže) [169].

Transkripcioni faktor je protein koji kontroliše transkripciju informacije sa DNK (dezoksiribonukleinska kiselina) na iRNK (informaciona ribonukleinska kiselina). Regulacija transkripcije je jedan od osnovnih procesa koji kontroliše ekspresiju i aktivnost gena, što vodi ka fenotipskoj raznolikosti [78]. Detaljnom analizom mreža vezivanja transkripcionih faktora dobijaju se nova saznanja o hijerarhiji genskih regulatorskih mreža. Jedna od sličnosti mreža proteinskih fosforilacija sa mrežama vezivanja transkripcionog faktora jeste to što imaju uporedive topološke parametre, odnosno stepen, rastojanje, dijametar, koeficijent klasterovanja i centralnost grafa, međutim mreže proteinskih fosforilacija su obično gušće od mreža vezivanja transkripcionog faktora [169].

Mreže metaboličkih interakcija su nastale kao rezultat kombinovanja biohemijskih informacija sa genomskim sekvencama, pa ove mreže sadrže informacije i o metabolitima i o proteinima. Interakcije u metaboličkim mrežama su usko povezane sa funkcijom gena, pa se mogu koristiti za interpretaciju uloge gena. Prikupljanjem različitih informacija o genskim interakcijama, poput informacija dobijenih eksperimentalnim tehnikama genetskog skirninga, formiraju se različite mreže genskih interakcija.

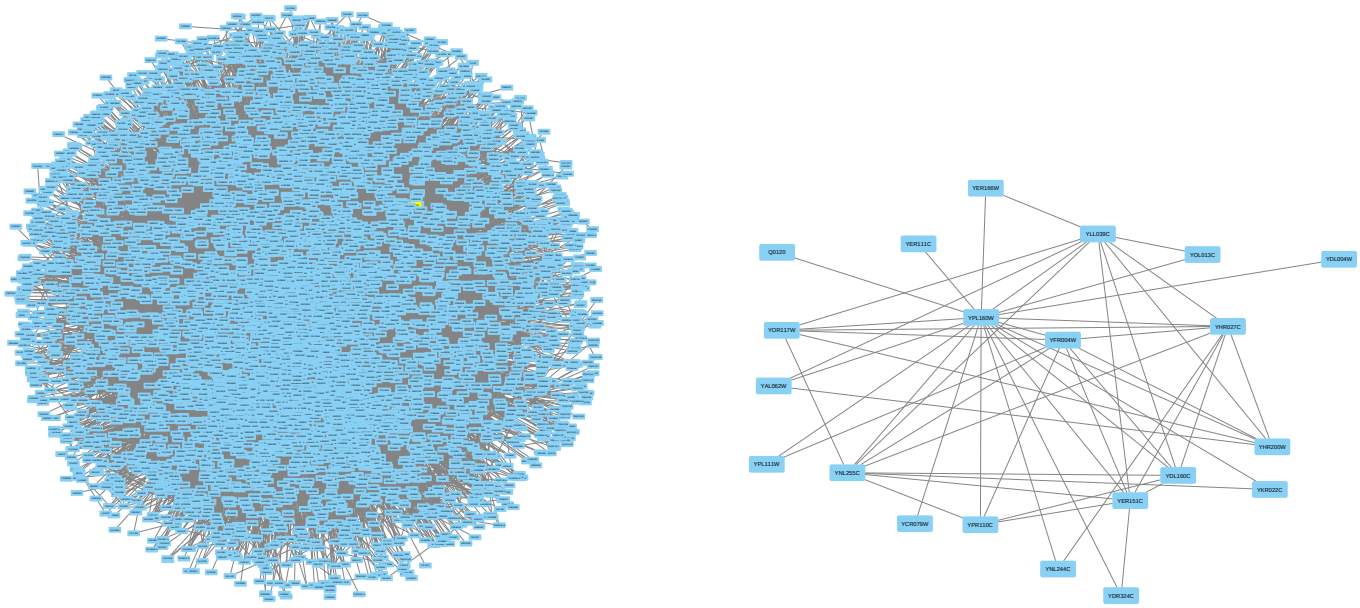
U ostale biološke mreže se ubrajaju mreže koje sadrže biološke komponente koje imaju neke zajedničke osobine, poput mreža ko-ekspresije, mreža proteina koji učestvuju u istim biološkim procesima i sl.

S obzirom da će se u istraživanjima predstavljenim u Poglavljima 4 i 5 koristite PPI mreže, sljedeći primjer ilustruje jednu mrežu ovog tipa.

Primjer 1.1. *U PPI mrežama proteini predstavljaju čvorove, a dva proteina (čvora) su povezana granom ako između njih postoji interakcija. U zavisnosti od načina formiranja i namjene PPI mreže, smatra se da interakcija između dva proteina postoji ako je dokazana in vivo fizičkim kontaktom [38] ili je riječ o interakciji koja je sa određenom pouzdanošću predviđena nekom računarskom metodom. Na Slici 1.1 (lijevo) prikazana je netežinska mreža proteinskih interakcija sa 5640 čvorova (proteina) i 59748 grana (interakcija). Na istoj slici desno se nalazi podmreža date mreže indukovana svim susjedima čvora koji predstavlja protein YPL160W.*

Brojne su baze podataka koje sadrže informacije o PPI mrežama. Neke od najpoznatijih baza PPI mreža su Database of Interacting Proteins (DIP) [163], IntAct Molecular Interaction Database (IntAct) [117], General Repository for Interaction Datasets (BioGRID) [141], STRING [102], Human Integrated Protein-Protein Interaction rEference (Hippie) [4].

Pored bioloških mreža koje se mogu naći u nekoj od postojećih baza, u cilju rješavanja specifičnih problema, često se nove mreže modeliraju na osnovu nekog postojećeg skupa bioloških informacija.



Slika 1.1: Netežinska PPI mreža (lijevo), indukovana podrmreža (desno)

Na primjer, na osnovu spiska metaboličkih reakcija mogu se formirati dvije vrste mreža. U prvoj, metaboliti su predstavljeni čvorovima, a između metabolita postoji grana ako se pojavljuju u bar jednoj zajedničkoj reakciji. Druga vrsta mreže može se konstruisati ako se metaboličke reakcije posmatraju kao čvorovi, a između čvorova postoji grana ako se bar jedan metabolit pojavljuje u obje reakcije. Ideja za ovakav način formiranja mreža je uvedena u [98], a detaljniji opis je dat u Odjeljku 2.4.

Kao što je već navedeno, pored netežinskih mogu se posmatrati i težinske biološke mreže, u kojima težina grane predstavlja jačinu, značajnost ili pouzdanost veze između čvorova koji su spojeni tom granom [6]. U nekim bazama već postoje informacije o težinama grana. Na primjer, u STRING bazi PPI mreža svaki par proteina prati informacija o intenzitetu (jačini) interakcije koji predstavlja vjerovatnoću postojanja interakcije na osnovu različitih izvora [158]. Pored toga težine na granama mogu predstavljati informaciju o topološkim osobinama mreže gdje npr. težina zavisi od broja zajedničkih susjeda dva čvora [88]. Težina može biti i biološka informacija, na primjer u metaboličkim mrežama iz [98], težina na grani je broj zajedničkih reakcija u kojima učestvuju metaboliti (prvi tip metaboličke mreže) ili broj zajedničkih metabolita za dvije reakcije (drugi tip). U PPI mreži za težinu se mogu uzeti i različite mjere sličnosti između GO (engl. Gene Ontology) termina za parove proteina [85].

1.2 Pregled problema definisanih nad biološkim mrežama

U literaturi se mogu naći brojna istraživanja o različitim problemima definisanim nad biološkim mrežama.

Za modeliranje genskih regulatorskih mreža, odnosno za opisivanje interakcija genskih regulatora se mogu koristiti Bulove mreže [70]. Bulove mreže predstavljaju dinamičke sisteme kod kojih svaka varijabla uzima jednu od dvije moguće vrijednosti (0 ili 1), u zavisnosti od funkcije koja joj je dodijeljena i vremenskog trenutka [86]. U radu [2] je predložen unaprijeđen algoritam za identifikaciju Bulovih mreža i odgovarajućih bioloških mreža, koji je baziran na brzom algoritmu za množenje

matrica i funkciji "otiska prstiju". Algoritam minimalnog dominirajućeg skupa (engl. The minimum dominating set (MDS)) je upotrebljen za obogaćivanje informacijama funkcionalnih klasa važnih proteina u PPI mreži, kao i za analizu kompleksnih bioloških mreža [108]. Jedan od često rješavnih problema u literaturi je identifikacija proteinskih kompleksa u PPI mrežama. Proteinski kompleksi čine bar dva proteina u stabilnoj, dugotrajnoj interakciji. Treba razlikovati proteinske komplekse od funkcionalnih modula [91]. Funkcionalni moduli se sastoje od proteina koji učestvuju u istom ćelijskom procesu pri čemu se međusobna interakcije dešavaju na različitom mjestu i u različitom trenutku tog ćelijskog procesa [140]. U radu [166] su kombinovane informacije o topologiji grafa i biološke informacije o genskoj ekspresiji za identifikaciju proteinskih kompleksa. Algoritam klasterovanja baziran na pronalaženju maksimalanih klika (engl. CMC algorithm - clustering-based on maximal cliques) je predložen u radu [88] i koristi se za pronalaženje kompleksa u težinskim PPI mrežama. Pored navedenog problema identifikacije proteinskih kompleksa, aktuelna istraživanja se bave i pretpostavkom da PPI mreže ne podržavaju identifikovane komplekse u dovoljnoj mjeri. Drugim riječima, u nekim PPI mrežama, proteini koji čine kompleks ne formiraju povezanu podmrežu. Za rješavanje problema dodavanja minimalnog broja grana u PPI mrežu tako da dati proteinski kompleksi postanu povezani predložen je algoritam cjelobrojnog linearnog programiranja, kao i pohlepni algoritam u radu [109].

Problem particionisanja težinskog grafa na *k-plex* podgrafove, takve da je suma težina grana u dobijenim podgrafovima što je moguće veća je definisan u radu [98]. Za podgraf sa m čvorova kažemo da je *k-plex* ako je stepen svakog čvora u tom podgrafu bar $(m-k)$. U navedenom radu je predloženo rješenje ovog optimizacionog problema metodom cjelobrojnog linearnog programiranja. Predloženi algoritam je testiran na metaboličkoj mreži, koja je formirana na osnovu spiska metaboličkih reakcija organizma *Saccharomyces cerevisiae*, koje su preuzete iz [46]. Iste metaboličke mreže su korištene i u radu [51], gdje je predstavljena nova formulacija problema pronalaženja maksimalne težinske klike.

Klasterovanje velikih bioloških mreža je koristan pristup za analizu funkcionalnih podgrupa. Problem brisanja grana uz očuvanje visoke povezanosti (engl. Highly connected deletion problem) podrazumijeva particionisanje grafa na visoko povezane podgrafove brisanjem što je moguće manje grana. Za podgraf sa n čvorova kažemo da je visoko povezan (engl. highly connected) ako je svaki čvor susjedan bar sa $n/2$ čvorova u tom podgrafu. Dokazano je da je ovaj problem NP težak [67]. U radu [67] je predstavljen egzaktan i heuristički algoritam za rješavanje ovog problema. Za testiranje algoritma su korištene PPI mreže, a dobijeni visoko povezani podgrafovi sadrže proteine koji imaju slične GO termine. Problem upita nad mrežom (engl. Network querying problem) se definiše na sljedeći način: za dati proteinski kompleks neke biološke vrste A i datu PPI mrežu neke biološke vrste B, potrebno je pronaći podmrežu mreže vrste B koja je po sekvenci, topologiji ili oboje slična datom upitu (upit je predstavljen skupom proteina). Metoda, koja je kombinacija dinamičkog programiranja i cjelobrojnog linearnog programiranja, za rješavanje ovog problema data je u [23].

Informacije o interakcijama proteina iz PPI mreža se mogu koristiti za predviđanje funkcije proteina. Na primjer, u radu [29] su informacije o indirektnim susjedima u mreži, odnosno proteinima koji nemaju direktnu zajedničku interakciju ali imaju istog zajedničkog susjeda, ulaz u statistički algoritam koji vrši predviđanje funkcije proteina.

Problemi koji su analizirani i rješavani u okviru ove disertacije su:

- Particionisanje rijetkih bioloških mreža na *k-plex* podmreže (Poglavlje 2);
- Predviđanje uloge metabolita u metaboličkim reakcijama (Poglavlje 3);

- Particionisanje bioloških mreža na visoko povezane komponente (Poglavlje 4);
- Identifikacija značajnih grupa proteina dodavanjem novih grana u težinsku PPI mrežu (Poglavlje 5).

U sljedećem Odjeljku biće opisane metode koje su korištene za rješavanje navedenih problema.

1.3 Metode statističkog modeliranja i optimizacione metode

Metode koje su korištene za rješavanje navedenih problema mogu se podijeliti na metode statističkog modeliranja i optimizacione metode. Konkretnije, od metoda statističkog modeliranja korištena je metoda uslovnih slučajnih polja čiji je detaljniji opis dat u 1.3.1, a od optimizacionih metoda promjenljivih okolina koja je opisana u 1.3.2.

1.3.1 Metoda uslovnih slučajnih polja

Uslovna slučajna polja (engl. Conditional Random Fields - CRF) su vjerovatnosni model za označavanje sekvencijalnih podataka koji je uveden u [80]. Problem sekvenciranja i označavanja sekvencijalnih podataka se može definisati kao predviđanje više varijabli koje međusobno zavise jedne od drugih i pojavljuje se u raznim naučnim oblastima, poput bioinformatike, analize teksta i slika. Predviđanje kodirajućih gena u DNK [14], označavanje piksela [59], poravnanja bioloških sekvenci, pronalaženje homologa za poznate evolucione familije, analiza sekundarne strukture RNK-a [43], procesiranja teksta i govora, označavanje vrste riječi (engl. part-of speech (POS) tagging), izvlačenje informacija [96] i sl. samo su neki od tih problema. Navedeni problemi su ranije uglavnom rješavani generativnim metodama, poput skrivenih Markovljevih modela (engl. Hidden Markov models (HMMs)) i stohastičkih gramatika (engl. stochastic grammars) [80]. U generativnim metodama izračunava se zajednička vjerovatnoća za ulazne podatke x i oznaku klase y [68], na primjer, zajednička vjerovatnoća da dvije uzastopne riječi u rečenici, koje neposredno prethode riječi koju klasifikujemo, imaju vrste riječi pridjev odnosno imenica, a da riječ koju klasifikujemo pripada klasi glagola. Određivanje zajedničke vjerovatnoće zahtijeva poznavanje informacija o svim mogućim stanjima sekvence, što nije praktično. Da bi se prevazišla navedena poteškoća potrebnog poznavanja svih mogućih stanja sekvence, uvedeni su Markovljevi modeli maksimalne entropije (engl. Maximum entropy Markov models), gdje se izračunavaju samo uslovne vjerovatnoće. Međutim, tu se pojavio novi problem - "label bias problem", odnosno problem naklonosti ka oznakama. Do ovog problema dolazi jer se u modelu maskimalne entropije prelaz određuje na osnovu uslovne vjerovatnoće mogućih sljedećih stanja s obzirom na trenutno stanje i posmatranu sekvencu, a ne u odnosu na sva moguća stanja [80]. Uvođenje uslovnih slučajnih polja je dovelo do rješenja i ovog problema.

Metoda uslovnih slučajnih polja pripada klasi diskriminativnih metoda. Pod diskriminativnim metodama se podrazumijevaju metode u kojima se izračunava uslovna vjerovatnoća za ulazne podatke x i oznaku klase y. Osnovna razlika između generativnih i diskriminativnih metoda je što uslovna vjerovatnoća $p(y|x)$ ne uključuje modelovanje $p(x)$, što, kako je već i rečeno, često nije ni praktično jer zahtijeva poznavanje mnogih zavisnosti. I generativne i diskriminativne metode imaju isti cilj, procjenu vjerovatnoće $p(y|x)$, ali do nje dolaze na različite načine. U [147] je detaljno pokazana veza između naivnog Bajesovog modela, koji je generativna metoda, i logističke regresije, koja je

diskriminativna metoda. Ova dva modela se pri rješavanju problema klasifikacije ponašaju identično. Naivni Bajesov algoritam i logistička regresija razmatraju isti prostor hipoteza, u smislu da bilo koji klasifikator logističke regresije može biti pretvoren u naivni Bajesov klasifikator sa istom granicom odluke (engl. decision boundary), i obrnuto. To znači da ako se naivni Bajesov model obuči da maksimizira uslovnu vjerovatnoću, dobiće se isti klasifikator kao i klasifikator logističke regresija. S druge strane, ako se model logističke regresije interpretira generativno i osposobi da maksimizira vjerovatnoću $p(y, x)$, tada se dobija naivni Bajesov klasifikator. Po terminologiji iz [113], naivni Bajesov model i model logističke regresije formiraju generativno-diskriminativni par. Odnos između naivnog Bajesovog modela i modela logističke regresije sličan je odnosu između HMMs i linearnog CRF. Kako je logistička regresija diskriminativni analog nivnom Bajesovom modelu, tako je CRF diskriminativni analog HMMs.

Kao što je navedeno jedan od problema sekvencijalnog predviđanja je označavanje dijelova teksta, pa neka je, na primjer, dat niz ulaznih vektora $\mathbf{w} = \{\mathbf{w_1}, \mathbf{w_2}, ..., \mathbf{w_T}\}$, pri čemu svaki vektor $\mathbf{w_s}$ sadrži informacije o riječi na poziciji s, za $0 \leqslant s \leqslant T$. Informacije o riječi mogu biti sama riječ, karakteristike poput prefiksa i sufiksa, članstvo u leksikonima specifičnim za domen i informacije u semantičkim bazama podataka kao što je *WordNet* i sl. Zadatak predviđanja je da se, za svako s, $0 \leqslant s \leqslant T$, riječi na poziciji s dodijeli odgovarajuća oznaka y_s, gdje, na primjer, oznaka može biti vrsta riječi kojoj data riječ pripada. Jedan od pristupa za rješavanje ovog problema je određivanje klasifikatora koji, nezavisno od pozicije, svakoj riječi dodjeljuje oznaku. Međutim, takvi algoritmi ne uzimaju u obzir činjenice poput one da u engleskom jeziku pridjevi stoje ispred imenice. Druga otežavajuća okolnost je što oznake mogu biti kompleksne strukture [147].

Prirodan način reprezentacije modela u kojima je više varijabli međusobno zavisno su grafovski modeli, poput Bajesovih mreža, faktor grafova, Markovljevih uslovnih slučajnih polja i sl., o kojima se više može naći u [73]. U grafovskim modelima, složene raspodjele nad više varijabli su predstavljene kao proizvodi lokalnih faktora nad manjim podskupovima varijabli. Na taj način moguće je opisati kako data faktorizacija odgovara određenom skupu uslovno nezavisnih odnosa. Takvo modeliranje je pogodnije, jer domensko znanje može da sugeriše razumne pretpostavke o uslovnim nezavisnostima, što dalje određuje izbor faktora [147]. U literaturi se mogu naći brojni radovi koji se bave "učenjem" grafovskih modela za rješavanje problema generativnim metodama. Kao što je već navedeno, nekoliko poteškoća se može pojaviti prilikom upotrebe generativnih metoda, poput velike dimenzije ulaznih podataka i kompleksne zavisnosti između varijabli, a izostavljanjem nekih zavisnosti može se dobiti model redukovanih performansi. Zbog svega navedenog, u zadnje vrijeme se sve više koriste i diskriminativne metode. Uslovna slučajna polja su u osnovi diskriminativni grafovski model, koji upravlja velikim brojem ulaznih karakteristika $\mathbf{w}$ i obezbjeđuje kompaktne izlazne informacije $\mathbf{y}$ o predviđanju, gdje je $\mathbf{y} = \{y_1, y_2, ..., y_T\}$ vektor oznaka [147].

Za izračunavanje uslovne vjerovatnoće u metodi linearnih uslovnih slučajnih polja obično se koristi formula

$$p(\mathbf{y}|\mathbf{w}) = \frac{1}{Z(\mathbf{w})} \exp\left\{ \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, \mathbf{w_t}) \right\}$$

gdje je $\mathbf{w} = \{\mathbf{w_1}, \mathbf{w_2}, ..., \mathbf{w_T}\}$ niz ulaznih vektora koji sadrže karakteristike objekata čija se oznaka predviđa, y_i, za $0 \leqslant i \leqslant T$, je oznaka i-tog objekta, f_k, $1 \leqslant k \leqslant K$ su karakteristične funkcije koje u opštem slučaju uzimaju realne vrijednosti, K je ukupan broj takvih funkcija, a θ_k, $1 \leqslant k \leqslant K$, su

komponente vektora parametara. Faktor normalizacije se definiše na sljedeći način

$$Z(\mathbf{w}) = \sum_{y \in Y^T} \exp\left\{ \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, \mathbf{w_t}) \right\}.$$

Više o linearnim uslovnim slučajnim poljima je dato u Odjeljku 3.3. Pored lineranih uslovnih slučajnih polja postoje i opšta uslovna slučajna polja (engl. General Conditional Random Fields), o kojima se više može naći u [147].

1.3.2 Metoda promjenljivih okolina

Za probleme računarske biologije formiraju se odgovarajući matematički modeli, čime se omogućava da se posmatrani problemi rješavaju matematičkim ili računarskim tehnikama. Veliki broj problema iz ovog domena spada u klasu NP teških problema, što znači da se egzaktne metode mogu koristiti samo za rješavanje tih problema gdje su dimenzije ulaznih podataka relativno male. S obzirom da su biološki podaci po svojoj samoj prirodi često velikih dimenzija, javlja se potreba za pronalaženjem kvalitetnih približnih rješenja, koja omogućavaju dalja istraživanja posmatranih bioloških struktura.

Do približnih rješenja se dolazi približnim ili aproksimativnim algoritmima. Kvalitet dobijenih približnih rješenja se procjenjuje time koliko maksimalno odstupaju od optimalnih rješenja, odnosno poznat je faktor aproksimacije. Za probleme minimizacije faktor aproksimacije je vrijednost $x > 1$ takva da dobijeno približno rješenje sigurno nije veće od $x \cdot OPT$, gdje je OPT optimalno rješenje. Faktor aproksimacije za probleme maksimizacije je vrijednost $x < 1$ takva da dobijeno približno rješenje sigurno nije manje od $x \cdot OPT$, gdje je OPT optimalno rješenje.

Za rješavanje nekih problema u ovoj disertaciji korištena je metoda promjenljivih okolina (engl. Variable Neighborhood Search - VNS). VNS je metaheuristika uvedena od strane Mladenovića i Hansena [104]. Metaheuristike pripadaju klasi tzv. "univerzalnih" heurističkih metoda, odnosno metoda koje se mogu koristiti za rješavanje različitih problema. Formalna definicija metaheuristike kao iterativnog procesa koji se zasniva na približnim metodama i kombinuje različite načine pretraživanja čitavog prostora rješenja u cilju što efikasnijeg pronalaska rješenja koja su bliska optimalnim rješenjima data je u [118]. Treba napomenuti da za razliku od aproksimativnih metoda, kod kojih se garantuje da će se dobijeno rješenje razlikovati od optimalnog za dati faktor, kod metaheurističkih metoda, u opštem slučaju, ne postoji taj faktor aproksimacije. Takođe, u radu [118] dat je i pregled različitih problema koji se mogu riješiti metaheurističkim metodama.

Posljednjih dvadesetak godina, pokazano je da je VNS efikasna i prilagodljiva tehnika koja se može koristiti za rješavanje različitih optimizacionih problema iz nauke i prakse. VNS pripada klasi optimizacionih metoda koje vrše "pretragu oko jedne tačke" (engl. single point search), jer osnovna ideja je da se istraži nekoliko okolina oko trenutno najboljeg rješenja. U svakoj okolini koju pretražuje VNS pokušava da pronađe lokalni optimum, a možda će jedan od tih lokalnih optimuma biti i globalni. Osnovni principi pretraživanja VNS metode su bazirani na empirijskim dokazima [55]:

(a) višestruki lokalni optimumi su obično blizu jedni drugih;

(b) lokalni optimum pronađen za jednu okolinu nije nužno i lokalni optimum za neku drugu okolinu.

Formalnije, neka je dat problem

$$min\{f(x)|x \in X, X \subset S\}$$

gdje su S, X, x i f prostor rješenja, prostor dopustivih rješenja, dopustivo rješenje i realna funkcija cilja, respektivno. Rješenje x^* se smatra optimalnim ako vrijedi

$$f(x^*) < f(x), \forall x \in X.$$

x_L je lokalni optimum datog problema ako vrijedi

$$f(x_L) < f(x), \forall x \in N(x_L) \cap X,$$

gdje $N(x_L)$ predstavlja okolinu rješenja x_L. Okolina rješenja x_L je skup drugih rješenja koje se po nekom definisanom kriterijumu razlikuju od rješenja x_L. Kako i sam naziv govori, VNS metoda je zasnovana na sistemu promjenljivih okolina. Najčešće se za cio broj k, $k_{min} \leqslant k \leqslant k_{max}$, formiraju okoline $N_k(x_L)$, koje na neki način zavise od k. Na primjer, ako je rješenje x_L predstavljeno vektorom $(x_{L_1}, x_{L_2}, ..., x_{L_n})$, onda bi k-ta okolina rješenja x_L mogla biti skup svih rješenja, čiji se odgovarajući vektor razlikuje od vektora datog rješenja na tačno k koordinata. Treba napomenuti da i kardinalnost k-te okoline najčešće zavisi od broja k.

VNS metoda se izvršava na sljedeći način. Nakon učitavanja ulaznih podataka inicijalizuje se početno rješenje x. Zatim se generišu okoline oko inicijalnog rješenja različite kardinalnosti. Da bi se došlo do lokalnih optimuma koji su bliži globalnom obično se koriste okoline rastuće kardinalnosti. Prije ulaska u iterativni proces ponavljanja definišu se kriterijumi zaustavljanja i to su najčešće: dostizanje maksimalnog broja iteracija, dostizanje maksimalnog broja iteracija bez poboljšanja ili dostizanje maksimalno dozvoljenog vremena izvršenja. Unutar iterativnog procesa ponavljaju se procedure razmrdavanja i lokalne pretrage. Procedura razmrdavanja razmatra trenutno posmatranu okolinu i predlaže potencijalno novo rješenje, koje je zatim ulaz u proceduru lokalne pretrage. Lokalna pretraga pokušava da unaprijedi kvalitet tog novog potencijalnog rješenja razmatrajući njegovu okolinu i rješenja unutar nje. Procedure razmrdavanja i lokalne pretrage se ponavljaju sve dok se ne dođe do poboljšanja. Kompletan postupak se ponavlja sve dok nije ispunjen neki od kriterijuma za zaustavljanje. Procedure razmrdavanja i lokalne pretrage zavise od problema koji se rješava, pa će detaljnije biti opisane u Poglavljima 2, 4 i 5.

Opisi različitih varijanti VNS metode, poput fiksne pretrage okoline (engl. Fixed neighborhood search), redukovanog VNS-a (engl. Reduced VNS), "ukošenog" VNS-a (engl. Skewed VNS), "ugnježdenog" VNS-a (engl. Nested VNS), paralelnog VNS-a (engl. Parallel VNS), Primal-dual VNS-a su dati u [55, 56].

1.4 Bioinformatički resursi i alati

Istraživanja iz bioinformatike, koja je interdisciplinarna nauka, često zahtijevaju integraciju podataka i rezultata iz različitih oblasti. U ovom odjeljku je dat pregled nekih od izvora podataka, alata i softverskih sistema koji su korišteni u istraživanjima opisanim u narednim poglavljima.

1.4.1 Genska ontologija

Genska ontologija (engl. Gene ontology - GO) obezbjeđuje sistemski skup oznaka kojima su opisani geni i genske komponente u tri osnovna domena: molekularna funkcija (engl. Molecular Function

- MF), biološki procesi (engl. Biological Process - BP) i ćelijska komponenta (engl. Cellular Component - CC) [34]. Pojmovi molekularne funkcije opisuju aktivnosti koje se odvijaju na molekularnom nivou, kao što su "kataliza" ili "transport". Termini koji odgovaraju biološkim procesima predstavljaju veće procese, ili "biološke programe" koji su se desili višestrukim molekularnim aktivnostima. Primjeri opštijih termina biološkog procesa su popravka DNK (engl. DNA repair) ili prenos signala (engl. signal transduction), a primjeri specifičnijih termina ove ontologije su proces biosinteze nukleobaze pirimidina (engl. pyrimidine nucleobase biosynthetic process) ili transmembranski transport glukoze (engl. glucose transmembrane transport). Zbog konzistentnosti, u nastavku će za termine ontologije biti korišteni engleski nazivi. Lokacije u odnosu na ćelijske strukture u kojima genski proizvod obavlja funkciju, kao npr. ćelijski kompartment (npr. mitochondrion) ili stabilni makromolekularni kompleksi kojima pripadaju (npr. ribosome) su predstavljeni terminima ontologije po ćelijskoj komponenti. Za razliku od ostalih aspekata GO, klase ćelijskih komponenata se ne odnose na procese, već na ćelijsku anatomiju.

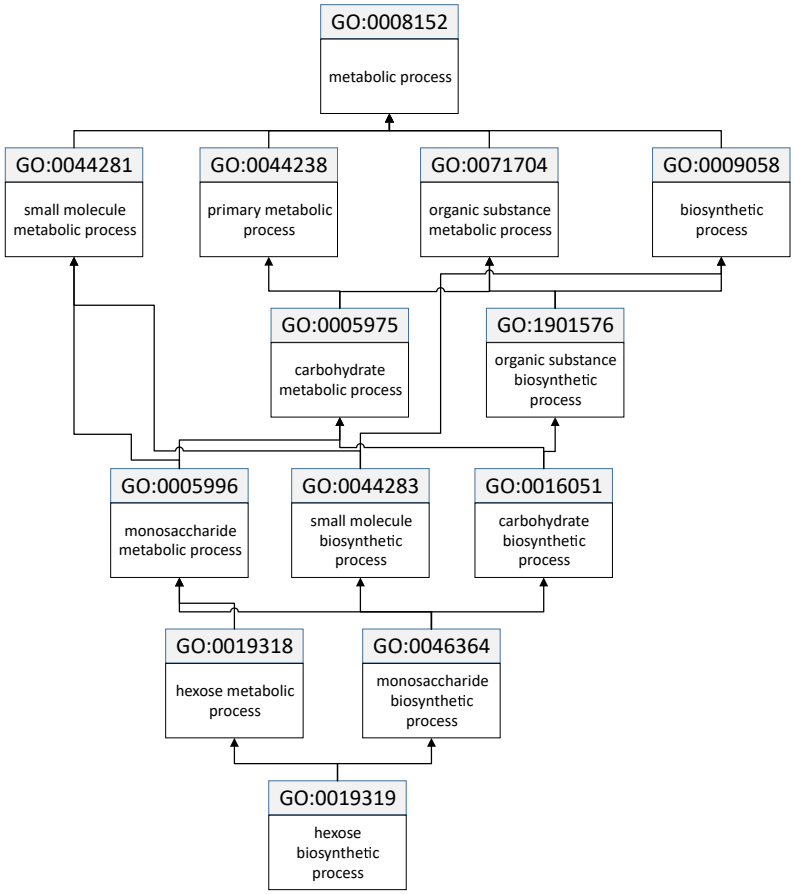
Za svaki domen formiran je usmjeren aciklični graf, u kojima čvor predstavlja jedan termin genske ontologije (GO termin), a grana između čvorova vezu između termina. Svakom terminu genske ontologije je dodijeljen jedinstven identifikator. Genska ontologija je uređena hijerahijski, pri čemu svaki dijete čvor predstavlja specijalizovaniji termin od termina roditeljskog čvora. Na primjer, na Slici 1.2 je prikazan dio grafa koji odgovara domenu bioloških procesa. Termin *hexose biosynthetic process* ima dva roditelja, *hexose metabolic process* i *monosaccharide biosynthetic process*. Tako je predstavljena činjenica da je *biosynthetic process* podtip *metabolic process*, a *hexose* je podtip *monosaccharide*. Kao što se može vidjeti sa slike, GO terminu *hexose biosynthetic process* je dodjeljen idetifikator GO:0019319, dok identifiktori GO:0019318 i GO:0046364 odgovaraju terminima *hexose metabolic process* i *monosaccharide biosynthetic process*, respektivno.

Od svakog termina postoji jedan ili više puteva kroz različite posredničke termine (čvorove) do korjenog čvora. Korjeni čvor predstavlja najopštiji termin i tri korjena čvora koja odgovaraju različitim domenima su nepovezana i nemaju zajednički nadređeni čvor, stoga postoje tri genske ontologije: ontologija po molekularnoj funkciji, ontologija po biološkim procesima i ontologija po ćelijskoj komponenti. Tri genske ontologije su disjunktne, odnosno ne postoje veze između termina iz različitih ontologija. Međutim, veze poput "dio" ili "regulacija" postoje između ontologija. Na primjer, *cyclin-dependent protein kinase activity* što je termin genske ontologije po molekularnoj funkciji je dio biološkog procesa *cell cycle* [9, 35].

1.4.2 Analiza i alati za obogaćivanje informacijama

Često se kao rezultat neke od bioloških analiza (proteomske, genske ili metaboličke) dobija lista biomolekula. Međutim, nemaju sve tako dobijene liste određen i lako prepoznatljiv kontekst, te je teško odrediti da li i kako dati geni ili proteini koje kodiraju uzajamno djeluju i kako utiču na biološke procese. Zbog toga je potrebno dodatno razmatranje informacija iz literature i baza podataka, na osnovu kojih bi se dobili odgovori na pitanja: Šta tačno radi ovaj gen i protein koji on kodira? Da li ima smisla da se ovaj gen nađe na ovoj listi? Kako to utiče na druge gene/proteine? i sl. [151]. Obavljanje ovakvog posla ručno je dugotrajan proces, a ako lista sadrži više desetina ili stotina elemenata često i nemoguć.

Analiza obogaćivanja informacijama (engl. enrichment analysis) razmatra skupove podataka, koji su obično biološki podaci poput gena ili metabolita. Rezultat analize obogaćivanja informacijama su obogaćeni termini, koji se odnose npr. na biološke funkcije, metaboličke puteve, bolesti i sl.



Slika 1.2: Dio grafa genske ontologije - biološki proces. Prilagođena slika, preuzeta sa http://geneontology.org/docs/ontology-documentation/.

Preciznije, preslikavanjem gena ili proteina iz liste u njihove pridružene biološke oznake (npr. GO termine) i uspoređivanjem distribucije termina sa distribucijom termina pozadinskog skupa, analiza obogaćivanja informacijama identifikuje termine koji su statistički prekomjerno ili nedovoljno zastupljeni u polaznoj listi. Obogaćeni pojmovi opisuju neki važan biološki proces ili ponašanje u koje su značajno uključeni geni iz polazne liste [151].

Brojni su alati koji se koriste za analizu obogaćivanja informacijama (engl. enrichment tools). U [65] data je sljedeća klasifikacija ovih alata:

- Pojedinačna analiza obogaćivanja (engl. Singular enrichment analysis (SEA));
- Analiza obogaćivanja skupa gena (engl. Gene set enrichment analysis (GSEA));
- Modularna analiza obogaćivanja (engl. Modular enrichment analysis (MEA)).

Pojedinačna analiza obogaćivanja je tradicionalna analiza obogaćivanja. U postupku ove analize testira se jedan po jedan termin nasprem liste od interesa, a p-vrijednost obogaćivanja se izračunava upoređivanjem uočene frekvencije pojavljivanja termina i očekivane frekvencije. Termini za koje je p-vrijednost manja od 0.05 se smatraju obogaćenim [151]. Analiza obogaćivanja skupa gena je metoda za interpretaciju podataka o ekspresiji gena, koja razmatra podskupove ulazne liste gena koji dijele zajedničku biološku funkciju, lokaciju u hromozomima ili regulaciju [144]. Preciznije, ova analiza, ili funkcionalna analiza obogaćivanja (engl. functional enrichment analysis) kako se

još naziva u literaturi, je metoda za identifikaciju klasa gena ili proteina koji su u značajnoj mjeri zastupljeni (engl. over-represented) u spisku gena ili proteina i mogu imati povezanost sa fenotipskim karakteristikama bolesti. Metode ove analize koriste statističke mehanizme za prepoznavanje grupa gena koje su značajno podržane ili slabo podržane informacijama. Jedan od ključnih elemenata ove analize je skor obogaćivanja informacijama (engl. Enrichment Score) koji se u različitim alatima računa na različite načine, a predstavlja informaciju u kojoj mjeri su geni, koji mogu biti grupisani na određen način, zastupljeni u ulaznom skupu podataka. Modularna analiza obogaćivanja nasleđuje osnovni proračun obogaćivanja iz SEA metode uključujući dodatne algoritme otkrivanja informacija razmatranjem veza između termina [65].

Prvoj klasi SEA metoda pripada često korišten Biological Networks Gene Ontology tool (BiNGO) alat [93]. Neki od alata pripadaju dvjema klasama, tako Database for Annotation, Visualization and Integrated Discovery (DAVID) [40] se smatra SEA/MEA metodom.

Više informacija o analizi i alatima za obogaćivanje informacijama o metabolitima je dato u Odjeljku 2.6.1.

1.4.3 Alati za vizuelizaciju

U istraživanjima je korisno vizuelno prikazati dobijene rezultate, posebno ako je riječ o objektima poput mreža, koje se lakše razumiju i tumače ako imaju grafički prikaz. Takođe, odnose u genskoj ontologiji, koja je usmjeren acikličan graf, jednostavnije je objasniti uz odgovarajuću grafičku reprezentaciju.

Brojni su alati za vizuelizaciju bioloških mreža. Pathway Studio, Osprey, PATIKA Project, VisANT su samo neki od tih alata. Više o navedenim alatima se može naći u [145]. Za potrebe istraživanja u ovoj disertaciji korišten je alat Cytoscape [134]. Pored samog grafičkog predstavljanja bioloških mreža, Cytoscape se može koristiti i za analizu mreža. Biološke mreže prikazane na Slici 1.1 su dobijene upotrebom Cytoscape softvera. Lijeva slika, koja predstavlja cijelu mrežu, je formirana na osnovu ulaznog fajla koji sadrži spisak svih grana u mreži. Desna slika, koja predstavlja indukovan podgraf polazne mreže, je dobijena upotrebom odgovarajućih opcija Cytoscape softvera. Cytoscape softver nije samo nužno alat za grafičku vizuelizaciju bioloških mreža, već može da se u interakciji sa drugim alatima, koristi i za dublju analizu mreža. Kao što je navedeno u [93], grafička reprezentacija mreža pomoću Cytoscape je jedna vrsta ulaznih podataka u BiNGO alat za obogaćivanje informacijama.

QuickGO je veb-baziran alat koji omogućava prikaz odnosa u genskoj ontologiji [15] i njegova velika prednost, u odnosu na druge alate, je što omogućava vizuelizaciju dijela genske ontologije na osnovu spiska termina odabranih od strane korisnika. Opcijom *Explore biology* QuickGO alata se za spisak GO termina, kao rezultat, dobija usmjeren graf koji predstavlja odnose između tih termina u genskoj ontologiji. Slike 4.17, 4.18 i 4.19 iz Odjeljka 4.5 su dobijene pomoću ovog alata. Pored QuickGO alata, koji je korišten u ovoj disertaciji, za vizuelizaciju odnosa u genskoj ontologiji se često koristi i AmiGO alat [25].

Glava 2

Particionisanje rijetkih bioloških mreža na *k-plex* podmreže

2.1 Uvod

Posljednjih nekoliko godina veliki napor se ulaže u pronalaženje algoritama koji bi obezbijedili bolje razumijevanje bioloških struktura i procesa. Jedan od često korištenih pristupa za otkrivanje novih osobina i funkcionalnosti je tehnika particionisanja velikih bioloških mreža u manje klastere ili funkcionalne module. U ovom poglavlju razmatran je problem particionisanja težinske mreže po granama u *k-plex* komponente. Podskup od m čvorova mreže se naziva *k-plex* ako je stepen svakog čvora u podmreži indukovanoj tim podskupom najmanje $m - k$. Problem *k-plex* maksimalnog težinskog particionisanja po granama (engl. maximum edge-weight k-plex partitioning problem - Max-EkPP) podrazumijeva particionisanje ulazne mreže u *k-plex* podmreže tako da je suma težina svih grana u dobijenim podmrežama maksimalna.

U literaturi se mogu naći brojni radovi koji se bave ovim i sličnim problemima. Tako je već pokazano da je metoda particionisanja mreža u gusto povezane podmreže, naročito klike, korisna tehnika za dobijanje novih informacija koje vode ka boljem razumijevanju odnosa i veza između bioloških elemenata. Na primjer, particionisanje u analizi proteinskih niti (engl. constrained threading problem) može se redukovati na problem pronalaženja maksimalnih težinskih klika po granama. Problem predviđanja lokacije atoma bočnog lanca u konačnoj konformaciji proteina (engl. protein side chain packing problem - SCPP) se može transformisati u problem pronalaženja maksimalne težinske klike. Svaka aminokiselina koja ulazi u sastav proteina se preslikava u proizvoljan broj rotamera, a zatim se pojedinačne konfiguracije (svi atomi iz proizvoljnog rotamera) predstavljaju čvorovima grafa [3]. Funkcija za postavljanje težine na granama je definisana tako da predstavlja frekvenciju pojavljivanja kontaktnih parova u bazi proteina [21]. Pronalaženje klika je takođe važan metod za identifikaciju klastera koji se kasnije dijele na proteinske komplekse i dinamičke funkcionalne module. Analiziranjem struktura koje se nalaze u PPI mrežama mogu se identifikovati međusobno gusto povezani molekulski moduli, koji nisu tako gusto povezani sa ostatkom mreže [140]. Na sličan način klike se mogu koristiti i za modularnu dekompoziciju PPI mreža. Dekompozicija dozvoljava udruživanje proteina u stvarne funkcionalne komplekse tako što identifikuje grupe proteina koji djeluju kao jedna jedinica [48].

S druge strane, brojne klase bioloških mreža sadrže rijetke mreže, čija podjela na klike može biti vrlo restriktivna. Stoga se mnoge potencijalne informacije o interferenciji bioloških objekata

mogu izgubiti. Dakle, pristup kod koga je uslov da se mreže particionišu na klike relaksiran mogao bi biti mnogo korisniji. Način particionisanja koji je predstavljen u ovom poglavlju na određen način zadržava jaku povezanost dobijenih komponenti, ali ne toliko restriktivnu kao kada su particije klike. Relaksirajući klike do rječe povezanih grafova, biološki objekti se povezuju u semantičke ili funkcionalne logičke grupe koje nazivamo *k-plex*-i, imajući na umu da ukupna suma težina po svim particijama treba da bude što je moguće veća.

2.2 Raniji rezultati

k-plex struktura je uvedena u radu [130] kasnih 1970-ih godina, kao struktura slična kliki promjenljive povezanosti. Identifikacija *k-plex* maksimalne kardinalnosti u rijetkom težinskom grafu je definisana kao optimizacioni problem. Ovaj problem se naziva maksimalni *k-plex* problem (engl. maximum *k*-plex problem - Max-kP problem).

Iako bi se moglo očekivati da će ovako formulisan problem zainteresovati istraživače, problem nije detaljno analiziran više od 30 godina. U međuvremenu, razvoj interneta i drugih tehnologija baziranih na računarima, uključujući bioinformatiku, doveo je do generisanja ogromne količine različitih podataka o interakcijama. Balasundaram i saradnici [11] su vratili pomenuti problem u centar pažnje naučne zajednice, tako što su prepoznali njegovu blisku vezu sa ponašanjem nekih stvarnih mreža, posebno društvenih mreža. U prethodno pomenutom radu je pokazano da je problem pronalaženja *k-plex*-a maksimalne kardinalnosti (engl. the problem of identification of a maximum cardinality k-plex - Max-kP) u rijetkim netežinskim mrežama NP težak problem i predstavljeno je rješenje metodom cjelobrojnog linearnog programiranja (engl. integer linear programming - ILP). ILP rješenje razvijeno za problem maksimalne klike iz [97] može se koristiti i za rješavanje drugih srodnih problema, uključujući i problem *k-plex*-a maksimalne veličine. Osim egzaktnih metoda baziranih na cjelobrojnom linearnom programiranju, postoje i heurističke metode za rješavanje Max-kP problema u rijetkim netežinskim grafovima. Na primjer, MekKloski i Hiks [100] su prilagodili kombinatorne klika algoritme za pronalaženje maksimalnih *k-plex*-a i predložili novu gornju granicu za kardinalnost *k-plex*-a. Mozer i saradnici [106] su predložili neke praktične algoritme za pronalaženje maksimalnih *k-plex*-a, koji su bolji od drugih pristupa. *k-plex* klasterovanje je takođe vrsta nehijerahijske dekompozicije grafa u klastere, što omogućava primjenu paralelnih algoritama.

Nekoliko drugih relaksacija klike, kao i adekvatnih rješenja matematičkog programiranja je razmatrano u [119]. Proteini koji imaju slične GO termine mogu se grupisati u iste klastere particionisanjem velikih bioloških mreža, poput PPI mreža, u gusto povezane komponente [67]. O ovom problemu će više biti riječi u Poglavlju 4. Klasterovanje velikih podataka ima važnu ulogu u analizi genske ekspresije. U [57] klaster analiza cDNK "otisaka prstiju" je korištena za identifikaciju klonova koji odgovaraju istom genu. U [110] mnoga blizu-optimalna (engl. near-optimal) klasterovanja su upotrebljena u cilju istraživanja dinamike mrežnih klasterovanja. To je kasnije primijenjeno na nekoliko bioloških i drugih mreža. Da bi pokazali koji tipovi zapažanja se mogu dobiti iz velikih kolekcija blizu-optimalnih rješenja, autori su analizirali ERK1/ERK2 mitogen-aktivni protein kinazu (MAPK18) signalno-transdukcione putanje i mrežu kortikalno-kortikalnih veza u ljudskom mozgu.

Identifikacija kohezivnih podgrupa (ne nužno klika i *k-plex*-a) je takođe primjenjena na brojne mreže koje nisu biološke: razmatranje terorističkih i drugih kriminalnih mreža [26], veb grafova [150], bežičnih mreža [75], za pronalaženje strukturalnih obrazaca u društvenim mrežama [107], istraživanje teksta [10], berze [17], itd.

Komplementaran problem problemu indetifikacije *k-plex*-a u grafu G je problem identifikacije *co-k-plex*-a u grafu $\overline{G}$. Podskup S skupa čvorova grafa G je *co-k-plex* ako je stepen svakog čvora u podgrafu indukovanom sa S najviše $k-1$. Iz definicije slijedi da je S *co-k-plex* u G ako i samo ako je S *k-plex* u komplementarnom grafu $\overline{G}$. Za $k=1$, *co-k-plex* je nezavisan skup čvorova. Takva relaksacija problema maksimalne klike je u bliskoj vezi sa problemom defektivnog bojenja [36, 153], koji je relaksacija poznatog problema bojenja čvorova u grafu. (κ, d)-bojenje je bojenje čvorova grafa sa κ boja takvo da nijedan čvor nije susjedan sa više od d čvorova iste boje. Za $d=0$, (κ, d)-bojenje je problem pravilnog bojenja grafa. Za dati broj d, pronalaženje odgovarajućeg *co-(d-1)-plex*-a odgovara (κ, d)-bojenju, gdje je dozvoljeno da čvorovi u *co-(d-1)-plex*-u budu obojeni istom bojom.

Veliki broj rezultata iz literature koji se odnose na particionisanje težinskog grafa u različite komponente je vezan za problem particionisanja u maksimalne težinske klike (engl. the maximum edge-weight cliques partitioning problem - Max-ECP). Cilj Max-ECP je klasterovanje čvorova u disjunktne klike, takve da je ukupna suma težina na granama po svim particijama što je moguće veća. Iako je Max-ECP specijalni slučaj problema Max-EkPP za $k=1$, razmatran je uglavnom na kompletnim grafovima i to u nekoliko radova [54, 42, 116, 160], kao i sa najnovijim predloženim heurističkim metodama [168, 20]. Kao što je već pomenuto, particionisanje rijetkih grafova u klike može biti previše restriktivno, jer mnoge korisne informacije o odnosima između elemenata se mogu izgubiti. Uzimajući u obzir to razmatranje, Martins [98] je predložio model cjelobrojnog linearnog programiranja za rješavanje Max-EkPP problem sa polinomijalnim brojem promjenljivih i ograničenja, takođe razmatrajući uključivanje dodatnih topoloških ograničenja u model. Predloženi ILP model je testiran na biološkim i vještačkim mrežama, koje su korištene i u ovom istraživanju.

2.3 Rješavanje Max-EkP problema

2.3.1 Definicija problema

Neka je dat graf $G=(V,E)$, gdje je $V=\{1,2,...,|V|\}$ skup čvorova, a E skup grana. Oznaka uv predstavlja granu koja povezuje čvorove u i v. Težina grane uv je realan broj $w_{uv}>0$. Sa $\mathcal{P}=(V_1,V_2,...,V_l)$ je označena particija skupa čvorova u l disjunktnih komponenti takvih da je $\bigcup_{i=1}^{l} V_i = V$. Neka je $k \geqslant 1$ cijeli broj. Kao što je već rečeno, V_i je *k-plex* ako je $\forall v \in V_i$ $deg(v) \geqslant |V_i|-k$ u grafu indukovanom skupom čvorova V_i. Težina komponente V_i jednaka je sumi težina svih grana koje se nalaze u grafu indukovanom čvorima iz V_i. Skup grana u tako indukovanom podgrafu je označen sa E_i. Težina cijele particije je suma težina svih komponenti te particije. Max-EkPP se definiše kao problem pronalaženja particije grafa G čija je ukupna težina maksimalna i svaka komponenta je *k-plex*.

Funkcija cilja je data formulom (2.1).

$$obj(\mathcal{P}) = \sum_{i=1}^{|\mathcal{P}|} \sum_{uv \in E_i} w_{uv} \tag{2.1}$$

Optimizacioni Max-EkPP problem je dat sa:

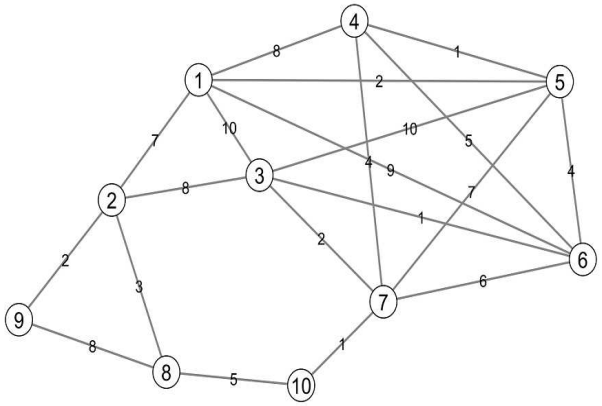
$$\max_{\mathcal{P}} obj(\mathcal{P}) \tag{2.2}$$

pri čemu treba da vrijedi

$$\forall V_i \in \mathcal{P},\ \forall v \in V_i,\ \deg(v) \geqslant |V_i| - k. \tag{2.3}$$

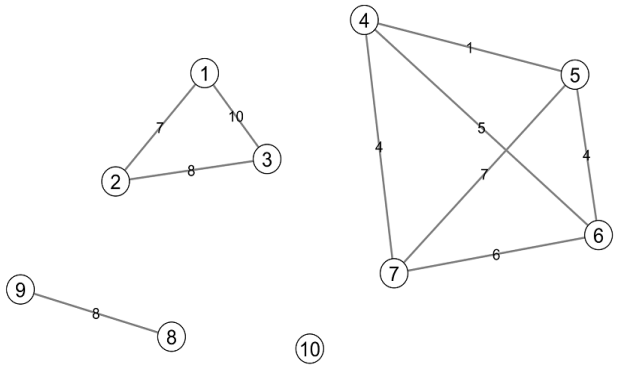
Skup ograničenja (2.3) obezbjeđuje da će svaki podskup biti k-plex.

Primjer 2.1. *Na Slici 2.1 prikazan je graf sa 10 čvorova i 20 grana, čija je gustina 0.444.*



Slika 2.1: Težinski neusmjeren graf

Optimalna rješenja Max-EkP problema za ovaj graf i vrijednosti parametra $k = 1, 2$ i 3 su data na Slikama 2.2, 2.3 i 2.4, respektivno. Za vrijednost $k = 1$ vrijednost funkcije cilja za optimalno rješenje je 60, dok je za $k = 2$ i $k = 3$ vrijednost funkcije cilja 82, odnosno 87.



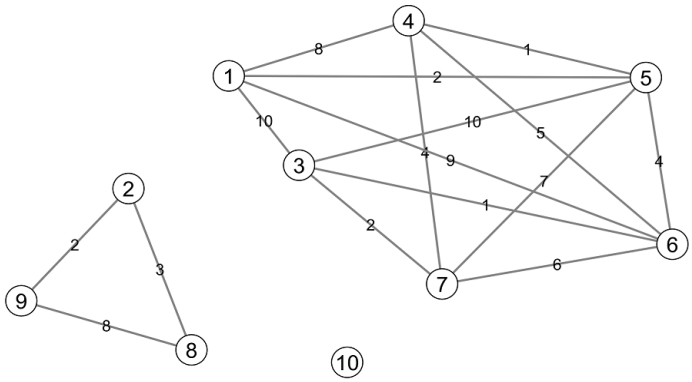
Slika 2.2: Optimalno rješenja za Max-E1PP ako je ulazni graf sa Slike 2.1

Za razliku od rješenja za vrijednost parametra $k = 1$ sa Slike 2.3 se može primijetiti da čvor 2 više nije u istom k-plex-u sa čvorovima 1 i 3, jer relaksacija uslova da svaki čvor ne mora biti povezan sa svakim drugim čvorom iz k-plex-a omogućava da se ova dva čvora sada nađu u $2-plex$-u koji ima šest čvorova (zajedno sa čvorovima 4,5,6 i 7).

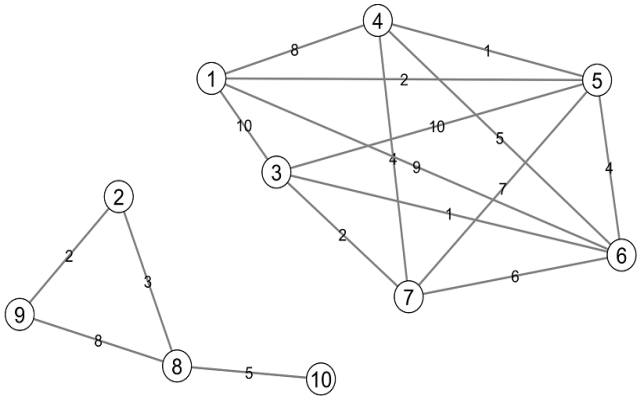
Dodatna relaksacija uslova za stepene povezanosti u k-plex-u za vrijednost parametra $k = 3$, omogućava dodatno "ukrupnjavanje" k-plex-a, te su sada čvorovi 2, 8, 9 i 10 u jednom k-plex-u.

2.3.2 Metoda promjenljivih okolina za rješavanje za Max-EkPP

U ovom poglavlju predloženo je rješenje Max-EkPP metodom promjenljivih okolina. Rezultati koji su prikazani ovdje su predstavljeni u radu [53]. Struktura VNS algoritma za rješavanje Max-EkPP prikazana je na Slici 2.5.



Slika 2.3: Optimalno rješenja za Max-E2PP ako je ulazni graf sa Slike 2.1



Slika 2.4: Optimalno rješenja za Max-E3PP ako je ulazni graf sa Slike 2.1

input: n_{min}, n_{max}, it_{max}, $itrep_{max}$, t_{max}, $prob$, k
output: **x**

```
1  x ← initializeSolution();
2  n ← n_min; it ← 1 ;
3  while it < it_max ∧ (it − it_lastimpr) < itrep_max ∧ t_run < t_max do
4  |  x′ ← shaking(x,n);
5  |  x″ ← localSearch(x′, k);
6  |  move ← shouldMove(x, x″, prob);
7  |  if move then
8  |  |  x ← x″ ;
9  |  else if n < n_max ∧ n < |x| then
10 |  |  n ← n + 1;
11 |  else
12 |  |  n ← n_min;
13 |  it ← it + 1;
14 end
```

Slika 2.5: Struktura VNS algoritma za Max-EkPP.

Pored grafa G ulazni podaci za VNS algoritam su:

- n_{min} i n_{max} su minimalna i maksimalna veličina okoline koju razmatra VNS;
- it_{max}, $itrep_{max}$, t_{max} su maksimalan broj ukupnih iteracija, maksimalan broj iteracija bez poboljšanja, maksimalno vrijeme izvršenja u sekundama;

- *prob* je vjerovatnoća prelaska iz jednog rješenja u drugo rješenje istog kvaliteta;
- k je cijeli broj koji odgovara vrijednosti parametra k za Max-EkPP.

Veličina n-te okoline, $n_{min} \leqslant n \leqslant n_{max}$, određuje koliko čvorova iz posmatranog rješenja će biti premješteno u drugu particiju, što će detaljnije biti opisano u Odjeljku 2.3.4.

VNS se obično realizuje kroz dvije osnovne procedure, odnosno proceduru "razmrdavanja" (engl. shaking) i proceduru lokalne pretrage (engl. Local Search - LS). Procedura razmrdavanja upravlja okolinama i u svakoj iteraciji slučajno bira novu tačku iz neke okoline trenutnog rješenja. Preciznije, u okviru ove procedure algoritam na osnovu trenutnog rješenja (trenutne particije) formira novu particiju premještanjem određenog broja čvorova (koji zavisi od veličine okoline) iz jedne komponente u drugu. Osnovni cilj procedure razmrdavanja je da riješi situacije kada procedura lokalne pretrage "zaglavi" u lokalnom suboptimalnom rješenju. Više detalja o proceduri razmrdavanja je dato u Odjeljku 2.3.4.

Na varijabli $it_{lastimpr}$ se čuva informacija o iteraciji u kojoj se desilo posljednje poboljšanje, a na varijabli t_{run} informacija o ukupnom vremenu izvršavanja. Unutar procedure lokalne pretrage algoritam pokušava da popravi rješenje predloženo od strane procedure razmrdavanja. Lokalna pretraga sistematično provjerava druga rješenja u najbližoj okolini predloženog rješenja. Pri tome, najbližoj okolini pripadaju ona rješenja kod kojih je tačno jedan čvor premješten iz jedne komponente u neku drugu. Unutar procedure `shouldMove()` algoritam odlučuje da li da nastavi sa trenutno najboljim rješenjem ili sa novim rješenjem koje je rezultat lokalne pretrage. Više detalja o lokalnoj pretrazi i proceduri `shouldMove()` je dato u Odjeljku 2.3.5.

U glavnoj petlji algoritma, procedura razmrdavanja se poziva iterativno sve dok se najbolje rješenja unutar trenutne okoline poboljšava. Nakon što nema daljeg poboljšanja algoritam prelazi u narednu okolinu. Nakon pretraživanja posljednje okoline čija je veličina n_{max}, pretraga ponovo počinje od okoline čija je veličina n_{min}.

Izvršavanje VNS algoritma se zaustavlja kada je ispunjen neki od sljedećih uslova: dostignut je maksimalan broj iteracija, dostignut je maksimalan broj iteracija bez poboljšanja za trenutno najbolje rješenje ili dostignuto je maksimalno vrijeme izvršavanja.

2.3.3 Reprezentacija rješenja i funkcija cilja

Rješenje predloženog VNS algoritma je predstavljeno nizom $\mathbf{x}$ cijelih brojeva dužine $|V|$. Svaki element niza odgovara jednom čvoru grafa, označavajući kojoj komponenti pripada odgovarajući čvor. Preciznije, čvor i je pridružen komponenti V_j ako je $x_i = j$.

Polazno rješenje se određuje na slučajan način, tako što se svakom elementu niza $\mathbf{x}$ dodjeljuje vrijednost slučajnog cijelog broja iz intervala $[1, 2, ..., UB]$. Izbor vrijednosti gornje granice (engl. upper bound - UB) je diskutovan kasnije. Rješenja koja zadovoljavaju uslove (2.3) su dopustiva rješenja, dok rješenja koja ne zadovoljavaju (2.3) su nedopustiva. Nedopustiva rješenja su implicitno dozvoljena na osnovu predložene reprezentacije i inicijalizacije početnog rješenja. To povoljno utiče na proces pretraživanja, jer omogućava da funkcija cilja usmjerava pretragu u perspektivnije i dopustivije oblasti, bez potrebe za uvođenjem teške funkcije kazne za blago nedopustiva rješenja.

Uobičajan način za prevazilaženje situacija u kojima se pojavljuju nedopustiva rješenja je upotreba funkcije kazne. Ovdje je funkcija kazne formirana sa dva cilja: da suptilno spriječi nedopustiva rješenja i da maksimizuje ukupnu težinu particije.

Neka je $\mathcal{P} = (V_1, V_2, ..., V_l)$ particija koja je (ne nužno dopustivo) rješenje za Max-EkPP. Sa w_{total} je označena ukupna suma težina svih grana u grafu G, tj. $w_{total} = \sum_{uv \in E} w_{uv}$. Čvor $v \in V_j$, $j \in \{1, 2, ..., l\}$, se smatra "korektnim" ako je stepen čvora v u grafu koji je indukovan skupom čvorova V_j najmanje $|V_j| - k$. To znači ako je svaki čvor u komponenti "korektan", onda je komponenta *k-plex*. Sa N_c je označen ukupan broj korektnih čvorova u rješenju.

Predložena funkcija kazne je data sa (2.4).

$$f_p(\mathcal{P}) = (N_c - |V|) \cdot w_{total} \tag{2.4}$$

Funkcija kazne se kombinuje sa funkcijom cilja koja je je uvedena formulom (2.1), pa je tako konačna VNS funkcija cilja koja se maksimizuje predstavljena formulom (2.5)

$$obj_{VNS}(\mathcal{P}) = obj(\mathcal{P}) + f_p(\mathcal{P}). \tag{2.5}$$

$|V|$ i w_{total} su konstante, a ako je rješenje dopustivo onda su svi čvorovi korektni, tj. $N_c = |V|$, odakle slijedi da je funkcija kazne jednaka 0. Vrijednost funkcije kazne za bilo koje nedopustivo rješenje je manje od nule, što znači da će se prednost dati dopustivim rješenjima. Dodatno, za dva nedopustiva rješenja ono koje ima veći broj korektnih čvorova će imati prednost jer u tom slučaju funkcija kazne manje smanjuje funkciju cilja. Kao posljedica svega navedenog, proces maksimizacije odbacuje rješenja koja imaju puno nekorektnih čvorova i usmjerava pretragu u dopustive oblasti. Istovremeno, predložena VNS funkcija cilja formira adekvatan raspored nedopustivih rješenja. Na taj način se daje šansa boljim nedopustivim rješenjima da se pojave, što za posljedicu može imati da se nakon lokalne pretrage transformišu u dopustiva rješenja boljeg kvaliteta.

Da bi početni broj komponenti bio određen što je moguće preciznije, razmatrano je nekoliko različitih gornjih granica:

(i) $UB = 1$,

(ii) $UB = \log |V|$,

(iii) $UB = c|V|,\ c \in \{0.1, 0.5, 1\}$,

(iv) $UB = \sqrt{|V|}$.

Rezultati testiranja na nekoliko različitih vrsta grafova su pokazali da je $\sqrt{|V|}$ najadekvatnija gornja granica. Razlog za to je što se predloženi VNS algoritam bolje ponaša u slučaju postepenog dodavanja novih komponenti, nego postepenog uklanjanja postojećih komponenti. Dakle, izabrana gornja granica bi trebalo da obezbjeđuje manji broj početnih komponenti nego konačno rješenje. Iako se čini da bi gornje granice $UB = 1$ i $UB = \log |V|$ bile bolji izbor, pokazalo se da imaju manje djelotvoran uticaj na kvalitet konačnog rješenja. Razlog za to je činjenica da je početni broj komponenti previše mali, što usporava konvergenciju algoritma, posebno za grafove veoma velikih dimenzija. Linearna funkcija nije odgovarajući izbor za gornju granicu jer odnos između prosječnog broja komponenti u rješenju i broja čvorova nije linearan. Stoga, izabrana gornja granica $\sqrt{|V|}$ je dobar kompromis za sve razmatrane aspekte.

2.3.4 Procedura razmrdavanja

Osnovna namjera procedure razmrdavanja je proširenje prostora pretrage trenutnog rješenja da bi se smanjila mogućnost da se algoritam "zaglavi" u lokalnom optimumu. Unutar procedure razmrda-

```
   input: x, n
   output: x''
1  positionIndices ← selectNRandomPositions(n);
2  l ← countDistinctValues(x)+1;
3  x' = x;
4  foreach index ∈ positionIndices do
5  |  x'_index ← q ← random(1, l);
6  end
```

Slika 2.6: Procedura razmrdavanja, gdje je $\mathbf{x}$ trenutno najbolje rješenje, a n veličina okoline koja se razmatra

vanja, koja je prikazana na Slici 2.6, algoritam formira sistem okolina koje koristi za usmjeravanje ka novom rješenju na osnovu trenutno najboljeg rješenja $\mathbf{x}$.

Za definisanje n-te okoline, $n_{min} \leqslant n \leqslant n_{max}$, korištena je sljedeća procedura. Na slučajan način, funkcijom `selectNRandomPositions(n)`, se bira n pozicija čvorova iz V. Čvorovi sa tih izabranih pozicija će u proceduri razmrdavanja biti premješteni iz postojećih u neke druge slučajno izabrane komponente. Prije samog prebacivanja, određuje se ukupan broj komponenti u posmatranom rješenju $\mathbf{x}$, funkcijom `countDistinctValues(x)`. Neka je sa l označen ukupan broj komponenti uvećan za jedan. Dalje, za svaku izabranu poziciju (4. linija pseudokoda sa Slike 2.6), algoritam mijenja komponentu kojoj čvor sa izabrane pozicije pripada na sljedeći način. Na slučajan način bira se cijeli broj q iz skupa $\{1, 2, ..., l\}$. Ako je $q < l$, onda se čvor premješta u postojeću particiju V_q. Ako je $q = l$, onda se formira nova particija koja sadrži samo čvor koji se premješta i ukupan broj particija se povećava za jedan. Ako komponenta u kojoj se prethodno nalazio izabrani čvor postane prazna, onda se ukupan broj komponenti smanjuje za jedan. Ovako definisana strategija dozvoljava promjenu ukupnog broja komponenti tokom procesa pretrage. Dakle, predložena procedura razmrdavanja ima dva cilja, odnosno prebacivanje čvorova iz jedne u drugu komponentu i mogućnost smanjenja ili povećanja ukupnog broja komponenti. Rješenje $\mathbf{x}'$, koje se dobija procedurom razmrdavanja, se dalje unaprijeđuje u fazi lokalne pretrage.

2.3.5 Lokalna pretraga

Procedurom lokalne pretrage istražuje se okolina novog rješenja dobijenog procedurom razmrdavanja, u cilju dostizanja lokalnog optimalnog rješenja. U predloženom VNS-u, lokalna pretraga je bazirana na premještanju elementa iz jedne komponente u drugu, uz primjenu strategije prvog unapređenja (engl. "1-swap first improvement") (Slika 2.7). Rješenje $\mathbf{x}'$ dobijeno procedurom razmrdavanja je ulaz u fazu lokalne pretrage. Zbog jasnije notacije, rješenje koje se na osnovu ulaznog rješenja $\mathbf{x}'$ formira u fazi lokalne pretrage označeno je sa $\mathbf{x}''$. Lokalna pretraga iterativno razmatra nova rješenja koja formira premještanjem jednog čvora iz komponente kojoj pripada u drugu komponentu, na sljedeći način. Na slučajan način se bira pozicija i (4. linija pseudokoda sa Slike 2.7), te se dalje razmatra premještanje čvora koji se nalazi na poziciji i. Označimo taj čvor sa v. Slično kao i u proceduri razmrdavanja, funkcijom `countDistinctValues(x'')` se prebroji broj komponenti rješenja $\mathbf{x}''$, a sa l je označen ukupan broj tih komponenti, uvećan za 1. Zatim, na slučajan način se bira cijeli broj p iz skupa $\{1, 2, ..., l\}$. Ako je $p < l$ čvor v se premješta u postojeću komponentu V_p, a u slučaju da je $p = l$ formira se nova komponenta $V_p = \{v\}$. Zatim se, u 9. liniji pseudokoda sa Slike 2.7, određuje vrijednost koju bi funkcija cilja imala ako se čvor v iz komponente u kojoj se trenutno nalazi premjesti u komponentu p. Parcijalno izračunavanje funkcije cilja se obavlja pomoću

```
   input: x′, k
   output: x″
 1 x″ ← x′;
 2 impr ← true;
 3 while impr do
 4 |   impr ← false; i ← ir ← random(1,|x″|);
 5 |   do
 6 |   |   l ←countDistinctValues(x″)+1;
 7 |   |   p ← pr ← random(1,l);
 8 |   |   do
 9 |   |   |   newObj ← repositionObjectiveValue(x″,i,p,k);
10 |   |   |   if newObj > x″.obj then
11 |   |   |   |   x″ ← reposition(x″,i,p,k);
12 |   |   |   |   impr ← true; break;
13 |   |   |   p ← (p  mod l) + 1;
14 |   |   while p ≠ pr;
15 |   |   if impr then
16 |   |   |   break;
17 |   |   i ← (i  mod |x″|) + 1;
18 |   while i ≠ ir;
19 end
```

Slika 2.7: Procedura lokalne pretrage za Max-EkPP, gdje je $\mathbf{x}'$ - rješenje dobijeno procedurom razmrdavanja, k cijeli broj koji odgovara vrijednosti parametra k za Max-EkPP

procedure `repositionObjectiveValue`, čiji je pseudokod dat na Slici 2.8, a koja će detaljno biti opisana u nastavku. Ako se postiglo poboljšanje rješenja, odmah dolazi do promjene i $\mathbf{x}''$ se ažurira (11. linija pseudokoda), a lokalna pretraga ponovo pokreće vanjsku petlju (`while` petlju u 3. liniji pseudokoda). Odnosno, pokušava se da se pronađe novo poboljšanje tako što se bira novi čvor koji je novi kandidat za promjenu komponente kojoj pripada (4. linija pseudokoda). Ako se nije postiglo poboljšanje, lokalna pretraga ponavlja postupak sa sljedećom kandidatskom komponentom (13. linija pseudokoda), sve dok ne razmotri sve kadidatske komponente ili dok ne pronađe poboljšanje. Nakon što razmotri sve kandidatske komponente, a nije pronađeno poboljšano rješenje, lokalna pretraga ponavlja postupak sa sljedećim čvorom (17. linija pseudokoda). Lokalna pretraga se zaustavlja ako su razmotreni svi čvorovi, a nije postignuto poboljšanje.

S obzirom da je lokalna pretraga vremenski najzahtjevnija faza u čitavom VNS algoritmu, od velike je važnosti formirati je na način tako da bude što je moguće efikasnija, uzimajući u obzir i kvalitet dobijenog lokalnog optimuma. Kao što je već rečeno, lokalna pretraga sistematski ispituje okoline datih rješenja, tako što pomjera jedan čvor iz njegove početne komponente u neku drugu komponentu. Ovo dovodi samo do djelimične promjene u strukturi rješenja, pa se može primijeniti parcijalno izračunavanje funkcije cilja novoformiranog rješenja. Na Slici 2.8 je prikazan pseudokod procedure `repositionObjectiveValue` za parcijalno računanja VNS funkcije cilja. Na osnovu formule (2.5), vrijednost VNS funkcije cilja se računa na osnovu dva sabirka: funkcije cilja samog problema MaxEkP i funkcije kazne. Ova procedura računa VNS funkciju cilja na osnovu vrijednosti funkcije cilja rješenja $\mathbf{x}''$ i pomjeranja čvora sa i pozicije iz trenutne u komponentu p. Ovo pomjeranje uzorkuje promjenu u lokalnoj strukturi rješenja, pa samo čvorovi iz inicijalne i ciljne komponente i njihove grane trebaju biti razmatrani u parcijalnom izračunavanju. Skup čvorova koji će se razmatrati je označen sa $V_{relevant}$ i formira se u 3. liniji pseudokoda sa Slike 2.8. Razmatra se svaki čvor u iz skupa $V_{relevant}$ (4. linija pseudokoda) i na osnovu njega podešava se vrijednost N_c, koja

predstavlja broj korektnih čvorova u rješenju. Dijelom pseudokoda od 5. do 9. linije, su razmotrene sve moguće situacije koje mogu nastati ovim premještanjem. Situacija u kojoj je čvor u bio korektan prije premještanja (ispunjen uslov iz 5. linije pseudokoda) vodi ka smanjenju vrijednosti N_c za 1, ali ako je čvor u i nakon premještanja ostao korektan (ispunjen uslov iz 8. linije pseudokoda), broj korektnih čvorova će se povećati za 1 u 9. liniji datog pseudokoda. Ako će čvor u od korektnog čvora, nakon premještanja postati nekorektan, onda će se broj korektnih čvorova smanjiti za 1 u 6. liniji pseudokoda (povećanje u 9. liniji se neće dogoditi jer je u nekorektan). Ako čvor u prije premještanja nije bio korektan (nije ispunjen uslov iz 5. linije pseudokoda), ali je postao korektan u novom rješenju (ispunjen je uslov iz 8. linije pseudokoda), onda se broj korektnih čvorova povećava za 1. Ako je u bio i ostao nekorektan čvor, onda nema promjene vrijednosti N_c. Slično, sve grane koje su incidentne sa nekim od čvorova iz skupa $V_{relevant}$ (skup takvih grana se formira u 10. liniji pseudokoda) se provjeravaju i podešava se vrijednost *obj*, pri čemu se grana smatra korektnom ako spaja dva korektna čvora. Do promjene vrijednosti *obj* dolazi u dvije situacije: grana je nakon pomjeranja postala korektna, a prije pomjeranja nije bila korektna (nije ispunjen uslov iz 13. linije pseudokoda, a ispunjen je uslov iz 15. linije) i obrnuto, prethodno je bila korektna, ali nakon pomjeranja više nije korektna (ispunjen uslov iz 13. linije pseudokoda, a nije ispunjen je uslov iz 15. linije). VNS funkcija cilja se na kraju računa u posljednjoj liniji pseudokoda sa Slike 2.8, prema formuli (2.5).

```
   input: x'', i, p, k
   output: obj_VNS
 1 N_c ← correctVertices(x'', k);
 2 obj ← sumOfEdges(x''); p_old ← x''_i;
 3 V_relevant ← {u|u ∈ x'', u = p_old} ∪ {u|u ∈ x'', u = p};
 4 foreach u ∈ V_relevant do
 5 |   if u_correct then
 6 |   |   N_c → N_c − 1;
 7 |   u'_correct = correctAfterReposition(u, p_old, p, k);
 8 |   if u'_correct then
 9 |   |   N_c → N_c + 1;
10 |   E_incident ← {(u, v)|(u, v) ∈ E, u < v};
11 |   foreach (u, v) ∈ E_incident do
12 |   |   v'_correct =correctAfterReposition(v, p_old, p, k);
13 |   |   if u_correct ∧ v_correct ∧ (¬u'_correct ∨ ¬v'_correct) then
14 |   |   |   obj ← obj − w_uv;
15 |   |   else if u'_correct ∧ v'_correct ∧ (¬u_correct ∨ ¬v_correct) then
16 |   |   |   obj ← obj + w_uv;
17 |   end
18 end
19 obj_VNS ← obj + (N_c − |V|) · w_total;
```

Slika 2.8: Parcijalno računanje VNS funkcije cilja za Max-EkPP, gdje je $\mathbf{x}''$ rješenje dobijeno nakon procedure lokalne pretrage, i izabrana pozicija čvora koji se premješta, p kandidatska particija u koju se čvor premješta i k cijeli broj koji odgovara vrijednosti parametra k za Max-EkPP

Tokom parcijalnog izračunanja funkcije cilja, procjenjuje se koliko je vremenski zahtjevna operacija premještanja jednog čvora iz jedne u neku drugu komponentu. Ovo premještanje obično uzrokuje samo promjene u lokalnoj strukturi. Međutim, u najgorem slučaju, koji je malo vjerovatan, kada je komponenta u koju se premješta ili komponenta iz koje se premješta veoma velika (tj. skoro velika kao i cijela mreža), operacija premještanja je vremenski zahtjevna. Takve situacije se dešavaju kada je čvor, koji se premješta, povezan sa većinom čvorova iz komponente, dok je svaki od susjednih

čvorova takođe povezan sa ostalim čvorovima unutar komponente. U najgorem slučaju, vremenska složenost ove procedure je $O(|V|^2)$. Dakle, vremenski najzahtijevnija iteracija unutar lokalne pretrage je kad se particionisanje sastoji od nekoliko velikih komponenti. U tom slučaju, pokušava se premještanje svakog čvora u neku drugu komponentu. Ovo vodi ka $O(|V|^3)$ vremenskoj složenosti za jednu iteraciju lokalne pretrage. Međutim, treba imati u vidu da je vjerovatnoća da će se desiti ovakav slučaj mala.

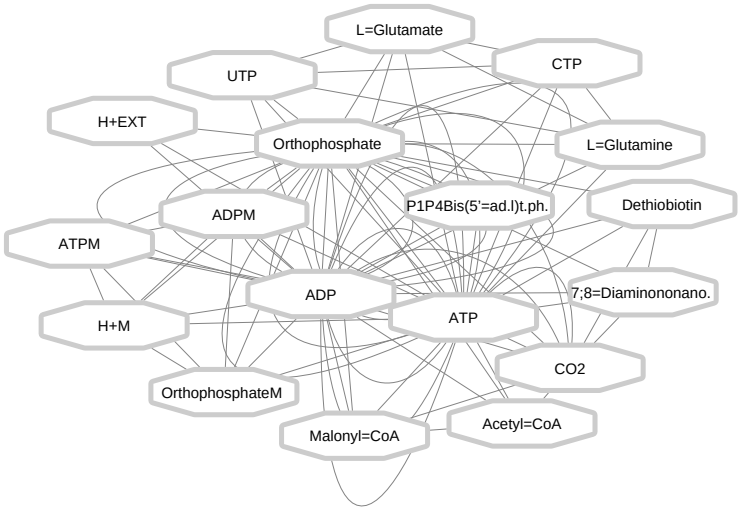
Ako se rješenje ne može više unaprijediti unutar lokalne pretrage, onda se dobijeno rješenje $\mathbf{x}''$ proslijeđuje glavnom dijelu VNS algoritma. Sljedeći korak algoritma je izvođenje procedure `shouldMove()`, koja poredi kvalitet trenutno najboljeg rješenja $\mathbf{x}$ i rješenja $\mathbf{x}''$, dobijenog nakon završetka procedura razmrdavanja i lokalne pretrage. Ako je vrijednost VNS funkcije cilja za rješenje $\mathbf{x}''$ veća od vrijednosti funkcije cilja za rješenje $\mathbf{x}$, onda $\mathbf{x}''$ postaje novo trenutno najbolje rješenje ($\mathbf{x} = \mathbf{x}''$). Ako je vrijednost VNS funkcije cilja za rješenje $\mathbf{x}''$ manje od vrijednosti funkcije cilja za rješenje $\mathbf{x}$, onda rješenje $\mathbf{x}$ ostaje trenutno najbolje. Ako su vrijednosti funkcija cilja za oba rješenja jednake, onda se $\mathbf{x}$ postavlja na $\mathbf{x}''$ sa vjerovatnoćom *prob*.

2.4 Formiranje mreže na osnovu metaboličkih reakcija

Za ovo istraživanje korištene su metaboličke mreže organizma *Saccharomyces cerevisiae*, formirane na osnovu spiska metaboličkih reakcija koji je preuzet iz [46]. Navedene mreže su konstruisane na osnovu genomskih, biohemijskih i fizioloških informacija. Otvoreni okviri čitanja (engl. Open reading frames - ORFs) u genomu i njihovi pridruženi proteini su identifikovani upotrebom genomskih informacija. Biohemijske funkcije identifikovanih enzima su pridružene na osnovu biohemijskih informacija. Fiziološke informacije su osnova za popunjavanje praznina u metaboličkim putanjama i za formulisanje biosintetičke kompozicije u ćeliji. Slično kao i u [51, 98] formirana su dva tipa metaboličkih mreža, one koje predstavljaju interakcije metabolita i one koje predstavljaju interakcije metaboličkih reakcija. U prvom tipu rekonstruisanih mreža, metaboliti su predstavljeni čvorovima, a dva metabolita su povezana granom ako se pojavljuju u najmanje jednoj zajedničkoj reakciji. Težina grane koja povezuje dva metabolita (čvora) jednaka je broju zajedničkih reakcija u kojima ta dva metabolita učestvuju. Čvorovi u drugom tipu rekonstruisanih mreža su metaboličke reakcije, a dvije reakcije su povezane granom ako u njima učestvuje bar jedan zajednički metabolit. Težina grane koja povezuje dvije reakcije (dva čvora) jednaka je broju zajedničkih metabolita koji se pojavljuju u tim reakcijama. Izolovani čvorovi su uklonjeni iz oba tipa mreža.

"ADP" + "ATPM" + "Orthophosphate" -> "ADPM" + "ATP" + "H+M" + "OrthophosphateM"
"Acetyl=CoA" + "ATP" + "CO2" -> "ADP" + "Malonyl=CoA" + "Orthophosphate"
"ADP" + "ATP" -> "Orthophosphate" + "P1,P4=Bis(5'=adenosyl)_tetraphosphate"
"ADP" + "Dethiobiotin" + "Orthophosphate" -> "7,8=Diaminononanoate" + "ATP" + "CO2"
"ATP" -> "ADP" + "H+EXT" + "Orthophosphate"
"ATP" -> "ADP" + "Orthophosphate"
"ATP" + "L=Glutamine" + "UTP" -> "ADP" + "CTP" + "L=Glutamate" + "Orthophosphate"
...
Example:
Metabolites "ADP" and "ATP" participate in 7 common reactions
Metabolites "Orthophosphate" and "CO2“ participate in 2 common reactions

Slika 2.9: Nekoliko reakcija sa kompletnog spiska reakcija, koje se koriste za formiranje mreže



Slika 2.10: Prva faza rekonstrukcije: graf sa paralelnim granama

Primjer 2.2. *U ovom primjeru je prikazana rekonstrukcija mreže na osnovu spiska reakcija. Na Slici 2.9 prikazano je prvih 7 metaboličkih reakcija.*

Sa slike se vidi da metaboliti "ADP" i "ATP" učestvuju u 7 zajedničkih reakcija, dok metaboliti "Orthophosphate" i "CO2" učestvuju u dvije zajedničke reakcije. Na Slici 2.10 je prikazan graf koji predstavlja mrežu formiranu na osnovu ovih 7 reakcija. Graf sa ove slike sadrži paralelne grane između čvorova (metabolita) koji učestvuju u više od jedne zajedničke reakcije.

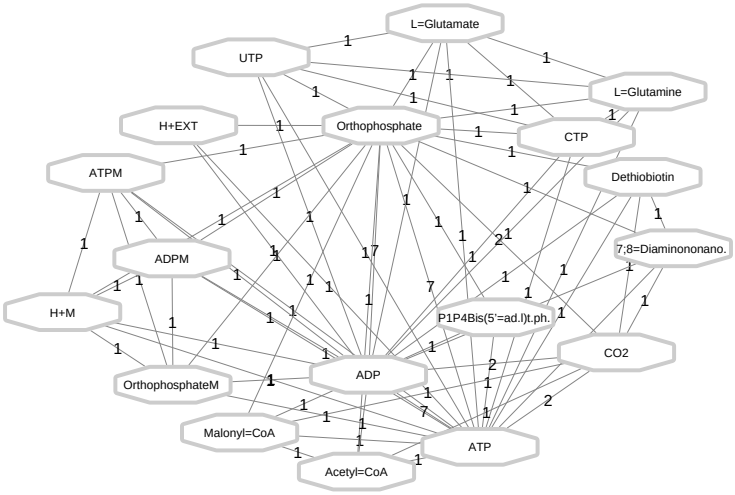
Na Slici 2.11 je prikazan graf u kojem su paralelne grane zamijenjene jednom granom i svakoj grani je pridružena odgovarajuća težina, odnosno broj koji predstavlja broj zajedničkih reakcija u kojima učestvuju krajevi grane (metaboliti).

Za potrebe testiranja predloženog VNS algoritma, za oba tipa mreža formirano je 5 različitih mreža [98], koje se razlikuju po gustini mreže. U prvom skupu mreža SC-NIP-m-tr, $1 \leqslant r \leqslant 5$, svi metaboliti koji nisu singltoni su predstavljeni kao čvorovi i dva čvora su povezana ako se nalaze u najmanje r zajedničkih reakcija, bez obzira sa koje strane se nalaze u reakciji, da li kao reaktanti ili kao proizvodi reakcije. U drugom skupu mreža SC-NIP-r-tm, $1 \leqslant m \leqslant 5$, reakcije predstavljaju čvorove i dva čvora su povezana ako imaju bar m zajedničkih metabolita, bez obzira sa koje strane reakcije se pojavljuju metaboliti. Izolovane reakcije, odnosno reakcije koje nemaju zajedničkih metabolita sa drugim reakcijama nisu razmatrane.

Dakle, ukupno je formirano 10 različitih mreža. U Tabeli 2.1 su prikazane informacije o svim mrežama. Naziv mreže je dat u prvoj koloni. Druga i treća kolona sadrže podatke o broju čvorova i broju grana, respektivno. Informacije o gustini mreže su prikazane u četvrtoj koloni, pri čemu se pod gustinom podrazumijeva odnos broja grana u mreži i broja grana koje bi imala kompletno povezana mreža.

2.5 Rezultati testiranja

Sva testiranja su obavljena na računaru Intel Xeon E5410 CPU @2.33 GHz sa 16 GB RAM i Windows Server 2012 2R 64Bit operativnim sistemom. Za svako izvršenje korišten je samo jedan procesor. Predloženi VNS algoritam je napisan u programskom jeziku C i kompajliran pomoću Visual



Slika 2.11: Druga faza rekonstrukcije: graf sa težinskim granama

Tabela 2.1: Informacije o rekonstruisanim biološkim mrežama

instanca	$\|V\|$	$\|E\|$	gustina
SC-NIP-m-t1	991	4161	0.008482402
SC-NIP-m-t2	602	1520	0.008402386
SC-NIP-m-t3	177	269	0.017270159
SC-NIP-m-t4	129	166	0.020106589
SC-NIP-m-t5	75	84	0.03027027
SC-NIP-r-t1	1393	56276	0.058044739
SC-NIP-r-t2	1183	17776	0.02542505
SC-NIP-r-t3	663	1782	0.00812019
SC-NIP-r-t4	377	321	0.004529037
SC-NIP-r-t5	45	27	0.027272727

Studio 2015 kompajlera.

Da bi poređenja sa postojećim metodom cjelobrojnog linearnog programiranja bila, što je moguće korektnija, korišten je isti skup podataka kao i u [98] i testiranja su izvršena za vrijednosti $k \in \{1, 2, 3\}$. Pored toga, ILP model iz [98] je implementiran i testiran pod istim uslovima kao i predloženi VNS. Vrijeme izvršenja (engl. CPU clock speed) implementiranog ILP model je manje nego u [98], a mogući razlog za to je upotreba novije verzije CPLEX-a. Zbog svega navedenog, rezultati prikazani u Tabelama 2.2, 2.3 i 2.4 su nešto malo drugačiji od rezultata iz rada [98].

Prva dva skupa instanci nad kojima je testiran algoritam su biološke mreže SC-NIP-m-tr i SC-NIP-r-tm, čije formiranje je opisano u Odjeljku 2.4. Treći skup instanci su poznate DIMACS instance iz literature, dostupne na adresi
http://www.dcs.gla.ac.uk/~pat/maxClique/distribution/DIMACS_cliques/. Zapravo, testirana su dva skupa DIMACS instanci. U prvom skupu su instance iz [98], odnosno DIMACS instance sa manje od 100 čvorova, kao i veće instance sa manje od 200 čvorova čija je gustina najviše 0.25. Drugi skup su preostale 73 DIMACS instance. S obzirom da originalne DIMACS instance nisu težinske, u ovom istraživanju je primijenjen princip računanja težina iz rada [122], a koji je također primjenjen

i u radovima [98, 51]. Težina w_{ij} grane koje spaja čvorove i i j se računa po formuli $w_{ij} = ((i + j) \bmod 200) + 1$.

Za svaku instancu, VNS je izvršen 10 puta sa različitim inicijalnim podešavanjem generatora pseudoslučajnih brojeva. Algoritam se zaustavlja ako je ispunjen jedan od sljedećih uslova: $it_{max} = 20000$, $itrep_{max} = 10000$ ili $t_{max} \geqslant 3600$ sekundi. Parametar n_{min} je postavljen na 1. U ranoj fazi pretrage rješenja mali broj perturbacija obično obezbjeđuje poboljšanje, tako da se algoritam relativno brzo vraća na minimalnu vrijednost od n, odnosno na n_{min}. Kako se pretraga nastavlja, VNS istražuje širi prostor pretrage. Dakle, važno je adekvatno postaviti vrijednost parametra n_{max}. U cilju da se postigne uniformno postavljanje parametra kroz sve grafovske instance, za svaku instancu sa $|V|$ čvorova, vrijednost n_{max} se postavlja na $\min\{|V|, 80\}$. U navedenom izboru između minimalne vrijednosti između broja čvorova i konstante 80, pored vrijednosti 80, razmatrane su i vrijednosti iz skupa $\{10, 20, 50, 100\}$, ali sve su se pokazale manje efikasnim od 80. Vjerovatonosni parametar *prob* je takođe povezan sa kompromisom između eksploatacije pojedinih područja pretrage i raspršenosti pretrage. Postavljanje parametra *prob* na 0.1 znači da ako je novo rješenje istog kvaliteta kao i trenutno najbolje, vjerovatnoća prelaza u novo rješenje je 10%. Veće vrijednosti ovog parametra čine algoritam manje stabilnim i svakako povećavaju raspršenost pretrage, ali i redukuju intenzitet pretraživanja u okolini trenutnog rješenja. I ova vrijednost je izabrana empirijski nakon testiranja vrijednosti iz skupa $prob \in \{0.1, 0.5, 0.9\}$.

Da bi se ispitala stabilnost VNS-a, za svaku instancu je izračunata prosječna relativna greška (engl. average gap) na osnovu formule $agap = \frac{1}{10}\sum_{i=1}^{10} gap_i$, pri čemu je $gap_i = 100 * \frac{Sol.res - sol_i}{Sol.res}$ i $Sol.res$ je jednako optimalnom rješenju ako je poznato, a u suprotnom najboljem poznatom rezultatu. sol_i je VNS rezultat dobijen u i-tom izvršenju. Kao što je već navedeno, ukupan broj izvršenja je 10.

2.5.1 Rezultati dobijeni za SC-NIP-m-tr instance

Rezultati dobijeni na skupu SC-NIP-m-tr instanci su prikazani u Tabeli 2.2. Ako je najbolje dostignuto VNS rješenje jednako poznatom optimalnom rješenju, onda se u koloni VNS_{best} nalazi oznaka *opt*. Ako optimalno rješenje nije poznato, onda je prikazan najbolji VNS rezultat. Oznaka (*) je postavljena ako nema prethodnih rezultata za razmatranu instancu. U posljednje dvije kolone prikazani su rezultati ILP metoda dobijeni pod istim uslovima, pri čemu su označeni sa *opt* ako je pronađeno optimalno rješenje. Ako ILP metod nije pronašao nijedno rješenje zbog memorijskih ograničenja, u kolonama se nalazi neka od oznaka "–" i o.m (out of memory).

Iz Tabele 2.2 se vidi da je predloženi VNS algoritam pronašao svih 10 poznatih optimalnih rješenja. Za ostalih 5 instanci, VNS je uspio da za razumno vrijeme (manje od jednog sata) pronađe rješenje. U slučaju instance SC-NIP-m-t3 i za vrijednost parametra $k = 3$, rezultat ILP-a nije verifikovan kao optimalan u vremenskom okviru od 3 sata, a oba metoda su dostigla isti rezultat. Prosječna relativna greška je prilično mala i manja od 1 za sve instance, što ukazuje da je VNS stabilan pri rješavanju ove klase instanci.

2.5.2 Rezultati dobijeni za SC-NIP-r-tm instance

Rezultati dobijeni pri testiranju algoritma nad SC-NIP-r-tm instancama su prikazani u Tabeli 2.3, koja je organizovana na sličan način kao i Tabela 2.2. Za 7/15 instanci VNS pronalazi svih 7 poznatih optimalnih rezultata. Za preostalih 8 instanci VNS pronalazi nova najbolja rješenja. Ako se posmatra odnos kvaliteta rješenja, situacija je slična kao i za SC-NIP-m-tr instance. ILP metod

Tabela 2.2: Rezultati testiranja dobijeni za SC-NIP-m-tr instance

k	instanca	*opt*	VNS_{best}	VNS_{avg}	VNS_{gap}	VNS_t^{tot}	ILP	ILP_t
1	SC-NIP-m-t1	1866	*opt*	1864	0.11	3600.22	*opt*	208.03
1	SC-NIP-m-t2	1538	*opt*	1538	0	1072.51	*opt*	2.63
1	SC-NIP-m-t3	910	*opt*	910	0	92.96	*opt*	0.45
1	SC-NIP-m-t4	831	*opt*	831	0	45.5	*opt*	0.06
1	SC-NIP-m-t5	723	*opt*	723	0	15.73	*opt*	0.63
2	SC-NIP-m-t1	-	2151*	2147.3	0.17	3600.14	-	o.m.
2	SC-NIP-m-t2	-	1773*	1771.8	0.07	1495.49	-	o.m.
2	SC-NIP-m-t3	1021	*opt*	1021	0	100.74	*opt*	60.06
2	SC-NIP-m-t4	907	*opt*	907	0	54.75	*opt*	10.77
2	SC-NIP-m-t5	801	*opt*	801	0	16.42	*opt*	1.75
3	SC-NIP-m-t1	-	2353*	2337.1	0.68	3600.18	-	o.m.
3	SC-NIP-m-t2	-	1943*	1939.4	0.19	1988.38	-	o.m.
3	SC-NIP-m-t3	-	1141	1141	0	121.08	1141	>10800
3	SC-NIP-m-t4	1022	*opt*	1022	0	69.79	*opt*	1354.25
3	SC-NIP-m-t5	887	*opt*	887	0	17.62	*opt*	34.2

je pronašao 7 optimalnih rezultata: četiri optimalna rješenja za $k = 1$, dva optimalna rješenja za $k = 2$ i jedno optimalno rješenje za $k = 3$. VNS je pronašao sva ova optimalna rješenja, a takođe je pronašao i rezultate za ostale instance. Za instancu SC-NIP-r-t1 i vrijednost parametra $k = 1$, ILP metod je nakon izvršavanja koje je trajalo tri sata pronašao rješenje čija je vrijednost 1317, što je značajno manja vrijednost od vrijednosti rješenja dobijenog VNS-om.

Posmatrajući dati problem sa računarske strane, može se primijetiti da su SC-NIP-r-tm instance zahtjevnije od SC-NIP-m-tr instanci, što se i može vidjeti iz Tabele 2.1. Zato je i vrijeme izvršavanja algoritma nad ovim instancama proporcionalno veće nego vrijeme izvršenja za SC-NIP-m-tr instance. Za pet SC-NIP-r-tm instanci izvršenje algoritma je zaustavljeno nakon što je dostignuto vremensko ograničenje od jednog sata, dok je za druge instance algoritam zaustavljen nakon što je dostignut maksimalan broj iteracija. Prosječna relativna greška je i za ovu klasu instanci uglavnom mala, manja od 1 za sve instance. Iz Tabela 2.2 i 2.3 se može zaključiti da za obje klase bioloških instanci, vrijeme izvršenja zavisi od gustine grafa, odnosno da je vrijeme izvršenja manje za grafove manje gustine. Prirodno objašnjenje za ovo je da manji broj grana uzorkuje manji broj ukupnih izvođenja procedure lokalne pretrage, što dalje vodi ka tome da je vrijeme potrebno za izvršenje kompletnog algoritma manje. Poredeći vrijednosti iz kolona *opt* i VNS_{best} za iste instance i različite vrijednosti parametra k, može se zaključiti da se vrijednost funkcije cilja povećava ako se poveća vrijednost parametra k. To se dešava jer se relaksacijom uslova za spajanje u klastere povećava ukupan broj grana koje se dodaju u klaster.

2.5.3 Rezultati dobijeni za DIMACS instance

Rezultati dobijeni pri testiranju DIMACS instanci su prikazani u Tabeli 2.4. Sve instance iz ove tabele pripadaju c-fat, MANN, hamming i johnson grupama instanci, koje se često koriste u literaturi pri rješavanju problema pronalaženja maksimalne klike. Više informacija o ovim instacama se može naći u radu [71]. Iz Tabele 2.4 se vidi da je predloženi VNS algoritam pronašao svih 10

Tabela 2.3: Rezultati testiranja dobijeni za SC-NIP-r-tm instance

k	instanca	*opt*	VNS_{best}	VNS_{avg}	VNS_{gap}	VNS_t^{tot}	ILP	ILP_t
1	SC-NIP-r-t1	-	57681	57544.6	0.24	3607.77	1317	>10800
1	SC-NIP-r-t2	34576	*opt*	34561.6	0.04	3601.2	*opt*	56.78
1	SC-NIP-r-t3	5411	*opt*	5411	0	1550.95	*opt*	4.19
1	SC-NIP-r-t4	1232	*opt*	1232	0	327.82	*opt*	0.09
1	SC-NIP-r-t5	140	*opt*	140	0	3.71	*opt*	0.3
2	SC-NIP-r-t1	-	57729*	57496	0.4	3602.58	-	o.m.
2	SC-NIP-r-t2	-	34592*	34563.6	0.08	3601.65	-	o.m.
2	SC-NIP-r-t3	-	5423*	5423	0	1569.11	-	o.m.
2	SC-NIP-r-t4	1245	*opt*	1245	0	331.75	*opt*	103.42
2	SC-NIP-r-t5	140	*opt*	140	0	3.82	*opt*	0.22
3	SC-NIP-r-t1	-	57775*	57587.4	0.33	3602.19	-	o.m.
3	SC-NIP-r-t2	-	34641*	34572.5	0.2	3601.26	-	o.m.
3	SC-NIP-r-t3	-	5465*	5465	0	1496.84	-	o.m.
3	SC-NIP-r-t4	-	1245*	1245	0	327.45	-	o.m.
3	SC-NIP-r-t5	140	140	140	0	3.84	*opt*	0.91

poznatih optimalnih rješenja. U preostalih 11 slučajeva, VNS pronalazi najbolja poznata rješenja. Rezultati dobijeni ILP metodom se neznato razlikuju od rezultata iz [98], zbog različitih verzija CPLEX rješavača. Da bi se ispitala efikasnost predloženog VNS algoritma izvršeno je testiranje i na preostale 73 instance, koje su veće. Iako se Max-EkPP uglavnom razmatra na rijetkim grafovima, da bi se kompletirao pristup prikazan u ovom istraživanju predloženi VNS je primijenjen i na gušće DIMACS instance. Do sada nisu poznati rezultati za ove instance u literaturi. Iako se optimalnost ne može dokazati, male vrijednosti prosječne relativne greške ukazuju na to da je VNS pronašao visoko kvalitetne rezultate. Rezultati su prikazani u Tabelama 2.5, 2.6 i 2.7.

Činjenica da je VNS algoritam pronašao rješenje za sve 73 velike DIMACS instance ukazuje na visok nivo skalabilnosti algoritma. Poređenje skalabilnosti između aproksimativnog VNS algoritma i tačnog ILP algoritma ne može se uraditi na potpuno ravnopravan način, prije svega zbog razlika u tipu izlaznog rezultata, ILP rješavač može da garantuje optimalnost dobijenog rezultata dok aproksimativni algoritam ne može. Važno je primijetiti da je VNS za Max-EkPP implementiran na efikasan način, zbog same prirode VNS metaheuristike. Efikasnost je dodatno poboljšana uvođenjem parcijalnog računanja funkcije cilja. Za razliku od ILP rješavača, VNS koristi manje memorije jer je najveća struktura podataka koja je potrebna tokom izračunavanja matrica incidencije veličine $|V|^2$.

2.6 Vizuelizacija i biološko obrazloženje dobijenih rezultata

Sa biološkog aspekta je analizirana najveća *S. cerevisiae* metabolička mreža SC-NIP-m-t1. Primjenom predloženog algoritma na pomenutu mrežu, dobijeno je nekoliko korisnih informacija. Prvo, dobijeni *k-plex*-i imaju biološko značenje, odnosno predstavljaju važne metaboličke procese. Dalje, varirajući vrijednosti parametra k, primjećuje se da relaksacija zahtjeva klasterovanja vodi ka tome da se dobija više informacija sa biološke tačke gledišta. Pored navedenog, *k-plex*-i dobijeni predloženim VNS-om mogu biti predmet dalje analize alatima za obogaćivanje informacijama, što

Tabela 2.4: Rezultati na manjim i rjeđim DIMACS instancama

k	instanca	*opt*	VNS_{best}	VNS_{avg}	VNS_{gap}	VNS_t^{tot}	ILP	ILP_t
1	c-fat200-1	98711	*opt*	98711	0	234.43	*opt*	2.63
2	c-fat200-1	98711	*opt*	98543.2	0.17	202.87	*opt*	157.98
3	c-fat200-1	-	98711	98571.8	0.14	193.7	95878	>10800
1	c-fat200-2	213248	*opt*	213246.8	0	540.89	*opt*	2.66
2	c-fat200-2	213248	*opt*	212194.6	0.49	360.5	*opt*	239.99
3	c-fat200-2	-	213248	211143.8	0.99	292.97	10200	>10800
1	hamming6-2	65472	*opt*	65472	0	114.53	*opt*	2.2
2	hamming6-2	-	65472	65472	0	61.91	63360	>10800
3	hamming6-2	-	65472	65472	0	46.15	52423	>10800
1	hamming6-4	6336	*opt*	6336	0	53.29	*opt*	1.5
2	hamming6-4	-	8184	8184	0	74.81	7758	>10800
3	hamming6-4	-	10560	10560	0	77.57	8014	>10800
1	johnson8-2-4	1260	*opt*	1260	0	7.63	*opt*	0.25
2	johnson8-2-4	-	1365	1363.5	0.11	10.41	1363	o.m.
3	johnson8-2-4	-	1996	1996	0	7.34	1996	o.m.
1	johnson8-4-4	-	27874	27874	0	169.18	25848	>10800
2	johnson8-4-4	-	31320	31147.2	0.55	124.87	11231	>10800
3	johnson8-4-4	-	37096	35910.3	3.2	155.73	9751	>10800
1	MANN_a9	14868	*opt*	14865	0.02	27.55	*opt*	924.61
2	MANN_a9	23055	*opt*	23053.8	0.01	25.96	*opt*	800.75
3	MANN_a9	33660	*opt*	33660	0	14.23	*opt*	185.39

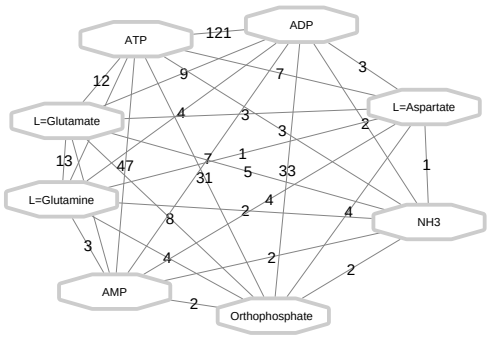
je i pokazano u Odjeljku 2.6.1. U Odjeljku 2.6.4 je prikazan test slučaj kako se proces klasterovanja može upotrijebiti za poboljšanje veza u metaboličkoj ontologiji.

Kao rezultat primjene predloženog VNS algoritma na instancu SC-NIP-m-t1, dobijeni su *k-plex*-i koji predstavljaju različite metaboličke procese. U nastavku su detaljnije opisani sljedeći procesi: degradacija aminokiselina, sinteza masnih kiselina, sinteza vitamina B6, oksidacija sukcinata do fumarata i oksidacija formalaldehida. Da bi se potvrdila pouzdanost dobijenih rezultata, pojedine informacije biohemijskih putanja razmatranog organizma *S. cerevisiae* su upoređene sa podacima predstavljenim u Yeast Pathways Database [164].

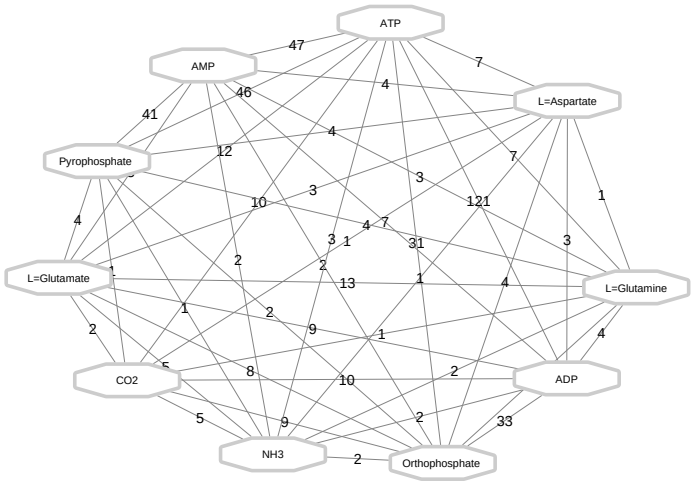
2.6.1 Proces degradacije aminokiselina

Na Slici 2.12 prikazan je najveći klaster dobijen za vrijednost parametra $k = 1$, koji predstavlja grubi prikaz procesa degradacije aminokiselina. Amonijak, koji je prisutan u organizmu, koristi se kao izvor azota za sintezu aminokiselina, a ako se oslobodi u većim količinama mora se degradirati kroz različite metaboličke puteve, zbog svoje toksičnosti. U razmatranom organizmu *S. cerevisiae*, amonijak se može ugraditi u amino grupu glutamata na dva načina: reduktivnom aminacijom 2-ketoglutarata, koja je katalizovana glutamat dehidrogenazom gdje NADPH služi kao izvor elektrona ili ATP zavisnom sintezom glutamina iz glutamata i amonijaka katalizovanog sintezom glutamina [94]. Klaster prikazan na Slici 2.12 je klika sa 8 čvorova i sadrži glavne međuproizvode (intermedijere) koji figurišu u sintezi amonijaka iz glutaminske i asparatinske kiseline. Glutamat veže ortofosfornu grupu iz ATP, čime nastaje glutamin, formira se ADP, a ortofosfat se oslobađa.

Na Slici 2.13 je prikazan najveći klaster dobijen za vrijednost parametra $k = 2$. Kao što se



Slika 2.12: Proces degradacije aminokisleina: slučaj $k = 1$



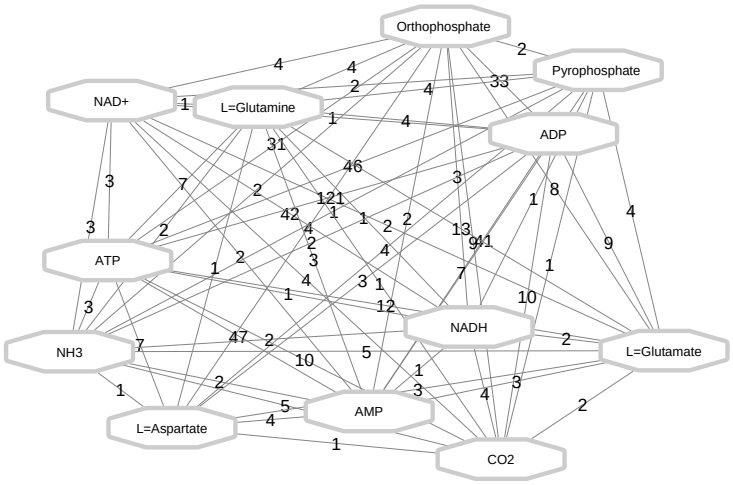
Slika 2.13: Proces degradacije aminokisleina: slučaj $k = 2$

može vidjeti, glutamat se ponovo pojavljuje u reakciji glutamina i L-aspartata uz potrošnju ATP-a. Kao rezultat nastaje asparagin, koji se potom konvertuje u aspartat deaminacijom, dok se amonijak oslobađa. Amonijak, takođe, nastaje i deaminacijom glutamina. Ako bi se opisani sistem proširio sa dva dodatna međuproizvoda (intermedijera), mogao bi da predstavlja još jedan način sinteze glutamata, tačnije reakciju CO_2 i glutamina ponovo uz potrošnju ATP-a.

Detaljniji prikaz je dat klasterom, koji se nalazi na Slici 2.14 i koji je dobijen particionisanjem pod manje strožim uslovima, tačnije za vrijednost parametra $k = 3$. Na ovoj slici je prikazan proces oksidativne deaminacije, koji se dešava u ćeliji uključujući aminokiselinu glutamat. Glutamat se oksidativno deaminizuje uz učešće enzima glutmat dehidrogenaze, koristeći NAD ili NADP kao koenzime. Ovim procesom se sintetišu dva toksična produkta: hidrogen peroksid i amonijak. VNS algoritam je grupisao sve ove međuproizvode (intermedijere) u jedan klaster, što nije bio slučaj u situacijama sa strožim uslovima (slučajevi $k = 1$ i $k = 2$).

Razmatranje *k-plex*-a alatima za obogaćivanje informacijama

U cilju daljeg utvrđivanja pouzdanosti i korisnosti predloženog metoda, razmatrana je relevantnost nekih *k-plex*-a alatom za obogaćivanje informacijama BiNChE [105], koji je baziran na ChEBI ontologiji [58]. Ulazni podatak za BiNChE alat je lista molekula, a kao rezultat se dobija usmjeren graf u kojem su označeni molekuli koji su značajniji za biohemijski proces. U testnom slučaju kao ulazna lista zadata je lista metabolita iz klastera prikazanog na Slici 2.14. Izlazni graf koji je dobijen



Slika 2.14: Proces degradacije aminokisleina: slučaj $k = 3$

BiNChE alatom i prikazan je na Slici 2.15 potvrđuje da su svih 12 metabolita, koji su grupisani u *k-plex*, značajni za ovaj biohemijski proces. Sa Slike 2.15 se vidi da su L-aspartat (engl. L-Aspartat) i L-glutamat (engl. L-Glutamat) prepoznati kao polarni aminokiselinski cviter joni (engl. zwitterions), što omogućava aminokiselinama da učestvuju u mnogim metaboličkim reakcijama. BiNChE alatom su ADP i ATP prepoznati kao adenozin 5'-fosfat ili adenozin monofosfat (engl. adenosine 5'-phosphates ili adenosine monophosphates) i kao purinski ribonukleotidi (engl. purine ribonucleotides). Adenin (engl. Adenine) koji je komponenta adenozin 5'-monofosfata je purinska baza, pa je identifikovan kao i ribonukleozid 5'-monofosfat (engl. ribonucleoside 5'-monophosphate). Nukleozidi proizvode nukleotide - ribonukleotide koji sadrže fosforilisani šećer ribozu sa 5C atoma. Dakle, ATP i ADP su u semantičkoj vezi sa adenozin 5'-fosfatom, purinskim ribonuklezidom 5'-monofosfatom i ribonukleozidom 5'-fosfatom i svi ovi molekuli su prepoznati kao značajni u BiNChE alatu. Pored navedenog BiNChE alatom je pokazano da oksid, amonijak i CO_2 pripadaju klasi molekula gasa koji se sastoje od heteroatoma. Oksid se može pojaviti ili kao oksoanjon ili kao organski oksid. Ova kratka analiza ukazuje da su metaboliti grupisani u isti *k-plex* semantički povezani. Upotreba ovog alata za obogaćivanje informacijama omogućava verifikaciju i bolje objašnjenje rezultata dobijenih predloženim VNS algoritmom, dajući tako širu sliku mogućih transformacija između metabolita.

2.6.2 Sinteza masnih kiselina

Na Slici 2.16 prikazan je drugi po veličini klaster dobijen za vrijednost parametra $k = 1$. Sa slike se može primijetiti da je algoritam grupisao intermedijere koji se pojavljuju u procesu sinteze masnih kiselina. Masne kiseline su dugački molekuli i proces njihove sinteze se može podijeliti u tri faze. U prvoj fazi se vrši sinteza malonil koenzima A (engl. malonil-CoA) iz acetil koenzima A (engl. acetil-CoA), jer je malonil koenzim A mnogo reaktivniji molekul i pogodniji za produženje lanca masnih kiselina. Acetil koenzim A se sintetiše iz koenzima A (engl. CoA) uz potrošnju ATP koji oslobađa ortofosfat i postaje ADP. Sa Slike 2.17 može se vidjeti da je grana koja povezuje acetil koenzim A i malonil koenzim A težine 2, jer je ova reakcija povratna i njihova veza se broji dva puta. Druga faza se sastoji od pet uzastopnih cikličih reakcija, počevši od vezivanja acetil koenzima A i malonil koenzima A direktno za prenosni protein (engl. carrier protein), nakon čega se formira malonil-acil prenosni protein (engl. malonil-acyl carrier protein (ACP)) i oslobađa koenzim A.

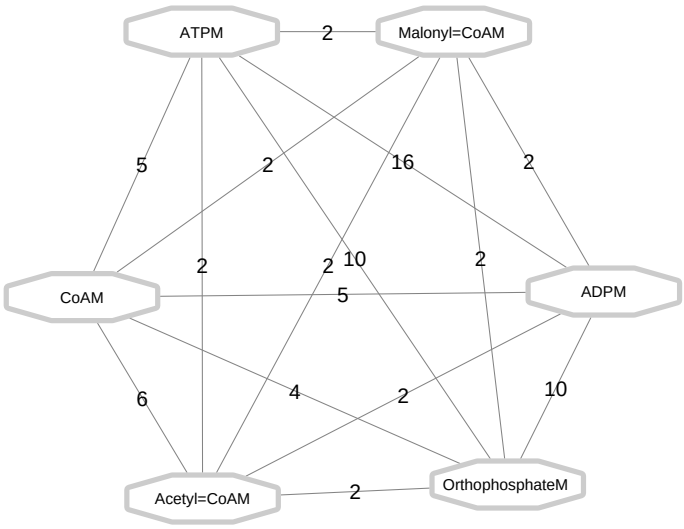
Na Slici 2.18 prikazana je reakcija vezivanja aminokiselina za acil prenosni protein (engl. acyl

Slika 2.15: Graf dobijen BiNChE alatom za *k-plex* sa slike 2.14

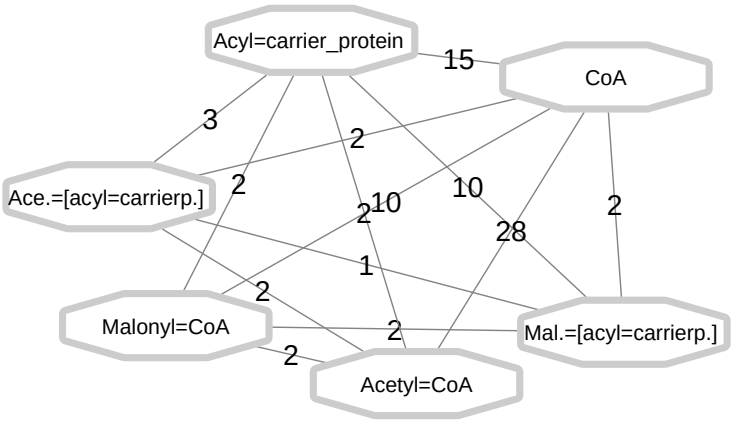
carrier protein (ACP)). Može se zaključiti da se najveći broj reakcija odnosi na sintezu acetil koenzima A iz koenzima A i na vezivanje malonil koenzim A za prenosni protein, što dokazuje da je algoritam prepoznao kompletan sistem biosinteze. Reakcija se nastavlja vezivanjem 2C atoma za lanac (formira se novi malonil koenzim A) sve dok se ne završi sintetisanje dugačkog lanca masnih kiselina. Svaki put kada se malonil koenzim A veže za ACP, koenzim A se oslobađa. Tokom kondenzacije sa ACP, CO_2 se oslobađa i pojavljuju se oksidativne komponente, koje se redukuju prisustvom NADPH (transformisanog u NADP+) i hidrolizuju se u enoil jedinjenja (engl. enoyl compounds) (ponovo redukovane sa NADPH). U trećoj fazi zasićena produžena masna kiselina prihvata novi malonil koenzim A i nastavlja se dalje produženje lanca već opisanom šemom (Slika 2.18).

2.6.3 Ostala korisna saznanja

Za vrijednost parametra $k = 2$ dobijena su još dva interesantna klastera, koji predstavljaju biološke procese sinteze vitamina B6 i oksidacije formaldehida zavisne od glutationa. Glavni in-



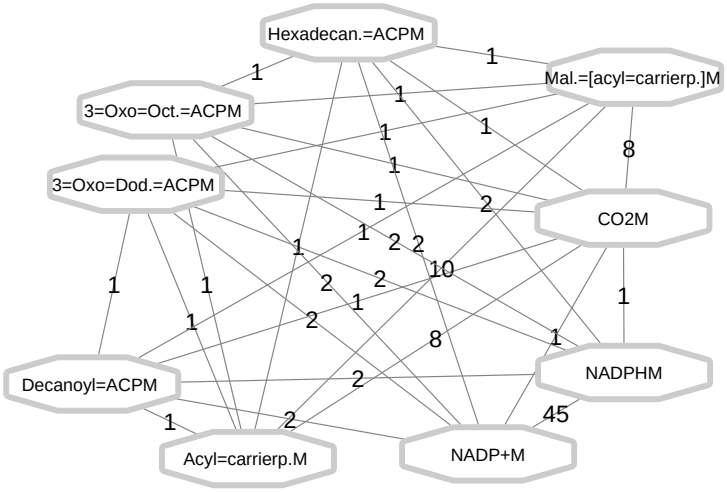
Slika 2.16: Sinteza masnih kiselina: slučaj $k = 1$



Slika 2.17: Sinteza masnih kiselina: slučaj $k = 2$

termedijeri za sintezu vitamina B6 su prikazani na Slici 2.19. Pirodoksal fosfat (engl. Pyridoxal phosphate (PLP)) je aktivna forma vitamina B6 i kofaktor u mnogim reakcijama metabolizma aminokiselina [164]. Dati grafovski prikaz ukazuje da je algoritam grupisao različite forme vitamina B6: pirodoksin (engl. Pyridoxine), pirodoksal (engl. pyridoxal (PL)) i pirodoksin 5'-fosfat (engl. pyridoxine 5'-phosphate (PNP)). *S. cerevisiae* sintetiše PLP kroz fungalni *de novo* tip PLP sintetičkog puta i put održavanja minimalne koncentracije ovog molekula. Kroz ove biohemijske putanje PLP može se dobiti iz PL ili sintezom iz pirodoksina. Ovaj proces se sastoji iz dvije faze, u prvoj fazi se pirodoksin 5'-fosfat sintetiše iz pirodoksina aktivacijom enzima pirodoksin kinaze (engl. pyridoxine kinase). Druga faza se bazira na okcidaciji pirodoksin 5'-fosfat u pirodoksal fosfat. Ova reakcija zahtijeva učešće kiseonika, koji se redukuje do peroksida (engl. peroxide) u ovoj reakciji.

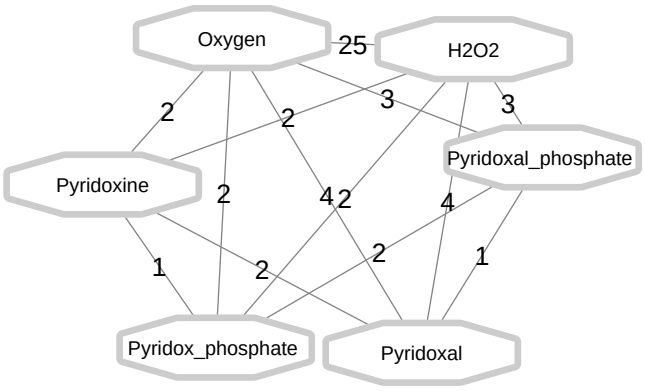
Proces uklanjanja veoma reaktivnog i toksičnog formaldehida je prikazan na Slici 2.20. Iako se formaldehid ne može metabolisati iz metanola (engl. methanol) u *S. cerevisiae*, može se nadograditi iz biljnog materijala ili iz zagađenog vazduha i vode [164]. Zbog navedenih razloga, potreban je sistem za uklanjanje ovog toksičnog jedinjenja. Metabolit koji ima važnu ulogu u kompletnom odbrambenom



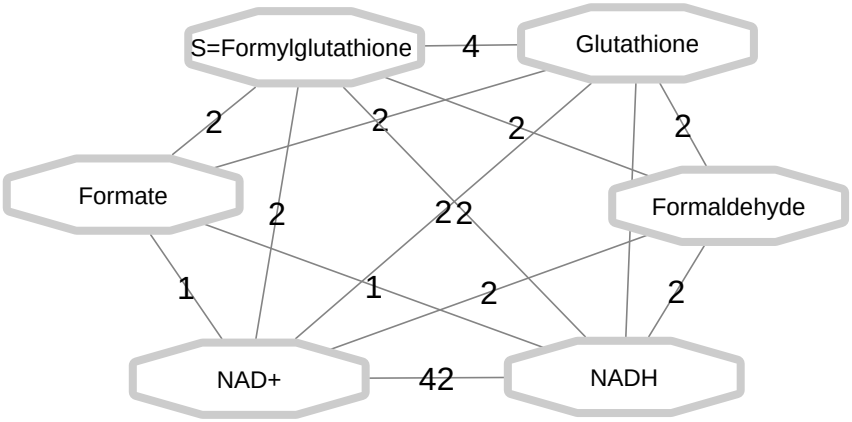
Slika 2.18: Sinteza masnih kiselina: slučaj $k = 3$

sistemu je glutation (engl. glutathione), koji ima sposobnost da veže formaldehid u spontanoj reakciji. Rezultujući S-hidroksilmetilglutation (engl. S-hydroxymethylglutathione) je oksidovan do S-formil-glutationa (engl. S-formyl-glutation) uz učešće NAD+ kao oksidacionog sredstva, koji se pri tome redukuje do NADH. Hidrolizom ovog jedinjenja nastaju glutation i netoksični format (engl. formate).

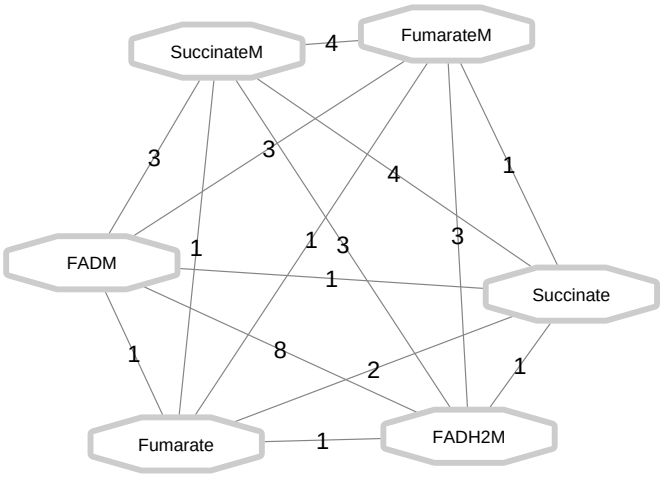
Na Slici 2.21 prikazan je proces oksidacije sukcinata do fumarata. Ova reakcija je moguća uz učešće enzima sukcinat dehidrogenaze, koji je kovalentno vezan za flavin adenin dinukleotid (engl. flavin adenine dinucleotide (FAD)), koji djeluje kao akceptor za jon vodonika (engl. hydrogen ion acceptor), pri čemu se redukuje do $FADH_2$. Svi intermedijeri koji su uključeni u ovu reakciju su prisutni čak i u klasteru dobijenom za vrijednost parametra $k = 1$, pa dalje relaksacije uslova ne mogu dodati nove elemente. Algoritam je prepoznao ovu situaciju i isti graf je dobijen kao rezultat za sve vrijednosti parametra k.



Slika 2.19: Sinteza vitamina B6



Slika 2.20: Uklanjanje formalaldehida

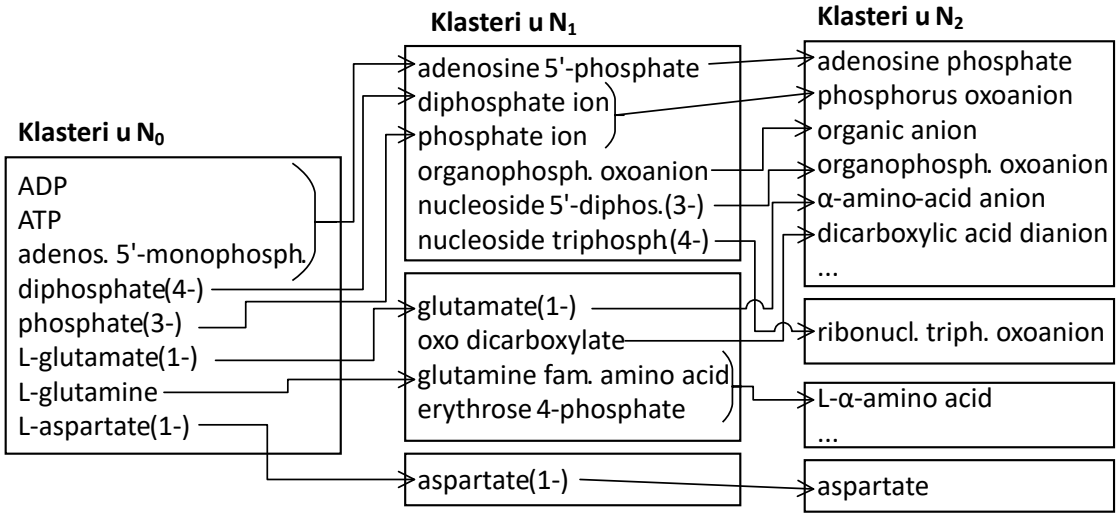


Slika 2.21: Oksidacija sukcinata do fumarata

2.6.4 Test slučaj integracije metaboličke ontologije sa procesom klasterovanja

Da bi se dalje ispitala korisnost predloženog algoritma, isti je primjenjen na niz metaboličih mreža formiranih u odnosu na metaboličku ontologiju [58]. Za ovaj testni slučaj izabrana je metabolička mreža SC-NIP-m-t3, u kojoj su čvorovi metaboliti organizma *S. cerevisiae* koji se pojavljuju u bar 3 reakcije. Ova mreža sadrži većinu najznačajnih metabolita, koji se često pojavljuju u metaboličkim procesima. Proces klasterovanja je ponovljen na SC-NIP-m-t3 mreži za vrijednost parametra $k = 3$ i identifikovan je najveći klaster. Na osnovu ove mreže, koja je označena sa N_0, formirana je "roditeljska" N_1 mreža, koja sadrži roditeljske čvorove čvorova iz N_0 mreže. Drugim riječima, svaki metabolit iz mreže N_0 je mapiran svojim roditeljskim čvorom u ontologiji. Ako dva čvora imaju isti roditeljski čvor, oni su spojeni u jedan čvor. Grane i težine grana iz početne mreže su takođe mapirani. Ako postoji grana između metabolita A i B težine w_{AB}, onda postoji grana i između njihovih roditeljskih čvorova P_A i P_B težine w_{AB}. Ako su pri tome P_A i P_B, takođe, roditelji i metabolita C i D respektivno, koji su povezani granom čija je težina w_{CD}, onda je težina grane između P_A i P_B jednaka sumi težina w_{AB} i w_{CD}. Zatim je proces ponovljen još jednom i formirana je N_2 mreža, koja sadrži roditeljske čvorove čvorova iz N_1 mreže. Na mreže N_1 i N_2 je takođe primijenjen VNS algoritam koji određuje *k-plex*-e. Neki od dobijenih klastera su prikazani na Slici

2.22. Sa slike se vidi da je 5 od 8 roditelja metabolita iz početnog klastera grupisano u isti klaster N_1 mreže, 2 roditelja se takođe nalaze u zajedničkom klasteru, dok jedan roditelj (aspartate 1-) se nalazi u pojedinačnom klasteru. Dalje, primjećuje se da su roditelji 5 od 6 metabolita iz prvog i 2 od 4 metabolita iz drugog klastera mreže N_1 grupisani u isti klaster N_2 mreže. Ova analiza pokazuje "konzistentnost" dobijenih klastera: Ako se čvorovi v_1, v_2,..., v_p pripadaju istom k-*plex*-u u mreži N_i $(i = 0, 1)$, onda će, u velikom broju slučajeva, i njihovi roditeljski čvorovi pripadati istom k-*plex*-u u mreži N_{i+1}. Dakle, može se pretpostaviti da postoji sematička povezanost između čvorova u istom klasteru u odnosu na metaboličku ontologiju. Ova saznanja su takođe provjerena i pomoću BiNChE alata za obogaćivanje informacijama. Ako je ulazni klaster iz N_0 mreže, onda se BiNChE alatom dobija graf u kojem su većina roditeljskih čvorova iz mreža N_1 i N_2 značajni. Ova činjenica otvara mogućnost poboljšanja odnosa u ontologiji razmatranjem odnosa između serije k-*plex*-a iz mreža N_0, N_1 i N_2.



Slika 2.22: Niz klastera za mreže N_0, N_1 i N_2, instance SC-NIP-m-t3 i $k = 3$. Strelice pokazuju od potomka ka roditeljskom čvoru.

2.7 Završna razmatranja

Dobijeni rezultati su pokazali da je predloženi VNS algoritam pronašao sva poznata optimalna ili najbolja rješenja posmatranog optimizacionog problema rješavanog nad biološkim i sintetičkim instancama iz literature. Takođe, pronašao je nova visoko kvalitetna rješenja za ostale, ranije neraz-matrane instance, u razumnom vremenu. Kroz dublju analizu klastera identifikovanih za različite vrijednosti parametra k nad biološkim metaboličkim instancama, potvrđeno je da algoritam pronalazi mnoge klastere u kojima su intermedijeri semantički povezani. Relaksacija uslova za particionisanje vodi ka dobijanju korisnijih klastera, što pomaže pri otkrivanju novih bioloških odnosa ili potvrđivanju postojećih.

U daljim istraživanjima bilo bi zanimljivo primijeniti predloženi VNS algoritam za rješavanje sličnih problema particionisanja mreža, koji imaju primjenu u biološkim ali i nekim drugim istraživanjima. Pored toga, mogla bi se razmatrati i paralelizacija predloženog VNS algoritma, kao i pokretanje na višeprocesorskim sistemima.

Tabela 2.5: Rezultati na srednje gustim DIMACS instancama

k	instanca	VNS_{best}	VNS_{avg}	VNS_{gap}	VNS_t^{tot}	VNS_t
1	brock200_2	83957	81541	2.88	3382.44	2124.66
2	brock200_2	99827	96704	3.13	3301.18	1757.93
3	brock200_2	113638	111516.5	1.87	3421.98	2197.47
1	brock200_3	106951	103959.2	2.8	3471.57	2309.01
2	brock200_3	127372	123697.3	2.89	3112.79	1704.6
3	brock200_3	151191	144467.1	4.45	3140.09	1674.75
1	brock200_4	125041	120718.3	3.46	3442.31	1706.72
2	brock200_4	149318	145804.2	2.35	3446.03	2009.04
3	brock200_4	172932	168376.5	2.63	3236.22	1688.54
1	brock800_1	610931	601163.6	1.6	3605.06	3135.25
2	brock800_1	733054	719803.8	1.81	3602.69	3237.25
3	brock800_1	840573	826170.7	1.71	3603.17	2990.6
1	brock800_2	623218	609875.5	2.14	3603.67	3175.32
2	brock800_2	733488	727198.8	0.86	3605.32	3224.69
3	brock800_2	855795	837945	2.09	3603.38	3115.73
1	brock800_3	614604	604339.7	1.67	3606.34	3038.42
2	brock800_3	735000	719006.2	2.18	3602.17	3254.49
3	brock800_3	862070	834743.8	3.17	3603.78	3104.96
1	brock800_4	616989	605585.5	1.85	3604.8	3410.98
2	brock800_4	726104	718721.3	1.02	3604.8	3231.53
3	brock800_4	855557	834975.1	2.41	3605.51	3235.18
1	C20005	1192290	1186846.1	0.46	3659.85	2744.78
2	C20005	1392299	1376962	1.1	3636.29	2057.07
3	C20005	1594213	1570267.9	1.5	3645.71	2248.83
1	C40005	2533903	2518046.8	0.63	4627.83	4562.85
2	C40005	2920948	2861905.8	2.02	4578.67	4534.81
3	C40005	3234853	3207006.3	0.86	4472.43	4423.51
1	c-fat200-5	526632	526632	0	2730.76	1.41
2	c-fat200-5	526632	526632	0	1762.96	16.44
3	c-fat200-5	526632	526632	0	1505.4	43.02
1	c-fat500-1	292180	290327.2	0.63	1039.89	128.87
2	c-fat500-1	292180	290713	0.5	1167.18	376.53
3	c-fat500-1	292180	290818.6	0.47	1250.86	480.01
1	c-fat500-10	3132604	3129593.8	0.1	3604.69	848.13
2	c-fat500-10	3132604	3126583.6	0.19	3604.65	283.95
3	c-fat500-10	3132604	3117553	0.48	3604.5	9.71
1	c-fat500-2	607420	602305	0.84	1805.71	29.44
2	c-fat500-2	607420	601883.2	0.91	1707.67	595.17
3	c-fat500-2	607420	593652.6	2.27	1918.26	976.62
1	c-fat500-5	1553956	1550935.6	0.19	3601.4	18.12
2	c-fat500-5	1553956	1550935.6	0.19	3596.94	35.66
3	c-fat500-5	1553956	1549425.4	0.29	2585.78	183.7
1	DSJC10005	546588	537181.1	1.72	3609.75	2661.45
2	DSJC10005	651009	636940.1	2.16	3605.05	3372.33
3	DSJC10005	733716	724846.6	1.21	3605.99	3332.16
1	DSJC5005	250614	245224.9	2.15	3601.27	3301.39
2	DSJC5005	297913	289818.5	2.72	3600.85	2910.57
3	DSJC5005	335359	329672.1	1.7	3601	2985.17
1	hamming8-4	192960	191400	0.81	3600.28	1963.09
2	hamming8-4	177892	168266.8	5.41	3600.15	2862.13
3	hamming8-4	203530	200470	1.5	3600.18	2887.56
1	keller4	76504	73273.8	4.22	2985.62	2284.24
2	keller4	103807	101761.7	1.97	2844.21	2059.06
3	keller4	127741	124771.7	2.32	2780.2	1937.81
1	p_hat1000-1	306225	301257.2	1.62	3603.76	3230.62
2	p_hat1000-1	344416	340452.9	1.15	3603.69	3243.66
3	p_hat1000-1	386953	381182.2	1.49	3603.49	3399.92
1	p_hat1000-2	620804	609955.2	1.75	3608.93	3156.85
2	p_hat1000-2	752708	741969.9	1.43	3610.79	3227.01
3	p_hat1000-2	876834	850376.4	3.02	3607.58	3284.72
1	p_hat1500-1	483776	480441.4	0.69	3613.84	2923.22
2	p_hat1500-1	553419	546879.6	1.18	3615.63	2879.43
3	p_hat1500-1	606570	601482.5	0.84	3614.14	2726.55
1	p_hat1500-2	1072696	1050749.8	2.05	3630	3164.49
2	p_hat1500-2	1287355	1264266.4	1.79	3650.16	3163.63
3	p_hat1500-2	1508515	1473066	2.35	3635.82	3229.25
1	p_hat300-1	75543	74191.1	1.79	3600.25	2344.72
2	p_hat300-1	88060	85403.7	3.02	3600.16	2998.49
3	p_hat300-1	98101	95630.6	2.52	3600.23	2926.7
1	p_hat300-2	146038	142308.6	2.55	3600.49	3057.49
2	p_hat300-2	174274	169489.3	2.75	3600.36	3223.7
3	p_hat300-2	200877	198456.1	1.21	3600.32	2960.37
1	p_hat500-1	139576	138355.9	0.87	3601.11	3183.64
2	p_hat500-1	163899	160305	2.19	3601.09	3233.04
3	p_hat500-1	180588	178385.9	1.22	3600.73	3463.87
1	p_hat500-2	296545	284583.2	4.03	3602.18	3455.99
2	p_hat500-2	351634	341516	2.88	3600.97	3426.38
3	p_hat500-2	401640	390645.9	2.74	3601.73	3223.11
1	p_hat700-1	206683	202079.5	2.23	3602.38	3392.02
2	p_hat700-1	238036	232822.5	2.19	3601.66	3422.51
3	p_hat700-1	259926	255331.7	1.77	3601.96	3348.71
1	p_hat700-2	435856	427743.3	1.86	3602.79	3348.54
2	p_hat700-2	521716	507486.9	2.73	3602.96	3255.04
3	p_hat700-2	604632	585263.2	3.2	3603.13	3290.68
1	san1000	472999	470872.9	0.45	3605.32	2946.36
2	san1000	909108	903832.8	0.58	3610.69	2728.4
3	san1000	1335594	1319144.4	1.23	3609.91	2973.47
1	san400_051	161298	160595.7	0.44	3600.71	3197.51
2	san400_051	304963	301161.1	1.25	3600.83	3234.66
3	san400_051	442973	438108.5	1.1	3600.78	3377.38
1	sanr400_05	190272	185531.2	2.49	3600.95	3060.33
2	sanr400_05	231407	222275.5	3.95	3600.84	3179.97
3	sanr400_05	260065	251051.8	3.47	3600.87	3070.45

Tabela 2.6: Rezultati na jako gustim DIMACS instancama (prvi dio)

k	instanca	VNS_{best}	VNS_{avg}	VNS_{gap}	VNS_t^{tot}	VNS_t
1	brock200_1	158707	154360.1	2.74	3500.13	2461.32
2	brock200_1	192439	187551.5	2.54	3545.54	2259.19
3	brock200_1	225269	218698.8	2.92	3539.2	2463.07
1	brock400_1	371933	359355.3	3.38	3601.17	3329.02
2	brock400_1	449671	435976.5	3.05	3600.79	3268.41
3	brock400_1	526530	507914.5	3.54	3600.76	3243.23
1	brock400_2	371578	361913.2	2.6	3600.63	3283.99
2	brock400_2	444406	432533	2.67	3600.81	3048.42
3	brock400_2	534939	512612.1	4.17	3600.48	3137.32
1	brock400_3	372822	363484.8	2.5	3600.88	3288.07
2	brock400_3	446998	437843.8	2.05	3600.69	3269.78
3	brock400_3	524128	512764.4	2.17	3600.57	3218.06
1	brock400_4	369639	363482.6	1.67	3600.39	3202.59
2	brock400_4	448574	440795.2	1.73	3600.61	3163.76
3	brock400_4	526372	511945.1	2.74	3600.67	3183.16
1	C10009	2355000	2276048.7	3.35	3606.39	3434.45
2	C10009	2852942	2814950.4	1.33	3609.12	3062.12
3	C10009	3345922	3301912.3	1.32	3607.95	2836.76
1	C1259	170385	166887.2	2.05	1089.76	593.72
2	C1259	215756	207682	3.74	919.72	486.96
3	C1259	252917	245691.3	2.86	663.14	229.86
1	C20009	5218768	5100344.8	2.27	3654.51	3124.75
2	C20009	6337227	6282601	0.86	3661.89	3560.99
3	C20009	7506419	7419466.1	1.16	3649.13	3123.3
1	C2509	403881	387649.8	4.02	3600.25	2569.97
2	C2509	488684	474146.4	2.98	3600.21	2455.68
3	C2509	581653	565259.2	2.82	3585.56	2436.05
1	C5009	1000447	974512.1	2.59	3601.25	3338.22
2	C5009	1230713	1190286.8	3.29	3600.96	3360.28
3	C5009	1447877	1408745.6	2.7	3601.09	3243.52
1	gen200_p09_55b	368479	361764.3	1.82	3405.15	1616.04
2	gen200_p09_55b	422470	409993.5	2.95	2876.7	1516.52
3	gen200_p09_55b	470495	448115.8	4.76	2794.22	1577.55
1	gen200_p0944b	313267	297793.3	4.94	3417.46	2039.88
2	gen200_p0944b	376447	366086	2.75	3260.92	1776.18
3	gen200_p0944b	468420	446571	4.66	2638.06	1502.81
1	gen400_p09_55b	743725	706633.4	4.99	3600.58	3327.88
2	gen400_p09_55b	945178	915300.8	3.16	3600.97	3404.11
3	gen400_p09_55b	1206175	1162360.7	3.63	3600.75	3436.65
1	gen400_p09_65b	816366	749095.3	8.24	3601.01	3267.38
2	gen400_p09_65b	1120767	1097093	2.11	3600.84	3330.96
3	gen400_p09_65b	1404041	1353213.9	3.62	3600.77	3335.37
1	gen400_p09_75b	908894	816825.2	10.13	3600.78	3365.44
2	gen400_p09_75b	1222436	1142968.7	6.5	3600.71	3257.6
3	gen400_p09_75b	1565327	1530803.7	2.21	3601.01	3144.91
1	hamming10-2	26281632	25538799.6	2.83	3622.42	800.49
2	hamming10-2	26281632	26258736	0.09	3625.05	626.21
3	hamming10-2	26235288	24883654	5.15	3610.54	2353.61
1	hamming10-4	1545048	1521115.8	1.55	3608.21	3177.76
2	hamming10-4	2005128	1981680.6	1.17	3606.59	3146.84
3	hamming10-4	2503246	2481878.9	0.85	3605.87	3232.16
1	hamming8-2	1601248	1601248	0	3600.92	7.89
2	hamming8-2	1601248	1554210	2.94	3147.23	59.63
3	hamming8-2	1591376	1586188.8	0.33	2651.78	141.73
1	johnson16-2-4	48840	48620	0.45	882.42	362.71
2	johnson16-2-4	58306	57888.9	0.72	979.4	850.65
3	johnson16-2-4	89160	88355.2	0.9	1054.68	763.41
1	johnson32-2-4	392760	388792.5	1.01	3601.04	3088.37
2	johnson32-2-4	532737	527346.3	1.01	3601.46	2939.88
3	johnson32-2-4	750614	746366.6	0.57	3601.19	3130.56

Tabela 2.7: Rezultati na jako gustim DIMACS instancama (drugi dio)

k	instanca	VNS_{best}	VNS_{avg}	VNS_{gap}	VNS_t^{tot}	VNS_t
1	keller5	682407	665354.3	2.5	3606.32	3119.63
2	keller5	945801	925477.8	2.15	3605.19	3109.03
3	keller5	1165321	1134296.7	2.66	3606.09	3113.98
1	keller6	5525571	5445673.4	1.45	3979.66	3848.26
2	keller6	7323268	7102166.1	3.02	4461.92	4398.88
3	keller6	8508595	8307213	2.37	4482.79	4431.42
1	MANN _a27	2343507	2330168.9	0.57	3600.48	3383.01
2	MANN _a27	3646589	3621435.2	0.69	3600.32	3329.47
3	MANN _a27	6125697	6125697	0	3321.03	8.1
1	MANN _a45	17150512	16983596.8	0.97	3603.09	3427.66
2	MANN _a45	27756773	27710448.5	0.17	3602.71	3444.64
3	MANN _a45	49146260	49146260	0	3603.95	158.28
1	MANN _a81	176457108	172772995.9	2.09	4345.39	4340.21
2	MANN _a81	289278389	287439874.6	0.64	4537.53	4534.35
3	MANN _a81	503419685	456648382	9.29	4564.84	4559.97
1	p _hat1000-3	1137442	1119540.7	1.57	3616.9	2561.13
2	p _hat1000-3	1450974	1408614.4	2.92	3612.38	2993.56
3	p _hat1000-3	1665322	1629319	2.16	3606.07	3548.09
1	p _hat1500-3	1963300	1928017.7	1.8	3670.69	3229.46
2	p _hat1500-3	2455440	2383550.4	2.93	3631.5	3053.33
3	p _hat1500-3	2836266	2778520.4	2.04	3649.51	3320.68
1	p _hat300-3	272754	262647.4	3.71	3600.35	2919
2	p _hat300-3	324530	318655.6	1.81	3600.51	2849.22
3	p _hat300-3	384709	372360	3.21	3600.44	2899.94
1	p _hat500-3	540234	512726.5	5.09	3602.06	3307.48
2	p _hat500-3	640868	630022.4	1.69	3601.84	3271.9
3	p _hat500-3	760971	742552.7	2.42	3601.71	3261.19
1	p _hat700-3	793145	772658.7	2.58	3606.52	3157.43
2	p _hat700-3	985487	954379.3	3.16	3602.88	3514.07
3	p _hat700-3	1148670	1119855.4	2.51	3603.43	3357.1
1	san200 _071	198080	190287.5	3.93	3600.13	3459.14
2	san200 _071	340771	337517.6	0.96	3600.27	3363.46
3	san200 _071	483227	481167	0.43	3600.23	3352.08
1	san200 _072	133475	131023.2	1.84	3600.1	3385.83
2	san200 _072	237161	233121.3	1.7	3600.14	3322.87
3	san200 _072	343431	341721.5	0.5	3600.24	2984.12
1	san200 _091	492107	492107	0	3560.62	930.2
2	san200 _091	666711	666108.4	0.09	3600.25	2688.28
3	san200 _091	1043442	1043442	0	3600.67	1.71
1	san200 _092	409580	401815.2	1.9	3558.99	2355.09
2	san200 _092	598830	598094.9	0.12	3600.18	2661.41
3	san200 _092	855085	855085	0	3513.21	631.71
1	san200 _093	316073	297292.8	5.94	3553.91	2595.78
2	san200 _093	478051	473735.6	0.9	3600.15	3174.13
3	san200 _093	610613	609708.7	0.15	3572.73	2310.79
1	san400 _071	494400	488846	1.12	3600.87	3317.96
2	san400 _071	959400	949800	1	3601.49	3246.48
3	san400 _071	1296354	1282667.8	1.06	3601.6	3325.4
1	san400 _072	375295	373622.2	0.45	3600.59	3337.29
2	san400 _072	715213	707452.3	1.09	3601.22	3345.91
3	san400 _072	1016218	1005445.1	1.06	3600.81	3316.08
1	san400 _073	291445	286186.9	1.8	3600.89	3392.7
2	san400 _073	532610	528741.2	0.73	3601.11	3348
3	san400 _073	787442	780179.6	0.92	3601.29	3275.22
1	san400 _091	1148400	1138517.9	0.86	3601.46	3434.89
2	san400 _091	2202600	2194980	0.35	3602.21	3112.71
3	san400 _091	2569764	2546100.2	0.92	3601.72	3293.57
1	sanr200 _07	133983	130200.8	2.82	3600.14	2597.59
2	sanr200 _07	165225	159593.6	3.41	3600.19	2700.83
3	sanr200 _07	187520	183874.4	1.94	3474.57	2468.73
1	sanr200 _09	307316	296899.7	3.39	3583.18	1608.38
2	sanr200 _09	380281	362800.5	4.6	3475.83	2465.84
3	sanr200 _09	445961	433429.7	2.81	3535.85	2300.92
1	sanr400 _07	315057	307111.7	2.52	3601.04	3432.48
2	sanr400 _07	380604	373912.6	1.76	3601.09	2984.13
3	sanr400 _07	446230	430625.8	3.5	3601.08	3323.89

Glava 3

Predviđanje uloge metabolita u metaboličkim rekacijama

3.1 Uvod

Predviđanje karakteristika podataka je jedan od najvažnijih problema u informacionim naukama. Brojni su primjeri primjene predviđanja u različitim oblastima, na primjer prediviđanje da li je primljena elektronska pošta poželjna ili nepoželjna , predviđanje tumorskih ćelija kao benignih ili malignih, predviđanje neuređenosti proteina i sl. Pod problemom predviđanja se mogu smatrati i različiti problemi klasifikacije poput klasifikacije teksta, klasifikacije validnosti kreditih kartica i sl.

Cilj istraživanja predstavljenog u ovom poglavlju je da se izvrši predviđanje uloge metabolita u metaboličkim reakcijama. Zatim se na osnovu rezultata predviđanja svaki metabolit može klasifikovati u određenu klasu, koja odgovara njegovoj ulozi u reakciji. Predviđanje uloge metabolita je važno za dalje razumijevanje metaboličkih puteva. Svaki metabolički put se može posmatrati kao serija hemijskih reakcija koja uključuju reaktante, proizvode i različite intermedijere. Postoje dva osnovna tipa biohemijskih reakcija, one koje učestvuju u anabolizmu i one koje učestvuju u katabolizmu. U anaboličkim putevima se sintetišu molekuli uz potrošnju energije, dok se pri kataboličkim putevima degradiraju molekuli uz oslobađanje energije u vidu ATP-a. Adenozin trifosfat (engl. Adenosine triphosphate - ATP) se pojavljuje u oba tipa metaboličkih puteva, u anaboličkim kao reaktant, a u kataboličkim kao proizvod reakcije. U anaboličkim putevima se ATP degradira do adenozin difosfata (engl. Adenosine diphosphate - ADP) ili adenozin monofosfata (engl. Adenosine monophosphate - AMP), oslobađajući pri tome jednu, odnosno dvije fosfatne grupe. Oslobođene fosfatne grupe ili ostaju slobodne ili se dalje vežu za neki od metabolita u procesu fosforilacije. U kataboličkim putevima se dešava obrnut proces, odnosno proces sinteze ATP iz nižih formi ADP ili AMP. Identifikacija uloge metabolita bi mogla biti korisna za dobijanje novih informacija o vezi između metabolita na osnovu njihovog učešća u istim reakcijama. Takođe, dobijeni rezultati se mogu koristiti za dalju analizu metabolizama određenih organizama.

U ovom poglavlju za predviđanje uloge metabolita u metaboličkim reakcijama koristi se metoda uslovnih slučajnih polja (engl. Conditional Random Fields - CRF), a dio rezultata je publikovan u radu [52]. Uslovna slučajna polja su metoda predviđanja koja uključuje zavisnost između varijabli u proces predviđanja. U istraživanju koje je predstavljeno u ovom poglavlju, linearna uslovna slučjana polja su prilagođena i primijenjena na listu hemijskih reakcija. Lista hemijskih reakcija se posmatra kao niz rečenica, gdje se svaka reakcija posmatra kao jedna rečenica. Izabrani skup rekacija pripada

određenom putu sa svojstvom da ili sintetišu metabolite uz upotrebu energije ili razlažu metabolite uz oslobađanje energije. S obzirom da uloga metabolita zavisi od metabolita koji su mu susjedi kao i od njihovih uloga, metoda bazirana na uslovnim slučajnim poljima se čini pogodnom za ovo predviđanje, jer uzima u obzir kontekst, tj. okruženje elementa za koji se vrši predviđanje. Kombinacija informacija o susjednim elementima i oznaka njihovih uloga u reakciji kao ulaz u model zasnovan na uslovnim slučajnim poljima bi mogla dati precizno predviđanje uloge metabolita u reakciji. Koliko je poznato, u literaturi problem predviđanja uloge metabolita još uvijek nije razmatran na ovaj način. Međutim, u literaturi se može naći nekoliko primjera primjene metoda zasnovanih na uslovnim slučajnim poljima na slična predviđanja bioloških elemenata.

3.2 Pregled rezultata nad srodnim problemima

Uslovna slučajna polja su često korištena metoda za rješavanje problema obrade teksta, kao što je problem određivanja vrste riječi ili problem identifikacije imenovanih entiteta. Problem identifikacije imenovanih entiteta podrazumijeva pronalaženje riječi ili fraza koje pripadaju određenoj klasi (npr. datuma, ličnih imena, naziva institucija i sl.). Za rješavanje ovog problema u [99] predstavljena je tehnika *WebListing*, bazirana na metodi uslovnih slučajnih polja. Ova tehnika formira sjemena za rječnike (engl. seeds for lexicons) koji su zasnovani na označenim podacima, proširujući ih dodatnim informacijama sa interneta. Različiti pristupi zasnovani na metodi uslovnih slučajnih polja koji se koriste za prepoznavanje specifičnih bioloških termina poput PROTEIN, DNA, RNA, CELL-LINE i CELL-TYPE u apstraktima biomedicinskih tekstova su prikazani u [131]. Sistem za identifikaciju imenovanih entiteta u biomedicinskim tekstovima *BANNER*, zasnovan je na mašinskom učenju i metodi uslovnih slučajnih polja, te dizajniran tako da maksimizuje nezavisnost domena, dostižući tako bolje performanse nego ostali sistemi [83]. Hibridni model LSTM-CRF, koji je kombinacija Long Short-term Memory Networks (LSTM) i CRF, se također koristi za rješavanje problema identifikacije imenovanih entiteta [81].

Brojne su primjene metode uslovnih slučajnih polja za analizu i procesiranje teksta. Na primjer, u [132] ova metoda se koristi za "plitko" (engl. shallow) parsiranje teksta, odnosno za analizu rečenica u kojoj se prvo prepoznaju imenice, glagoli, pridjevi itd., koji se grupišu u termine višeg reda poput fraza. Prepoznavanje vrste riječi (engl. Part Of Speech (POS) i Chunking) upotrebom metoda koje se zasnivaju na uslovnim slučajnim poljima predstavljeno je i u [80]. Za dodjele semantičkih oznaka koristi se stablo uslovnih slučajnih polja [32]. U [120] uslovna slučajna polja su upotrebljena za izdvajanje informacija iz tabela. Metoda uslovnih slučajnih polja se također može koristiti i za poravnanje riječi [16], sažimanje dokumenata [137] i interaktivno odgovaranje na pitanja [62].

Uslovna slučajna polja su osnova za neke metode segmentacije slika. Diskriminativna uslovna slučajna polja (engl. Discriminative Random Fields (DRF)), u čijoj su osnovi uslovna slučajna polja, imaju primjenu u modeliranju prostornih zavisnosti [79]. Oblici i teksture se modeluju u složenije oblike - tekstone (engl. textons), koji su novi atributi uključeni u model uslovnih slučajnih polja za segmentaciju slika [138]. Segmentacija dijelova slike koji su u "prvom planu" ili sjenki može biti urađena pomoću posebne klase uslovnih slučajnih polja, koja su uvedena u [162] i nazvana dinamička uslovna slučajna polja (engl. Dynamic conditional random fields (DCRF)).

U [129] uslovna slučajna polja se koriste kao zamjena za heuristički pristup algoritmima za prostornu analizu slika (engl. stereo vision). Formiran je veliki broj prostornih skupova podataka sa utvrđenim, polaznim raznolikostima i podskupovi ovih skupova se koriste za učenje parametara

uslovnih slučajnih polja.

Metoda uslovnih slučajnih polja se intenzivno koristi u različitim oblastima bioinformatike. U [101] se koristi za označavanje gena i proteina koji se pominju u tekstu. Kao što je već navedeno, metoda uslovnih slučajnih polja je pogodna za prepoznavanje imenovanih entiteta u biomedicinskim tekstovima [131]. Prvi komparativni prediktor gena zasnovan na polu-Markovljevim uslovnim slučajnim poljima (engl. semi-Markov conditional random fields (SMCRFs)) pod imenom *Conard* predstavljen je u [39]. Jedan od najvažnijih problema u bioinformatici je problem predviđanja uvijanja proteina. Efikasno rješenje ovog problema bazirano na segmentacionim uslovnim slučajnim poljima (engl. segmentation conditional random fields (SCRFs)) je predstavljeno u [90]. Procjena parametara koji se koriste za RNK strukturna poravnanja i pretraga strukturnih poravnanja bazirane na metodi uslovnih slučajnih polja su bolja i tačnija nego druge metode [128]. Markovljeva slučajna polja, u kojima je zajednička raspodjela varijabli Gausova, nazivaju se Gausova uslovna slučjana polja (engl. Gaussian CRF (GCRFs)) i imaju primjenu u računarskom razumijevanju slika i videa (engl. computer vision) [148]. Usmjerena Gausova slučajna polja (engl. Directed Gaussian conditional random fields (DirGCRF)) su proširenje Gausovih uslovnih slučajnih polja uvedena radi modelovanja asimetričnih odnosa (npr. prijateljstvo, uticaj, ljubav, solidarnost i sl.) [159].

Predviđanje uloge bioloških elemenata u različitim procesima analizirano je u nekoliko radova. U [154] autori razmatraju ulogu metabolita u predviđenim interakcijama između lijekova (engl. drug-drug interactions). Fokus navedenog istraživanja je na inhibiciji citohroma P450 enzima i na metabolitima koji su fundamentalni za tu inhibiciju. Bajesov pristup i označeni susjedi atoma sa više nivoa (engl. Labelled Multilevel Neighborhoods of Atoms (LMNA) descriptors) se koriste za predviđanje reaktivnih atoma u molekulima [126]. Metoda za predviđanje kojem metaboličkom putu pripada određena komponenta je opisana u [64] i zasnovana je na najvišim intenzitetima interakcija. Pristup baziran na slučajnim šumama (engl. The Random Forest) za predviđanje metaboličkih enzima i crijevnih bakterija je predstavljen u [136].

3.3 Metoda uslovnih slučajnih polja za predviđanje uloge metabolita

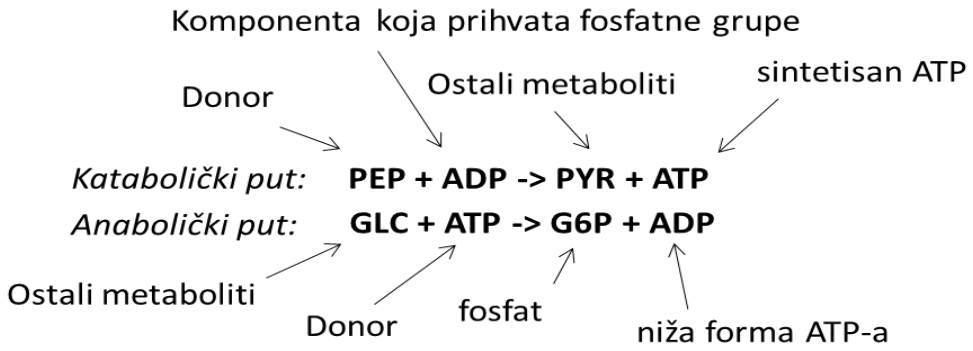
3.3.1 Definicja problema

Neka je data lista metaboličkih reakcija. Svaka reakcija se sastoji od nekoliko metabolita, koji se pojavljuju sa lijeve ili desne strane strelice (vidjeti primjer na Slici 3.1). Metaboliti sa lijeve strane strelice se nazivaju reaktantima reakcije, a metaboliti sa desne strane strelice proizvodima reakcije. Problem klasifikacije se definiše na sljedeći način: za datu reakciju pridružiti oznaku svakom metabolitu u reakciji, tako da oznaka predstavlja ulogu metabolita u reakciji. Skup oznaka je dat unaprijed i u opštem slučaju se zasniva na specifičnoj potrebi u datom kontekstu.

Jedan od mogućih načina dodjeljivanja oznaka se zasniva na učešću metabolita u prenosu energije ili u procesu fosforilacije. U ovom istraživanju je identifikovano sljedećih osam klasa metabolita:

- Label 1 - donor jedne ili dvije fosfatne grupe;
- Label 2 - komponenta koja prihvata jednu ili dvije fosfatne grupe;
- Label 3 - slobodna fosfatna grupa/grupe;

- Label 4 - fosfat (komponenta formirana vezivanjem jedne ili dvije fosfatne grupe za metabolit);
- Label 5 - niža forma ATP-a u anaboličkom putu;
- Label 6 - ATP sintetisan u kataboličkom putu;
- Label 7 - fosfatna grupa koja se veže u kataboličkom putu;
- Label 8 - ostali metaboliti, koji ne pripadaju nijednoj od navedenih klasa.



Slika 3.1: Primjer označavanja metabolita u reakciji

Reakcija se posmatra kao niz od nekoliko elemenata koji se nalaze sa lijeve ili desne strane strelice. Zato je važno prepoznati strelicu kao znak koji razdvaja te dvije strane reakcije, pa je iz navednih razloga uvedena dodatna oznaka Label 9: "strelica". U slučaju da se posmatra neka druga lista reakcija, lista klasa može biti proširena, skraćena ili promijenjena.

Primjer 3.1. *Na Slici 3.1 su prikazane dvije reakcije. Prva reakcija je dio kataboličkog puta. PEP ili fosfoenolpiruvat je važno jedinjenje, koje sadrži energetski bogatu fosfatnu vezu [13]. U datoj reakciji PEP je donor jedne fosfatne grupe. Drugi reaktant je adenozin difosfat (ADP) koji se sastoji od tri komponente: adenina, šećera i dvije fosfate grupe [111]. ADP je u datoj reakciji označen sa Label 2 jer prihvata jednu fosfatnu grupu koju je otpustio PEP. Tako ADP veže otpuštenu fosfatnu grupu i formira ATP (adenozin trifosfat). Zbog navedenog ATP u ovoj reakciji ima oznaku klase Label 6 - ATP sintetisan u kataboličkom putu. Druga reakcija sa Slike 3.1 je dio anaboličkog puta, pa je ATP donor jedne fosfatne grupe, odnosno, ima oznaku Label 1, dok je ADP niža forma ATP-a u anaboličkom putu (Label 5). G6P je fosfat (komponenta formirana vezivanjem jedne ili dvije fosfatne grupe za metabolit), a GLC ima oznaku klase Label 8 - ostali metaboliti, koji ne pripadaju nijednoj od navedenih klasa.*

3.3.2 Metoda uslovnih slučajnih polja za klasifikaciju metabolita

Uslovna slučajna polja su diskriminativni vjerovatnosni model mašinskog učenja koji se koristi za strukturalno predviđanje. Strukturalno predviđanje je nadgledana tehnika mašinskog učenja kojom se vrši predviđanje struktuiranih objekata poput sekvenci, grafova, drveta i sl. Uslovna slučajna polja su uvedena u [80]. Osnovni princip se može objasniti na problemu označavanja sekvencijalnih podataka. Neka je T dužina sekvence i neka $\mathbf{w} = \{\mathbf{w_1}, \mathbf{w_2}, ..., \mathbf{w_T}\}$ predstavlja karakteristike elemenata sekvence, gdje je svaki $\mathbf{w_i}$ vektor karakteristika elementa x_i, odnosno elementa koji se nalazi na

poziciji i. Zadatak je da se, za svaki element x_i, na osnovu datog vektora karakteristika $\mathbf{w_i}$ i susjednih oznaka, pronađe odgovarajuća oznaka y_i. Za rješavanje ovog problema dat je skup trening podataka $(\mathbf{w_i}, y_i)$ sa tačnim informacijama o oznakama. Problem predviđanja je zapravo problem pronalaženja vjerovatnoće $p(\mathbf{y}|\mathbf{w})$, gdje je $\mathbf{y} = \{y_1, y_2, ..., y_T\}$. Pomenuta vjerovatnoća se može odrediti na dva načina:

- na osnovu zajedničke vjerovatnoće $p(\mathbf{y}, \mathbf{w})$ i vjerovatnoće $p(\mathbf{w})$ - generativni pristup,
- direktno, modelovanjem uslovne vjerovatnoće $p(\mathbf{y}|\mathbf{w})$ - diskriminativni pristup.

Kao što je već pomenuto, uslovna slučajna polja pripadaju klasi diskriminativnih modela, pa se vrši direktno izračunavanje uslovne vjerovatnoće $p(\mathbf{y}|\mathbf{w})$. Generativni analog uslovnim slučajnim poljima su skriveni Markovljevi modeli.

Za izračunavanje uslovne vjerovatnoće u metodi uslovnih slučajnih polja obično se koristi formula

$$p(\mathbf{y}|\mathbf{w}) = \frac{1}{Z(\mathbf{w})} \exp\left\{ \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, \mathbf{w_t}) \right\}$$

gdje je f_k karakteristična funkcija, a θ_k su komponente vektora parametara za $1 \leq k \leq K$. Karakteristična funkcija je šablon koji opisuje situaciju da je vektor karakteristika na poziciji t - $\mathbf{w_t}$, a da su oznake na pozicijama $t-1$ i t baš y_{t-1} i y_t. Ona je zapravo indikatorska funkcija koja će imati vrijednost 1 ukoliko podaci zadovoljavaju taj šablon, a 0 inače. Faktor normalizacije se definiše na sljedeći način

$$Z(\mathbf{w}) = \sum_{y \in Y^T} \exp\left\{ \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, \mathbf{w_t}) \right\}.$$

Može se primijetiti da je vektor $\mathbf{w_t}$ argument karakteristične funkcije $f_k(y_{t-1}, y_t, \mathbf{w_t})$ što ukazuje na to da su dostupne sve komponente globalnog posmatranja $\mathbf{w}$, koje su potrebne za određivanje karakteristika za element sa pozicije t. Na primjer, ako se sljedeći element x_{t+1} koristi kao karakteristika u metodi uslovnih slučajnih polja, pretpostavlja se da je informacija o identitetu tog elementa uključena u vektor $\mathbf{w_t}$ [147].

Hemijske reakcije se mogu posmatrati kao rečenice. Iako ne postoje striktna pravila za zapisivanje hemijskih reakcija, ipak postoje neke opšte konvencije. Na primjer, donor jedne ili dvije fosfatne grupe se obično nalazi na prvoj poziciji ili na poziciji prije metabolita koji će prihvatiti otpuštenu fosfatnu grupu ili otpuštene fosfatne grupe, a niže forme ATP-a se obično nalaze na pretposljednjoj ili posljednjoj poziciji. Dakle, za određivanje uloge pojedinačnog metabolita je opravdano posmatrati njemu susjedne elemente u reakciji i njihove oznake. Da bi se testirala ova hipoteza formirana su i razmatrana tri različita modela (nazvana A, B i C), koja koriste različite karakteristične funkcije.

Model A

U prvom modelu karakteristične funkcije su bazirane na informacijama o susjednim metabolitima u reakciji i informaciji o oznaci posmatranog elementa. Preciznije, za određivanje oznake metabolita koji se u reakciji nalazi na poziciji t, u obzir se uzima informacija o najbližim metabolitima kao i informacija o najbližim uzastopnim parovima metabolita (sa lijeve i desne strane). Formalnije, razmatraju se informacije o metabolitima sa pozicija iz skupa $\{t-2, t-1, t, t+1, t+2\}$ i informacija

o oznaci posmatranog elementa. Koristi se sljedeći skup karakterističnih funkcija:

$$f^{-2}_{x,y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_{t-2} = x, y_t = y),$$

$$f^{-1}_{x,y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_{t-1} = x, y_t = y),$$

$$f^{0}_{x,y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_t = x, y_t = y),$$

$$f^{+1}_{x,y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_{t+1} = x, y_t = y),$$

$$f^{+2}_{x,y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_{t+2} = x, y_t = y),$$

$$f^{-1,0}_{x,x',y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_{t-1} = x, x_t = x', y_t = y),$$

$$f^{0,+1}_{x,x',y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_t = x, x_{t+1} = x', y_t = y).$$

U svim navedenim formulama I je indikatorska funkcija, odnosno funkcija čija je vrijednost 1 ako je uslov ispunjen, u suprotnom 0. Gornji indeksi u oznaci funkcije f, odnosno oznake $-i$ (respektivno $+i$) za $i \in \{1, 2\}$, ukazuju na to da se u obzir uzimaju informacije o metabolitima koji su za i pozicija ispred (odnosno iza) od posmatranog metabolita. Nula u gornjem indeksu znači da se u obzir uzimaju informacije o posmatranom elementu. Sve navedene funkcije čine skup funkcija F_A koje se koriste u modelu A, odnosno

$$F_A = \{f^{-2}_{z,y}, f^{-1}_{z,y}, f^{0}_{z,y}, f^{+1}_{x,y}, f^{+2}_{x,y}, f^{-1,0}_{x,x',y}, f^{0,+1}_{x,x',y} | x, x' \in X, y \in Y\}$$

gdje je X skup svih metabolita koji se nalaze u datim reakcijama, a Y skup svih oznaka.

Model B

Prvih pet funkcija iz modela A se također koristi i u modelu B. Taj skup karakterističnih funkcija je proširen funkcijama koje sadrže informacije o oznaci metabolita koji prethodi posmatranom metabolitu u reakciji, kao i informacije o najbližim metabolitima, odnosno sljedećim funkcijama

$$g^{-1,0}_{x',x,y',y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_{t-1} = x', x_t = x, y_{t-1} = y', y_t = y),$$

$$g^{0,+1}_{x',x,y',y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_t = x', x_{t+1} = x, y_{t-1} = y', y_t = y),$$

$$g^{-2,-1,0}_{x'',x',x,y',y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_{t-2} = x'', x_{t-1} = x', x_t = x, y_{t-1} = y', y_t = y),$$

$$g^{-1,0,+1}_{x'',x',x,y',y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_{t-1} = x'', x_t = x', x_{t+1} = x, y_{t-1} = y', y_t = y),$$

$$g^{0,+1,+2}_{x'',x',x,y',y}(y_{t-1}, y_t, \mathbf{w_t}) = I(x_t = x'', x_{t+1} = x', x_{t+2} = x, y_{t-1} = y', y_t = y).$$

Skup karakterističnih funkcija koje se koriste u modelu B je

$$F_B = \{f^{-2}_{x,y}, f^{-1}_{x,y}, f^{0}_{x,y} f^{+1}_{x,y}, f^{+2}_{x,y}, g^{-1,0}_{x',x,y',y}, g^{0,+1}_{x',x,y',y}, g^{-2,-1,0}_{x'',x',x,y',y}, g^{-1,0,+1}_{x'',x',x,y',y}$$
$$, g^{0,+1,+2}_{x'',x',x,y',y} | x, x', x'' \in X, y, y' \in Y\}$$

Funkcija $g^{-2,-1,0}_{x'',x',x,y}$ uzima u obzir informacije o metabolitima na pozicijama $t-2, t-1$ i t. Slično, u funkciji $g^{0,+1,+2}_{x'',x',x,y}$ se posmatraju elementi na pozicijama $t, t+1$ i $t+2$, dok se za izračunavanje vrijednosti funkcije $g^{-1,0,+1}_{x'',x',x,y}$ koriste informacije o metabolitima na pozicijama $t-1, t$ i $t+1$. Informacija o oznakama klase prethodnog i posmatranog elementa se koriste u svim navedenim funkcijama.

Model C

Skup funkcija koje se koriste u modelu C sadrži samo one funkcije koje u obzir uzimaju informaciju o oznaci klase prethodnog elementa i može se dobiti iz skupa F_B izostavljanjem nekih funkcija iz modela A, odnosno

$$F_C = F_B \setminus \{f_{x,y}^{-2}, f_{x,y}^{-1}, f_{x,y}^{0}, f_{x,y}^{+1}, f_{x,y}^{+2} | x \in X, y \in Y\}.$$

Sljedećim primjerom će biti ilustrovano izračunavanje vrijednosti karakterističnih funkcija koje se koriste u predloženim modelima.

Primjer 3.2. *Neka je data reakcija*

$$ATP + AC + COA \rightarrow AMP + PPI + ACCOA.$$

koja je dio anaboličkog puta. Metaboliti koji učestvuju u ovoj reakciji su: adenozin trifosfat (ATP), acetil (AC), koenzim A (COA), adenozin monofosfat (AMP), fosfatne grupe (PPI) i acetil koenzim A (ACCOA). U ovoj reakciji će ATP osloboditi dvije fosfatne grupe, koje će ostati slobodne i pri tome će preći u "nižu" formu - AMP. Reaktanti i proizvodi reakcije imaju sljedeće oznake klasa:

1. *ATP - Label 1: donor jedne ili dvije fosfatne grupe*
2. *AC - Label 8: ostali metaboliti*
3. *COA - Label 8: ostali metaboliti*
4. *$\rightarrow$ Label 9: strelica*
5. *AMP - Label 5: niža forma ATP-a u anaboličkom putu*
6. *PPI - Label 3: slobodna fosfatna grupa/grupe*
7. *ACCOA - Label 8: ostali metaboliti*

Svakom metabolitu u reakciji je pridružen redni broj njegove pozicije. Na poziciji $t = 3$ se nalazi metabolit $x_3 = COA$ i vrijednosti karakterističnih funkcija za ovu poziciju su:

$$f_{x,y}^{-2} = \begin{cases} 1, x{=}ATP\ i\ y{=}Label\ 8 \\ 0, inače \end{cases}$$

$$f_{x,y}^{-1} = \begin{cases} 1, x{=}AC\ i\ y{=}Label\ 8 \\ 0, inače \end{cases}$$

$$f_{x,y}^{0} = \begin{cases} 1, x{=}COA\ i\ y{=}Label\ 8 \\ 0, inače \end{cases}$$

$$f_{x,y}^{+1} = \begin{cases} 1, x{=}\rightarrow\ i\ y{=}Label\ 8 \\ 0, inače \end{cases}$$

$$f_{x,y}^{+2} = \begin{cases} 1, x{=}AMP\ i\ y{=}Label\ 8 \\ 0, inače \end{cases}$$

$$f^{-1,0}_{x,x',y} = \begin{cases} 1, x\text{= } AC,\ x\text{' = } COA\ i\ y\text{=Label 8} \\ 0, inače \end{cases}$$

$$f^{0,+1}_{x,x',y} = \begin{cases} 1, x\text{= } COA,\ x\text{' = } \rightarrow\ i\ y\text{=Label 8} \\ 0, inače \end{cases}$$

$$g^{-1,0}_{x',x,y',y} = \begin{cases} 1, x\text{'= } AC,\ x\text{ = } COA,\ y\text{' = Label 8 } i\ y\text{=Label 8} \\ 0, inače \end{cases}$$

$$g^{0,+1}_{x',x,y',y} = \begin{cases} 1, x\text{'= } COA,\ x\text{ = } \rightarrow,\ y\text{' = Label 8 } i\ y\text{=Label 8} \\ 0, inače \end{cases}$$

$$g^{-2,-1,0}_{x'',x',x,y',y} = \begin{cases} 1, x\text{''= } ATP,\ x\text{' = } AC,\ x\text{ = } COA,\ y\text{' = Label 8 } i\ y\text{=Label 8} \\ 0, inače \end{cases}$$

$$g^{-1,0,1}_{x'',x',x,y',y} = \begin{cases} 1, x\text{''= } AC,\ x\text{' = } COA,\ x\text{ = } \rightarrow,\ y\text{' = Label 8 } i\ y\text{=Label 8} \\ 0, inače \end{cases}$$

$$g^{0,+1,+2}_{x'',x',x,y',y} = \begin{cases} 1, x\text{''= } COA,\ x\text{' = } \rightarrow,\ x\text{ = } AMP,\ y\text{' = Label 8 } i\ y\text{=Label 8} \\ 0, inače \end{cases}$$

Karakteristične funkcije za druge metabolite mogu se analizirati na sličan način. Da ne bi došlo da zabune, važno je napomenuti da je u ovom primjeru PPI oznaka metabolita, a u ostatku teksta oznaka mreže proteinskih interakcija.

3.4 Rezultati testiranja

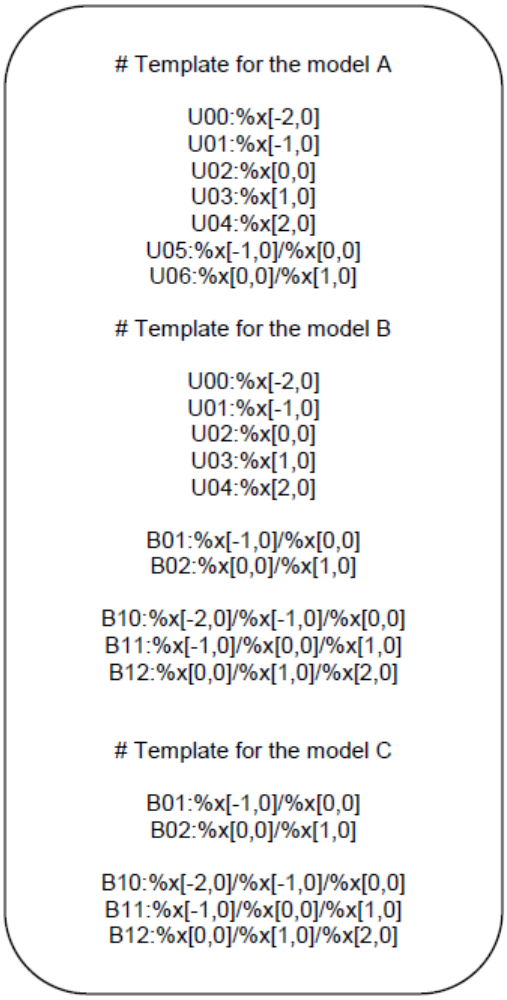
Testiranja iz ovog poglavlja su vršena na računaru Intel i5 @2.5 GHz sa 8 GB RAM. Za implementiranje predloženih modela A, B i C korišten je softverski paket CRF++ [77]. CRF++ dozvoljava upotrebu korisnički definisanih šablona, pa je pogodan za pristup koji uključuje korisnički definisane karakteristične funkcije. Za izračunavanja modela uslovnih slučajnih polja koriste se algoritam unaprijed - unazad i logaritamska izračunavanja odgovarajućih karakterističnih funkcija, čime se izbjegavaju prekoračenja [139].

Dodatno, CRF++ paket dozvoljava podešavanje još dva parametra:

- parametra *c*, koji se koristi za postizanje balansa između preprilagođavanja i potprilagođavanja. Podrazumijevana vrijednost ovog parametra je 1, a u ovom istraživanju testirane su još dvije dodatne vrijednosti: 1.5 i 2.

- parametra *f*, koji je cijeli broj i predstavlja prag odsijecanja (engl. cut-off threshold) za atribute, pri čemu se atributi formiraju na osnovu predloženih šablona i skupa trening podataka. Jedino atributi čiji je broj pojavljivanja veći ili jednak od *f* se uzimaju u obzir. Podrazumijevana vrijednost ovog parametra je 1. Testirane su još dvije vrijednosti: 2 i 3.

Skup podataka koji je korišten za testiranje sadrži biološke informacije o metabolizmu organizma *Saccharomyces cerevisiae - yeast*. Preciznije, skup se sastoji od liste od 157 metaboličkih reakcija koje su preuzete iz [46]. Iz liste koja sadrži sve metaboličke reakcije pomenutog organizma izabrane su one reakcije koje su dio procesa transfera energije ili procesa fosforilacije.

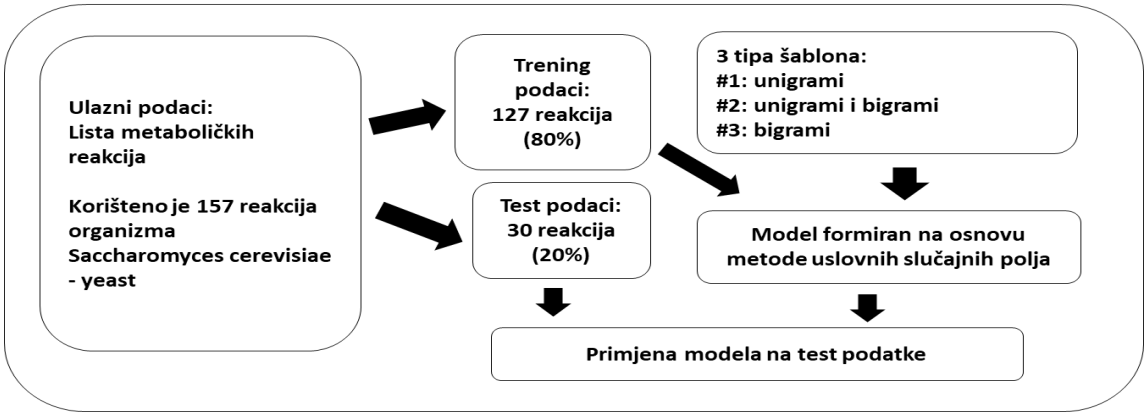


Slika 3.2: Izvod iz datoteke korištenih šablona

Proces testiranja se sastoji od četiri faze. U prvoj fazi skup izabranih reakcija je podijeljen na trening i test podatke u odnosu 80% : 20%. Šabloni, koji se formiraju u drugoj fazi, uzimaju u obzir informacije o posmatranom metabolitu i o drugim metabolitima koji učestvuju u istoj reakiji. Koriste se tri različita šablona koji odgovaraju modelima A, B i C, koji su opisani u Odjeljku 3.3.2. Šabloni čiji naziv počinje sa U nazivaju se unigrami i koriste samo informacije o oznaci trenutno posmatranog elementa, dok su šabloni sa oznakom B bigrami i pored informacije o oznaci trenutno posmatranog elementa, u obzir uzimaju i informaciju o oznaci elementa koji mu prethodi u reakciji. Primjer korištenih šablona dat je na Slici 3.2. Druga koordinata (*col*) u zapisima šablona *%x[row,col]* predstavlja poziciju karakteristike. Sa date slike se može primijetiti da je ta druga koordinata uvijek 0, što je posljedica činjenice da su jedine karakteristike koje se koriste sami metaboliti, koji se nalaze na indeksu 0. Nakon formiranja šablona, algoritam ulazi u treću fazu u kojoj vrši konstrukciju modela uslovnih slučajnih polja koji je baziran na trening podacima i fajlu šablona. Konačno, u četvrtoj fazi dobijeni model se primjenjuje na test podatke. Grafički prikaz kompletnog algoritma je dat na Slici 3.3.

Da bi se dobila dublja analiza predloženog modela uslovnih slučajnih polja, izvršen je niz testiranja sa sljedećim kombinacijama kontrolnih parametara:

- podrazumijevana kombinacija, u kojoj je *f=1* i *c=1*;



Slika 3.3: Grafički prikaz kompletnog postupka primjene CRF metode na određivanje uloge metabolita

Tabela 3.1: Rezultati testiranja dobijeni CRF++

Parametri / Model	*f=3, c=1.5*	*f=2, c=1.5*	*f=3, c=2*	*f=2, c=2*	*f=1, c=1*
Model A	93.60465%	94.18605 %	94.18605%	94.18605%	93.02326%
Model B	93.60465%	93.60465%	93.60465%	93.60465%	91.86047%
Model C	66.86047%	68.02326%	66.86047%	68.02326%	77.32558%

- *f=3* i *c=1.5*;
- *f=2* i *c=1.5*;
- *f=3* i *c=2*;
- *f=2* i *c=2*.

Dobijeni rezultati su prikazani u Tabeli 3.1. Prva kolona sadrži ime modela, a u ostatku tabele, za svaku kombinaciju kontrolnih parametara i za svaki model, prikazana je dobijena tačnost nad test podacima. Tačnost je računata na standardni način, kao odnos korektno klasifikovanih elemenata i ukupnog broja elemenata.

Iz Tabele 3.1 se može zaključiti da su rezultati dobijeni modelima A i B tačniji od rezultata dobijenih modelom C, te da su rezultati dobijeni modelom A nešto bolji od rezultata dobijenih modelom B. Pri tome razlika između rezultata dobijenih modelom A pri različitim kombinacijama parametara je veoma mala, što ukazuje na to da je model A najstabilniji. Ako posmatramo sva tri modela može se primijetiti da se pri kombinaciji podrazumijevanih parametara (posljednja kolona Tabele 3.1) za modele A i B dobijaju nešto lošiji rezultati u odnosu na druge kombinacije parametara, dok su za model C oni značajno bolji.

Da bi se validirao kvalitet predloženih modela, isti skup podataka je prilagođen i testiran drugim softverom, koji se takođe zasniva na metodi uslovnih slučajnih polja, - CRFsuite. Ovaj softver čita trening podatke i automatski generiše sva neophodna stanja i tranzicije atributa na osnovu tih podataka [115]. Tačnost modela dobijenog CRFsuite-om je 94.76% što je veoma blizu najboljim rezultatima koji su dobijeni modelom A.

U cilju daljeg razmatranja performansi predloženih modela urađena je komparativna analiza dobijenih rezultata za svaku od devet oznaka klasa upotrebom oba paketa, koja su implementirana

Tabela 3.2: Performanse modela A sa kontrolnim parametrima *f=3, c=2*

klasa	*#match*	*#model*	*#ref*	*precision*	*recall*	*F1*
Label 1	30	31	30	0.967742	1	0.9836066
Label 2	2	2	2	1	1	1
Label 3	11	11	12	1	0.917	0.9565217
Label 4	11	12	17	0.916667	0.647	0.7586207
Label 5	27	28	27	0.964286	1	0.9818182
Label 6	2	2	2	1	1	1
Label 7	1	1	1	1	1	1
Label 8	47	54	50	0.87037	0.94	0.9038462
Label 9	31	31	31	1	1	1
prosjek				0.968785	0.945	0.9538237

na osnovu metode uslovnih slučajnih polja. Za CRF++ paket izabrana je kombinacija modela A i kontrolnih parametara *f=3, c=2*, kojom su postignuti najbolji rezultati. Za svaku od devet identifikovanih oznaka klasa izračunate su tri mjere:

- preciznost
$$precision = \frac{\#match}{\#model}$$
- odziv
$$recall = \frac{\#match}{\#ref}$$
- *F1* mjera
$$F1 = \frac{2 * precision * recall}{precision + recall}$$

gdje su $\#match, \#model$ i $\#ref$ ukupan broj pogođenih oznaka, ukupan broj oznaka u modelu i ukupan broj oznaka u skupu testnih podataka, respektivno.

Rezultati dobijeni za svaku od oznaka klasa modelom A i softverom CRF++, kao i softverom CRFsuite su prikazani u Tabelama 3.2 i 3.3, respektivno. Obje tabele su organizovane na sljedeći način. Prva kolona sadrži oznaku klase, u sljedeće tri kolone prikazane su vrijednosti $\#match$, $\#model$ i $\#ref$, a zadnje tri kolone sadrže vrijednosti mjera preciznost, odziv i F1 mjera. U posljednjem redu su prikazane prosječne vrijednosti navedenih mjera.

Iz Tabela 3.2 i 3.3 se može primijetiti da oba modela ostvaruju visoku tačnost za sve oznake klasa. Ako se posmatraju prosječne vrijednosti svake od mjera može se vidjeti da je model A nešto bolji od CRFsuite modela. Takođe, rezultati iz ovih tabela ukazuju da se oba modela slično ponašaju nad razmatranim skupom podataka.

3.5 Završna razmatranja

Klasifikacija metabolita po njihovim ulogama može biti od velike koristi za bolje razumijevanje metaboličkih procesa različitih organizama. Predloženi pristup klasifikacije je baziran na metodi uslovnih slučajnih polja. Metaboličke reakcije se posmatraju kao nizovi elemenata, što omogućava formiranje različitih karakterističnih funkcija na osnovu unigrama i/ili bigrama. Razvijena su tri

Tabela 3.3: Performanse CRFsuite modela

klasa	*#match*	*#model*	*#ref*	*precision*	*recall*	*F1*
Label 1	29	30	30	0.966667	0.966667	0.966667
Label 2	2	2	2	1	1	1
Label 3	10	11	12	0.909091	0.833333	0.869565
Label 4	16	20	17	0.8	0.941176	0.864865
Label 5	27	29	27	0.931034	1	0.964286
Label 6	1	1	2	1	0.5	0.666667
Label 7	1	1	1	1	1	1
Label 8	46	47	50	0.978723	0.92	0.948454
Label 9	31	31	31	1	1	1
prosjek				0.953946	0.906797	0.920056

različita modela za formiranje šablona, koji se dalje koriste za formiranje modela uslovnih slučajnih polja. Predloženi modeli su testirani na skupu stvarnih bioloških podataka. Dobijeni rezultati ukazuju na visoku tačnost predloženih modela. Detaljna analiza pokazuje da je razvijeni model pronašao odgovarajuće oznake za većinu metabolita, što dalje ukazuje da se može koristiti za rješavanje posmatranog problema.

Istraživanje prikazano u ovom poglavlju se može proširiti na nekoliko načina. Prije svega može se vršiti primjena predloženog metoda na druge skupove podataka, posebno na veće skupove podataka. Takođe, bilo bi interesantno razviti modele koji se zasnivaju na karakterističnim funkcijama koje sadrže dodatne informacije o metabolitima.

Glava 4

Particionisanje bioloških mreža na visoko povezane komponente

4.1 Uvod

U ovom poglavlju razmatran je problem particionisanja velikih bioloških mreža u visoko povezane komponente (engl. highly connected components), uklanjanjem što je moguće manje grana. Odnosno, razmatra se particionisanje velike biološke mreže na komponente čija je povezanost visoka, a između samih komponenti nema puno veza (grana). Graf sa n čvorova se smatra visoko povezanim (engl. highly connected) ako je stepen svakog čvora veći od $n/2$. Najmanja visoko povezana komponenta je trougao. Odnosno, duž i pojedinačni čvor se ne smatraju visoko povezanim komponentama. U literaturi je ovaj problem poznat pod nazivom Problem brisanja grana uz očuvanje visoke povezanosti (engl. Highly connected deletion problem - HCD). Navedeni problem ima nekoliko primjena u računarskoj biologiji, npr. za pronalaženje grupa proteina sa sličnom GO anotacijom [67] ili pronalaženje grupe gena sa sličnim profilom ekspresije [57].

Brojni su radovi u literaturi koji se bave problemom klasterovanja grafova. Klasterovanje podrazumijeva podjelu skupa podataka na grupe, tako da su podaci u istoj grupi međusobno sličniji nego podaci koji se nalaze u različitim grupama. Pri klasterovanju grafova podaci su najčešće predstavljeni čvorovima, a grane između čvorova predstavljaju veze između podataka. Pregled nekih problema klasterovanja grafova je već prikazan u Poglavlju 2, dok se u ovom poglavlju posebno razmatraju problemi klasterovanja koji su zasnovani na brisanju grana iz polaznog grafa. Jedan od najčešćih problema koji se razmatra pri klasterovanju grafova je problem brisanja što je moguće manje grana iz grafa da bi dobijene komponente povezanosti bile klike. U literaturi je ovaj NP težak problem poznat pod nazivom Cluster Deletion Problem [133]. Za p-verziju ovog problema (engl. p-Cluster Deletion Problem), odnosno problem brisanja što je moguće manje grana tako da se dobije p komponenti poveznanosti koje su klike, u [133] je pokazano da se za $p = 2$ može riješiti u polinomskom vremenu, a da je za $p > 2$ taj problem NP kompletan. Problem koji se razmatra u ovom poglavlju zapravo postavlja manje striktne uslove za komponente povezanosti nego problemi iz [133]. Preciznije, dozvoljene su i komponente rjeđe od klika, ali ipak dovoljno guste da sačuvaju važne informacije o samoj strukturi. Pored ovog problema, u literaturi postoje brojni problemi koji se bave particionisanjem grafova na komponente koje su relaksirane klike, odnosno gusti grafovi koji nisu klike. U [89] se vrši particionisanje na *s-club* podgrafove brisanjem što je moguće manje grana (engl. s-Club Cluster Edge Deletion Problem). Podskup skupa čvorova $S \subset V$, grafa $G = (V, E)$, se naziva *s-club* ako

je dijametar podgrafa indukovanog skupom čvorova S najviše s. U [89] je pokazano da je za $s = 2$ problem NP kompletan i predstavljen je algoritam fiksnih parametara za rješavanje ovog problema. Preciznije, pokazano je da je za dati fiksni parametar k vremenska zavisnost predstavljenog algoritma $O(2.74^k P(n))$, gdje je $P(n)$ polinom od n, a n dimenzija problema. U literaturi se takođe mogu naći problemi koji se bave klasterovanjem grafova ne samo minimizacijom broja grana koje se brišu, nego i minimizacijom broja operacija dodavanja ili brisanja grana [133] ili brisanjem što je moguće manje čvorova [89]. Pregled definicija još nekih gustih podgrafova i algoritama za njihovu identifikaciju se može naći u [84].

Problem pronalaženja kompletnog skupa često ukrštenih-po-grafovima kvazi-klika (engl. frequent cross-graph quasi-cliques) je razmatran u [69]. Za skup čvorova S u nekom grafu kažemo da je γ kvazi-klika, za $0 < \gamma \leqslant 1$ ako je svaki čvor iz S direktno povezan sa bar $\gamma(|S| - 1)$ drugih čvorova iz S. Neka je dat skup grafova $G_1, ..., G_n$ i parametar $0 < min\_sup \leqslant 1$, skup čvorova S je skup često ukrštenih-po-grafovima kvazi-klika ako je S γ kvazi-klika u najmanje $min\_sup \cdot n$ grafova i ne postoji nijedan pravi podskup od S sa tom osobinom. U navedenom radu je dato rješenje koje je kombinacija nekoliko efikasnih tehnika i heuristika za redukciju broja čvorova, broja grana, formiranje kombinacije grafova i slično. Biološka validacija rezultata, dobijenih primjenom na PPI mreže, je pokazala da ova metoda identifikuje neke od poznatih proteinskih kompleksa. Za vrijednost parametra $\gamma = 0.5$ ovaj problem je sličan HCD problemu.

Navedeni teorijski koncepti mogu poslužiti kao osnova za analizu kompleksnih, velikih, različitih bioloških mreža. Tako je u [5] dat pregled kako se pomoću grafovske interpretacije i analize može doći do boljeg biološkog razumijevanja mreža ćelijskih interakcija. Jedno od zapažanja prikazano u navedenom radu je da se za identifikaciju motiva u PPI mrežama često koriste klike. Metode za predviđanje funkcionalne anotacije proteina na osnovu PPI mreže mogu koristiti samo informacije iz najbližeg susjedstva u mreži, globalnu topologiju cijele mreže ili se zasnivaju na Markovljevoj pretpostavci (funkcija proteina je nezavisna u odnosu na ostale proteine u mreži ako su date funckije neposrednih susjeda) [135]. U Odjeljku 4.5 će biti prikazan primjer upotrebe visoko povezanih komponenti PPI mreže kao osnove za predloženu metodu predviđanja novih GO anotacija koje se dodjeljuju proteinima.

Definicija visoko povezanog grafa je uvedena u [57], gdje je predložen algoritam za pronalaženje visoko povezanih podgrafova (engl. Highly Connected Subgraphs - HCS), koji iterativno uklanja mali broj grana sve dok dobijene komponente nisu visoko povezane. Iako je time garantovano da su dobijene komponente visoko povezane, algoritam iterativno koristi pohlepni korak za brisanje grana, čime se ne garantuje maksimalan broj grana unutar komponenti, odnosno minimalan broj grana između komponenti. Dakle, time nije obezbijeđeno da će broj obrisanih grana biti minimalan. U istraživanju predstavljenom u [67] formalno je uveden odgovarajući problem kombinatorne optimizacije, za koji je pokazano da je NP težak. Problem je rješavan egzaktnom ILP metodom i dvjema heurističkim metodama. Navedene metode su primijenjene na redukovane instance (neka od pravila redukcije su opisana u Odjeljku 4.2.2). Egzaktan ILP metod (ILP - Column Generation) je uspio da pronađe rješenja za sve razmatrane instance, osim jedne. Prva heuristička metoda je zasnovana na *min cut* algoritmu iz [57], dok je druga zasnovana na heurističkom pretraživanju okolina (engl. Neighborhood Heuristic), koja primjenjuje pohlepno brisanje grane čiji krajevi imaju najmanje zajedničkih susjeda.

U ovom poglavlju se posmatrani problem rješava metodom promjenljivih okolina.

4.2 Rješavanje HCD problema

4.2.1 Definicija problema

Neka je dat graf $G = (V, E)$ sa skupom čvorova V i skupom grana E. Problem brisanja minimalnog broja grana uz očuvanje visoke povezanosti podrazumijeva pronalaženje skupa grana $E' \subset E$ minimalne kardinalnosti, koje je potrebno obrisati, tako da svaka komponenta novonastalog grafa $G' = (V, E \setminus E')$ bude visoko povezana. Za komponentu od s čvorova se kaže da je visoko povezana ako je stepen svakog čvora u toj komponenti veći od $s/2$. Particija $\mathcal{P} = (V_1, V_2, ..., V_l)$, skupa čvorova u l disjunktnih komponenti takvih da je $\bigcup_{i=1}^{l} V_i = V$, je particija na visoko povezane komponente ako važi

$$\forall V_i \in \mathcal{P},\ \forall v \in V_i,\ (\deg(v) > |V_i|/2), \tag{4.1}$$

gdje je $\deg(v)$ stepen čvora u podgrafu koji je indukovan skupom čvorova V_i. Rješenja koja zadovoljavaju uslov (4.1) su dopustiva rješenja, dok rješenja koja ne zadovoljavaju (4.1) su nedopustiva. Čvor koji nije povezan sa više od polovine čvorova u komponenti kojoj pripada smatra se nekorektnim. Kao što je već napomenuto, kompletan graf sa 2 čvora (u oznaci K_2) se ne smatra visoko povezanom komponentom, što je i u skladu sa datim uslovom (4.1). Svi singltoni se smatraju neklasterovanim.

4.2.2 Faza pretprocesiranja

Zbog ubrzanja kompletnog procesa, korisno je prije primjene konkretne metode obrisati one grane koje se ne mogu javiti ni u jednom dopustivnom rješenju. Brisanje takvih grana se obavlja u fazi pretprocesiranja i izvršava se u polinomskom vremenu. Preciznije, u implementaciji koja je primijenjena u ovom istraživanju u vremenu $O(n^4)$, dok bi se uz upotrebu drugačijih struktura podataka mogla realizovati i u vremenu $O(n^3)$, gdje je n broj čvorova u grafu. U [67] je korišteno pet pravila na osnovu kojih se brišu grane u fazi pretprocesiranja, dok je u istraživanju, koje je prikazano u ovom poglavlju, implementirano jedno od tih pravila, tačnije pravilo:

Ako postoje dva čvora u i v takva da su povezana granom, ali da nemaju nijednog zajedničkog susjeda, onda obrisati granu koja ih povezuje i broj obrisanih grana uvećati za 1.

Od svih pravila redukcije prikazanih u [67], ovo pravilo u najvećoj mjeri pojednostavljuje polazne grafove, pa je iz tog razloga implemetirano i u ovom istraživanju. Formalno, dato pravilo slijedi direktno iz sljedeće leme.

Lema 1. *[67] Neka je G visoko povezan graf i neka su u i v dva čvora u grafu G. Ako su u i v povezani granom, onda imaju najmanje jednog zajedničkog susjeda, u suprotnom imaju najmanje tri zajednička susjeda.*

Dokaz. Neka je $|V| = n$ i neka je sa n_{uv} označen broj zajedničkih susjeda za čvorove u i v. Sa n_u i n_v je označen broj susjeda čvora u ne računajući čvor v i njihove zajedničke susjede, odnosno broj susjeda čvora v ne računajući čvor u i njihove zajedničke susjede, respektivno. Neka je

$$c = \begin{cases} 1, & \text{ako } \{u, v\} \in E, \\ 0, & \text{u suprotnom.} \end{cases}$$

Tada vrijedi

$$n_{uv} + n_u + c > n/2$$

jer lijeva strana predstavlja broj susjeda čvora u, koji zbog činjenice da je G visoko povezan graf, mora biti veći od $n/2$. Analogno, vrijedi i

$$n_{uv} + n_v + c > n/2,$$

pa je

$$2n_{uv} + n_u + n_v + 2c \geqslant n + 1.$$

S obzirom da je

$$n \geqslant n_{uv} + n_u + n_v + 2,$$

slijedi

$$n_{uv} + 2c - 2 \geqslant 1,$$

odnosno

$$n_{uv} \geqslant 3 - 2c.$$

□

Više o ostalim pravilima i teorijskim osnovama na kojima se zasnivaju može se naći u [67].

4.2.3 Metoda promjenljivih okolina za rješavanje HCD problema

Kao što je već pomenuto u uvodnom dijelu, za rješavanje i ovog problema razvijena je metoda promjenljivih okolina. Prije primjene same metode, izvršeno je pretprocesiranje, čiji je opis dat u Odjeljku 4.2.2. Osnovni principi funkcionisanja VNS-a dati su u uvodnom poglavlju, dok će u narednim odjeljcima biti detaljno opisana VNS metoda koja je razvijena za rješavanje HCD problema.

Ulazni podaci za predloženi VNS algoritam su:

- graf $G = (V, E)$;
- n_{min} i n_{max} su minimalna i maksimalna veličina okoline koju razmatra VNS;
- it_{max}, $itrep_{max}$ su maksimalan broj iteracija i maksimalan broj iteracija bez poboljšanja;
- *prob* je vjerovatnoća prelaska iz jednog rješenja u drugo rješenja istog kvaliteta.

4.2.4 Inicijalizacija i funkcija cilja

Rješenje predloženog VNS algoritma je predstavljeno nizom $\mathbf{x}$ cijelih brojeva dužine $|V|$. Svaki element niza odgovara jednom čvoru grafa, označavajući kojoj komponenti pripada odgovarajući čvor. Preciznije, čvor i je pridružen komponenti V_j ako je $x_i = j$.

Inicijalno rješenje se formira tako da svaki čvor čini pojedinačnu komponentu, odnosno $x_i = i$, za $1 \leqslant i \leqslant |V|$. Drugim riječima, inicijalno rješenje je formirano tako da su svi čvorovi neklasterovani. Tokom procesa pretraživanja moguće je da se pojave i nedopustiva rješenja, u smislu da jedna ili više komponenti tog rješenja ne budu visoko povezane. Pojava nedopustivih rješenja u nekim fazama algoritma je dozvoljena jer se "popravljanjem" takvih nedopustivih rješenja mogu dobiti kvalitetnija dopustiva rješenja nego u slučaju kada se prostor pretraživanja ograniči isključivo na dopustiva rješenja. U tu svrhu, uvedena je specifična funkcija cilja koja sa jedne strane sadrži informaciju o broju obrisanih grana, a sa druge kažnjava nedopustiva rješenja.

Neka je particija $\mathcal{P} = (V_1, V_2, ..., V_l)$ predloženo, ne obavezno dopustivo rješenje HCD problema. Neka je sa V_{nc} označen broj nekorektnih čvorova u posmatranom rješenju, a sa e_d broj obrisanih grana. VNS funkcija cilja koja se minimizuje predstavljena je formulom

$$obj_{VNS}(\mathcal{P}) = V_{nc} + \frac{e_d}{|E|}. \tag{4.2}$$

Ako je rješenje dopustivo, onda su svi čvorovi korektni pa je prvi sabirak u formuli 4.2 jednak nuli. Drugi sabirak je odnos između broja obrisanih grana i ukupnog broja grana u polaznom grafu, pa je zbog toga manji ili jednak od 1. Pored toga, ako e_d ima manju vrijednost (odnosno, ako je broj obrisanih grana manji), to je i ovaj sabirak manji. Dakle, za dva dopustiva rješenja, vrijednosti funkcija cilja će se porediti po broju obrisanih grana, jer će za oba prvi sabirak biti jednak nuli. S druge strane, ako se porede vrijednosti funkcije cilja za jedno dopustivo $\mathcal{P}_d$ i jedno nedopustivo rješenje $\mathcal{P}_{nd}$ uvijek će važiti $obj_{VNS}(\mathcal{P}_d) < obj_{VNS}(\mathcal{P}_{nd})$, jer će nedopustivo rješenje imati bar jedan nekorektan čvor što će uticati da $obj_{VNS}(\mathcal{P}_{nd}) > 1$. U slučaju poređenja dva nedopustiva rješenja prednost će imati ono rješenje koje ima manje nekorektnih čvorova.

4.2.5 Procedura razmrdavanja

Iz okoline trenutno najboljeg rješenja, u proceduri razmrdavanja se bira novo rješenje u cilju izbjegavanja situacije da algoritam "zaglavi" u suboptimalnom rješenju. Da bi se to postiglo, formira se sistem okolina oko trenutno najboljeg rješenja $\mathbf{x}$.

Procedura razmrdavanja je slična proceduri razmrdavanja opisanoj u Odjeljku 2.3.4, koja je razvijena za rješavanje Max-EkP problema. Za formiranje κ-te okoline na slučajan način se bira κ čvorova iz skupa V. Zatim se, za svaki izabrani čvor, na slučajan način bira komponenta u koju će biti premješten. Odnosno, ako je l ukupan broj komponenti, onda se cijeli broj q na slučajan način bira iz skupa $\{1, 2, ..., l + 1\}$. Ovako definisana procedura razmrdavanja omogućava promjenu ukupnog broja particija. Ako je $q < l + 1$, onda se čvor premješta u postojeću particiju V_q. Ako je $q = l + 1$, onda se formira nova particija koja sadrži samo čvor koji se premješta i ukupan broj particija se povećava za jedan. Ako time particija u kojoj se prethodno nalazio izabrani čvor postane prazna, onda se ukupan broj particija smanjuje za jedan.

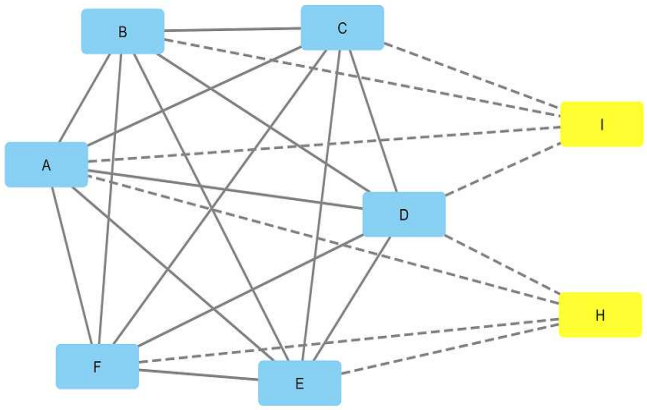
Rješenje $\mathbf{x}'$, koji se dobija nakon procedure razmrdavanja, je predmet daljeg unapređenja, prvo u proceduri spajanja komponenti, a nakon toga i u fazi lokalne pretrage. Navedene procedure su opisane u narednim odjeljcima.

4.2.6 Procedura spajanja komponenti

Procedura `join_components` ima za cilj poboljšanje posmatranog rješenja tako što razmatra mogućnosti sljedećih spajanja:

- spajanje tri singltona u visoko povezanu komponentu K_3 (trougao);
- spajanje dvije visoko povezane komponente u jednu.

Singltoni koji su nastali particonisanjem možda mogu biti spojeni u trougao. Zbog toga, u proceduri `join_components` se pokušava sa pronalaženjem takvih trouglova i time se nastoji smanjiti broj neklasterovanih čvorova. U tom cilju, za dva slučajno izabrana čvora koja su u početnom grafu povezana granom, a koji su u trenutnom rješenju singltoni (odnosno svaki je klasterovan u zasebnu



Slika 4.1: Jedna visoko povezana komponenta od 6 čvorova i 2 singltona

komponentu koja je singlton), algoritam traži treći singlton (ako takav postoji) sa kojim mogu formirati trougao na osnovu prisustva grana u početnoj mreži. Ako je to moguće, onda ta tri čvora formiraju jednu visoko povezanu komponentu, kompletno povezan graf od tri čvora K_3.

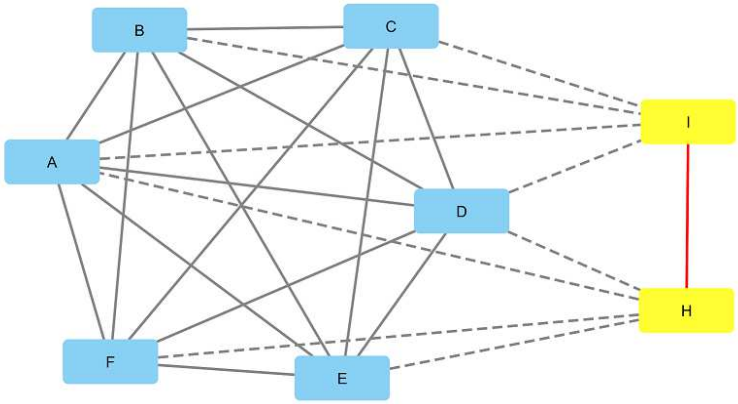
Nakon formiranja trouglova, procedura `join_components` ponovo razmatra neklasterovane singltone i pokušava da ih spoji u duži, ako postoji grana između njih u početnoj mreži. Razlog za ovo spajanje je pretpostavka da će se dva singltona lakše dodati nekoj od visoko povezanih komponenti ako su ta dva singltona povezana granom. Treba napomenuti da dodavanje i nepovezanih singltona takođe može povećati komponentu, ali su za to šanse manje nego u slučaju singltona koji su u početnoj mreži povezani granom. Stoga se, zbog efikasnosti, u ovoj proceduri razmatraju samo singltoni koji su u početnoj mreži povezani granom. Sljedeći primjer ilustruje kako postojanje grane između singltona može da poboljša rješenje u odnosu na situaciju kada takve grane nema.

Primjer 4.1. *Na Slici 4.1 prikazana je jedna visoko povezana komponenta i 2 singltona. Struktura sastavljena od šest plavih čvorova $\{A, B, C, D, E, F\}$ je visoko povezana komponenta, jer je stepen svakog čvora jednak 5. Žuti čvorovi I i H predstavljaju singltone u posmatranom rješenju. Isprekidane grane postoje u polaznoj mreži, ali su trenutno isključene iz rješenja, jer bi se njihovim uključivanjem narušio uslov visoke povezanosti (stepen čvorova I i H bi u komponenti od 8 čvorova bio 4). Na Slici 4.2 je predstavljena situacija kada između ova dva singltona postoji grana. U ovoj situaciji stepeni posmatranih čvorova imaju vrijednosti: $deg(A) = 7$, $deg(B) = 6$, $deg(C) = 6$, $deg(D) = 6$, $deg(E) = 6$, $deg(F) = 6$, $deg(I) = 5$ i $deg(H) = 5$. Kao što se može primijetiti, stepen svakog čvora je veći od 4, pa svih osam čvorova čini visoko povezanu komponentu. Dakle, postojanje ove grane obezbjeđuje korektnost svakog čvora i smanjuje ukupan broj obrisanih grana za 9.*

U posljednjoj fazi izvršenja procedure `join_components` razmatraju se svi parovi različitih komponenti trenutnog rješenja, koje nisu singltoni. Procedura pokušava da spoji par komponenti u jednu i u slučaju da je novoformirana komponenta visoko povezana, ona se uključuje u rješenje.

4.2.7 Lokalna pretraga

Za poboljšanje rješenja koje je dobijeno nakon primjene procedura razmrdavanja i spajanja komponenti, koriste se dvije procedure lokalnog pretraživanja, LS1 i LS2. Procedura LS1 je vremenski manje zahtjevna i primjenjuje se u svakoj iteraciji, dok je procedura LS2 vremenski zahtjevnija i primjenjuje se samo u situacijama kada procedura LS1 nije uspjela da poboljša rješenje nakon unaprijed



Slika 4.2: Jedna visoko povezana komponenta sa 8 čvorova

zadatog broja iteracija, koji je u ovoj implementaciji podešen na 1000.

Rješenje $\mathbf{x}''$, dobijeno nakon primjene procedura spajanja komponenti i lokalne pretrage, se dalje razmatra na sljedeći način. Ako je vrijednost VNS funkcije cilja za rješenje $\mathbf{x}''$ manja od vrijednosti funkcije cilja za trenutno najbolje rješenje $\mathbf{x}$, onda $\mathbf{x}''$ postaje novo trenutno najbolje rješenje ($\mathbf{x} = \mathbf{x}''$). Ako je vrijednost VNS funkcije cilja za rješenje $\mathbf{x}''$ veće od vrijednosti funkcije cilja za rješenje $\mathbf{x}$, onda rješenje $\mathbf{x}$ ostaje trenutno najbolje. Ako su vrijednosti VNS funkcija cilja za oba rješenja jednake, onda se $\mathbf{x}$ postavlja na $\mathbf{x}''$ sa vjerovatnoćom *prob*. U ovoj implementaciji parametar *prob* je 0.5.

Lokalna pretraga LS1

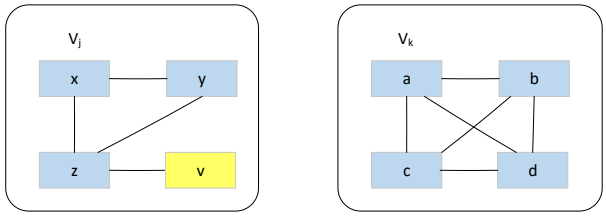
Unutar procedure LS1, poboljšanje se pokušava postići pomjeranjem izabranog čvora iz komponente kojoj trenutno pripada u neku drugu komponentu.

Na početku izvršenja LS1 procedure, na slučajan način bira se čvor v_i, koji pripada komponenti V_j, za neke $1 \leqslant i \leqslant n$ i $1 \leqslant j \leqslant l$, gdje je l broj trenutnih particija. Dalje, izabrani čvor v_i se prebacuje iz komponente V_j u slučajno izabranu komponentu V_k, za $1 \leqslant k \leqslant l+1$. Procedura LS1 koristi "strategiju prvog unapređenja" (engl. first improvement strategy), koja podrazumijeva da algoritam prelazi u novo rješenje čim dođe do poboljšanja.

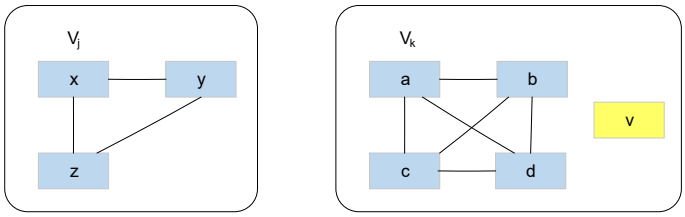
Za razliku od lokalnog pretraživanja za problem Max-EkPP prikazanog u Odjeljku 2.3.5, ovdje se izbjegava prebacivanje singltona u drugu particiju koja sadrži samo singlton, odnosno iz V_j u V_k, takve da je $|V_j| = 1$ i $|V_k| = 1$. Prebacivanje čvora koji je singlton u komponentu koja sadrži samo singlton ne vodi ka poboljšanju rješenja, jer čak i ako bi postojala grana između tih singltona u početnom grafu, opet se ne bi formirala visoko povezana komponenta (graf K_2 nije visoko povezana komponenta).

Nakon što se na slučajan način izaberu čvor v_i i komponenta V_k, vrši se brzo računanje funkcije cilja novoformiranog rješenja. Kardinalnost komponente V_j, iz koje se izbacuje izabrani čvor, se smanjuje za jedan.

Ako je slučajno izabrana komponenta V_{l+1} ($k = l+1$), formira se nova komponenta koja sadrži samo čvor v_i i ukupan broj komponenti se povećava za jedan, osim u slučaju da nakon prebacivanja komponenta V_j nije ostala prazna (tada broj komponenti ostaje isti). Ako je $k < l+1$, čvor v_i se dodaje nekoj postojećoj komponenti i ukupan broj komponenti ostaje isti, osim opet u slučaju da



Slika 4.3: Situacija prije prebacivanja čvora v u komponentu V_k



Slika 4.4: Situacija nakon prebacivanja čvora v u komponentu V_k

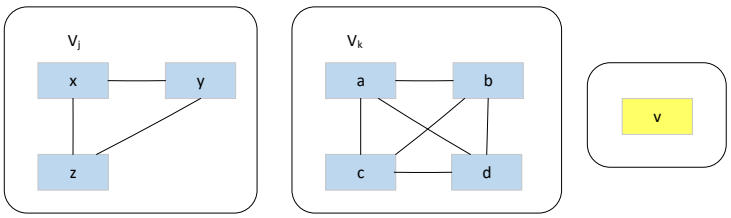
nakon prebacivanja komponenta V_j nije ostala prazna. Ovdje se vrši dodatno razmatranje: ako čvor v_i nije povezan granom ni sa jednim čvorom iz komponente V_k, onda se taj čvor ne ubacuje u tu komponentu već postaje singlton. Razlog za ovo je ilustrovan sljedećim primjerom.

Primjer 4.2. *Neka izabrani čvor v (žuti čvor na Slici 4.3) pripada komponenti V_j i neka komponenta V_k, u koju će biti prebačen čvor v, ima strukturu kao na Slici 4.3. Sa slike se vidi da u particiji koja sadrži ove dvije komponente postoje 3 nekorektna čvora (x,y,v).*

Prebacivanjem čvora v u komponentu V_k dobija se situacija prikazana na Slici 4.4. Kako čvor v nije povezan granom ni sa jednim čvorom iz komponente V_k, njegov stepen je nula, pa je on jedini nekorektan čvor u ove dvije komponente. Izdvajanjem čvora v u posebnu komponentu, koja sadrži samo taj čvor, dobija se situacija prikazana na Slici 4.5, gdje su svi čvorovi korektni.

Ažuriranje particije V_j, iz koje je izbačen čvor, se vrši na sljedeći način. Prolazi se kroz sve čvorove te particije i za svaki čvor se provjerava da li je povezan granom sa čvorom v_i. U slučaju da jeste, broj obrisanih grana se povećava za jedan i stepen tog čvora se smanjuje za jedan. Nakon razmatranja svih čvorova iz particije V_j, ažurira se broj korektnih čvorova. Dalje se ažurira particija V_k, u koju je premješten čvor v_i. Razmatraju se svi čvorove particije V_k i za svaki od njih se provjerava da li postoji grana sa čvorom v_i. Ako postoji takva grana, broj obrisanih grana se smanjuje za jedan, a stepen čvora se povećava za jedan. Kao i u prethodnom slučaju, nakon ažuriranja provjerava se korektnost čvora. Nakon ažuriranja komponenti V_j i V_k, provjerava se korektnost čvora v_i i ažurira se informacija o tome.

Sljedeći korak lokalne pretrage je računanje vrijednosti funkcije cilja novog rješenja, u kojem je čvor v_i premješten u komponentu V_k. Samo izračunavanje je ubrzano jer se informacija o broju obrisanih grana koristi iz rješenja prije premještanja i mijenja se tokom ažuriranja komponenti, a broj nekorektnih čvorova se dobija tako što se od ukupnog broja čvorova oduzima broj korektnih čvorova. Informacija o broju korektnih čvorova se takođe ažurirala tokom razmatranja particija iz i u koje se vrši premještanje. Tako dobijene informacije se koriste u formuli 4.2 i time se dobija vrijednost funkcije cilja za novoformirano rješenje.



Slika 4.5: Situacija nakon izdavajanja čvora v u posebnu komponentu $\{v\}$

Lokalna pretraga LS2

Kao što je već pomenuto, zbog veće vremenske složenosti, LS2 se primjenjuje na dato rješenje samo ako je razlika između rednog broja trenutne iteracije i rednog broja iteracije u kojoj se desilo posljednje poboljšanje rješenja veća od 1000. U ovoj proceduri razmatraju se svi čvorovi na sljedeći način. Neka je čvor koji se trenutno razmatra čvora v_i, koji nije singlton. Čvor v_i se izbacuje iz komponente kojoj pripada, a traži se singlton čije ubacivanje u komponentu, iz koje je izbačen čvor v_i, vodi ka najboljem poboljšanju rješenja, ako je poboljšanje uopšte moguće. Da bi se pronašao singlton koji će dati najbolje poboljšanje rješenja, prolazi se po svim čvorovima koji su singltoni i vrši se prebacivanje u komponentu iz koje je izbačen izabrani čvor. Pri tome, kao i u proceduri LS1, ažuriraju se sve potrebne informacije i vrši se brzo računanje parcijalne funkcija cilja. Ovaj postupak se ponavlja sve dok se ne dođe do poboljšanja, tj. sve dok se ne dobije rješenje koje je bolje od trenutno najboljeg, ili dok se ne prođe po svim čvorovima koji nisu singltoni i koji su kandidati za uklanjanje iz komponente kojoj pripadaju.

4.3 Rezultati testiranja

Sva testiranja su vršena na računaru Intel i7-4770 CPU@3.40GHz sa 8 GB RAM i Windows 7 64Bit operativnim sistemom. Za svako izvršenje koristi se jedna nit/procesor. VNS algoritam je implementiran u programskom jeziku C i kompajliran Visual Studio 2019 kompajlerom.

Parametri koji kontrolišu rad VNS algoritma imaju sljedeće vrijednosti. Veličina minimalne okoline trenutnog rješenja koja se razmatra je postavljena na 1, dok je veličina maksimalne okoline trenutnog rješenja koja se razmatra postavljena na 20. Ukupan broj iteracija je postavljen na 10000, a maksimalan broj iteracija bez poboljšanja na 2000.

4.3.1 Skupovi podataka

Za testiranje predloženog algoritma korištene su dvije vrste bioloških mreža, odnosno PPI mreže i metaboličke mreže. Korištene su PPI mreže kao i u [67] i odgovaraju sljedećim organizmima:

- *Arabidopsis thaliana*;
- *Caenorhabditis elegans*;
- *Schizosaccharomyces pombe*;
- *Mus musculus*.

Testirana su dva tipa ovih mreža, odnosno mreže koje sadrže sve interakcije i mreže koje sadrže samo fizičke interakcije. U tabelama koje su prikazane u nastavku će se koristiti skraćeni nazivi ovih instanci, At, Ce, Sp i Mm, respektivno uz oznake all, za mrežu koja sadrži sve interackije, i phys, za mreže koje sadrže samo fizičke interakcije. Druga vrsta bioloških mreža su metaboličke mreže za 4 organizma:

- *Saccharomyces cerevisiae - yeast*;
- *Staphylococcus aureus*;
- *Tuberculosis -Mycobacterium tuberculosis*;
- *Escherichia coli.*

Metaboličke mreže su formirane na osnovu spiskova metaboličkih reakcija datih organizama postupkom opisanim u Odjeljku 2.4, pri čemu su metaboliti čvorovi, a grana između dva metabolita postoji ako učestvuju u bar jednoj zajedničkoj reakciji. Spiskovi reakcija su preuzeti sa sajta Systems Biology Research Group at the University of California, San Diego http://systemsbiology.ucsd.edu/Downloads. U tabelama koje su prikazane u nastavku će se koristiti skraćeni nazivi ovih instanci, Sc, Sa, Tu i Ec, respektivno.

4.3.2 Rezultati za PPI mreže

U Tabeli 4.1 su prikazane inforamcije o PPI mrežama, kao i rezultati dobijeni primjenom VNS algoritma na ovim mrežama. Prva kolona sadrži skraćeni naziv mreže. U drugoj i trećoj koloni su prikazane informacije o broju čvorova i broju grana u polaznoj mreži. Broj grana koje su obrisane u fazi redukcije je prikazan u četvrtoj koloni i označen je sa $|E_d|$. Nakon faze redukcije, u kojoj je obrisano $|E_d|$ grana, dobijene su redukovane mreže. Naredne dvije kolone sadrže informacije o redukovanim mrežama, odnosno broj čvorova (označen sa $|V_r|$) i broj grana (kolona $|E_r|$). Mreža sa $|V_r|$ čvorova i $|E_r|$ grana je ulaz u VNS algoritam. U ostatku tabele se nalaze sljedeće informacije o rezultatima dobijenim VNS algoritmom:

- k_{best} - najbolje rješenje dobijeno u 10 izvršenja VNS algoritma i predstavlja broj obrisanih grana;
- k_{avg} - prosječno rješenje dobijeno u 10 izvršenja VNS algoritma;
- $t[s]$ - prosječno vrijeme izvršenja u sekundama;
- $|comp|$ - broju visoko povezanih komponenti u najboljem rješenju;
- n_c - broj čvorova u najvećoj visoko povezanoj komponenti najboljeg rješenja;
- m_c - broj grana u najvećoj visoko povezanoj komponenti najboljeg rješenja.

Može se primijetiti da se broj grana obrisanih u fazi redukcije, razlikuje od broja obrisanih grana u fazi redukcije u [67]. To je posljedica toga da u postupku redukcije, koji je opisan u Odjeljku 4.2.2, nisu primijenjena sva pravila iz [67]. Ipak, vidi se da broj obrisanih grana nije značajno manji, što potvrđuje konstataciju da pravilo redukcije, koje je implementirano, najviše utiče na redukciju mreže.

Tabela 4.1: Informacije i rezultati nad PPI mrežama

instanca	$\|V\|$	$\|E\|$	$\|E_d\|$	$\|V_r\|$	$\|E_r\|$	k_{best}	k_{avg}	$t[s]$	$\|comp\|$	n_c	m_c
At-all	6038	13680	8799	1630	4881	3290	3317.3	2632.92	962	23	186
At-phys	5999	13571	8762	1619	4809	3247	3274.9	2921.54	957	21	154
Ce-all	3866	7707	5487	670	2220	1819	1839.7	254.89	485	17	94
Ce-phys	3176	5465	4503	396	962	681	684.4	80.46	241	9	30
Mm-all	7414	14687	10285	1531	4402	3369	3384.3	2670.31	970	13	60
Mm-phys	7354	14509	10204	1503	4305	3296	3308.6	2601.16	953	11	44
Sp-all	3735	51620	6168	2916	45452	42638	42762.4	6277.42	1955	51	854
Sp-phys	1963	4772	1918	965	2854	1920	1932.3	1069.07	598	17	96

Zbog poređenja sa rezultatima iz [67] prikazana je i Tabela 4.2. Tabela je organizovana na sljedeći način. Skraćeni naziv instance je prikazan u prvoj koloni. Zatim se za algoritme min-cut without DR, min-cut with DR, neighborhood with DR i Column Generation with DR iz [67] i predloženi VNS algoritam iz ovog istraživanja nalaze sljedeće informacije:

- k - ukupan broj obrisanih grana;
- n_c - broj čvorova u najvećoj visoko povezanoj komponenti;
- m_c - broj grana u najvećoj visoko povezanoj komponenti;
- $t[s]$ - vrijeme izvršenja u sekundama.

Za algoritme koji vrše pretprocesiranje mreža, odnosno za algoritme min-cut with DR, neighborhood with DR, Column Generation with DR iz [67] i VNS opisan u ovom poglavlju, vrijednost k koja je prikazana u Tabeli 4.2 je jednaka zbiru broja grana obrisanih u fazi pretprocesiranja i broja grana obrisanih samim algoritmom. Konkretno, za predloženi VNS algoritam ova vrijednost je zbir vrijednosti $|E_d|$ i k_{best} iz Tabele 4.1.

Upoređujući informacije o ukupnom broju obrisanih grana, koje su prikazane u Tabeli 4.2, može se primijetiti da od svih približnih algoritama (min-cut without DR, min-cut with DR, neighborhood with DR, VNS) predloženi VNS algoritam pronalazi rješenja koja su najbliža poznatim optimalnim rješenjima, dobijenim metodom Column Generation (u Tabeli 4.2 označen sa cgDR) iz [67]. Za instancu *Caenorhabditis elegans - phys* pronalazi i optimalno rješenje, dok za instancu *Schizosaccharomyces pombe-all* za koju nije poznato optimalno rješenje, od svih približnih algoritama, pronalazi rješenje kojim se briše najmanji broj grana. Egzaktni algoritam Column Generation iz [67] za instancu *Schizosaccharomyces pombe-all* nije uspio da pronađe rješenje za 32 sata, vjerovatno zbog velike gustine ove mreže. Rezultati koji se odnose na broj čvorova i broj grana u najvećoj komponenti koja je dobijena pri particionisanjima su uglavnom slični za sve algoritme.

Tabela 4.2: Rezulatati za PPI mreže

instanca	mc				mcDR				nDR				cgDR				VNS			
	k	n	m	$t[s]$	k	n_c	m_c	$t[s]$	k	n	m	$t[s]$	k	n	m	$t[s]$	k	n	m	$t[s]$
At-all	13121	23	190	616	12613	23	190	10	12222	22	178	10	11972	23	190	10536	12089	23	186	2632.92
At-phys	13009	23	190	602	12497	23	190	10	12119	22	178	10	11885	23	190	16721	12009	21	154	2921.54
Ce-all	7613	17	94	93	7491	17	94	3	7382	15	78	4	7295	19	113	149	7306	17	94	254.89
Ce-phys	5437	7	16	56	5268	9	30	1	5215	9	30	1	5184	9	30	34	5184	9	30	80.46
Mm-all	14591	13	69	1253	14265	13	50	15	13791	13	69	16	13591	13	67	2458	13654	13	60	2670.31
Mm-phys	14413	13	69	1198	14078	13	50	15	13636	13	69	15	13428	13	67	2190	13500	11	44	2601.16
Sp-all	50343	63	1268	526	50331	63	1268	214	49514	60	1175	3491	-	-	-	-	48806	51	854	6277.42
Sp-phys	4324	17	96	16	4165	17	96	2	3961	15	71	2	3811	17	96	102	3838	17	96	1069.07

Zbog preglednosti Tabele uvedene su skraćene oznake za nazive algoritama: mc za min-cut without DR, mcDR za min-cut with DR, nDR za neighborhood with DR i cgDR za Column Generation with DR.

Tabela 4.3: Rezultati nad metaboličkim mrežama

instanca	$\|V\|$	$\|E\|$	$\|E_d\|$	$\|V_r\|$	$\|E_r\|$	k_{best}	k_{avg}	$t[s]$	$\|comp\|$	n_c	m_c
Sc	1061	6549	153	1021	6396	5218	5271.8	1374.93	683	27	251
Sa	644	5644	23	631	5621	3263	3277.8	516.8	439	61	1830
Tu	827	5793	46	795	5747	4860	4900.8	1311.62	549	31	317
Ec	537	2844	44	517	2800	2294	2336.2	210.69	376	23	178

4.3.3 Rezultati za metaboličke mreže

U Tabeli 4.3 su prikazani rezultati dobijeni nad metaboličkim mrežama. Tabela je organizovana na isti način kao i Tabela 4.1.

Na osnovu dobijenih rezultata može se primijetiti da je faza redukcije znatno manje relaksirala metaboličke mreže u odnosu na PPI mreže. Iz Tabele 4.1 se može vidjeti da je broj obrisanih grana u fazi redukcije preko 50% u odnosu na ukupan broj grana, za većinu PPI mreža, dok je za metaboličke mreže procenat obrisanih grana maksimalno 2.33%. Razlog za ovo je što u metaboličkim mrežama veliki broj čvorova formira trouglove, pa samim tim nije zadovoljen uslov pravila iz 4.2.2 na osnovu kojeg se vrši brisanje grana.

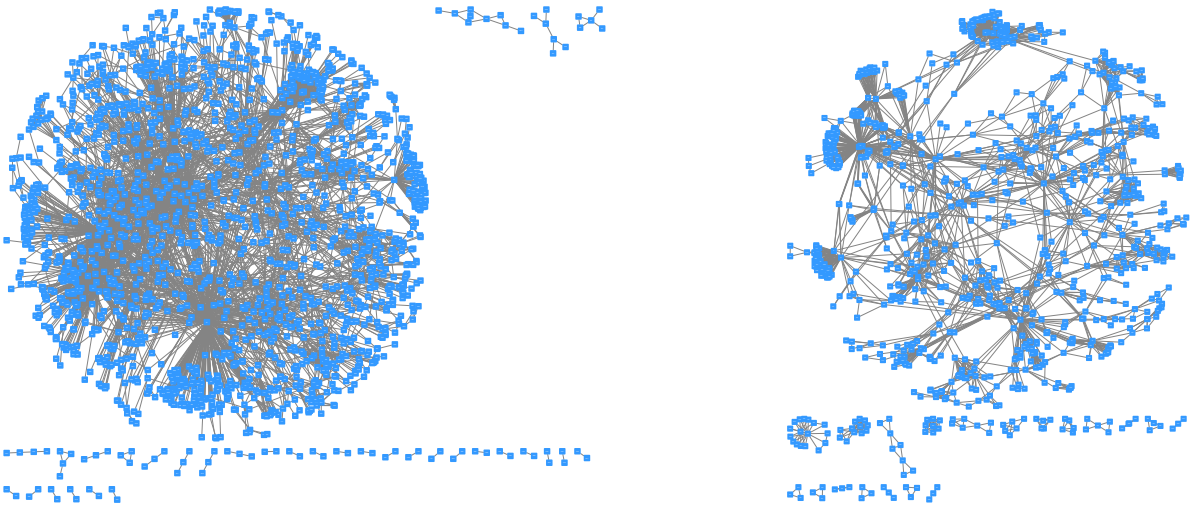
S obzirom da ove instance nisu ranije razmatrane u literaturi, nije moguće poređenje dobijenih rezultata VNS algoritmom sa nekim drugim rezultatima. Dobijene najveće komponente sadrže po nekoliko desetina čvorova, dok najveća među njima (najveća komponenta organizma *Staphylococcus aureus*) je kompletno povezan graf.

4.4 Biološka evaluacija dobijenih visoko povezanih komponenti

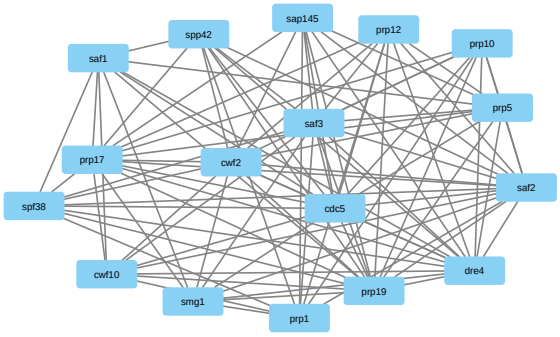
Kao što je već rečeno, velike biološke mreže mogu biti analizirane identifikacijom funkcionalnih podgrupa, poput visoko povezanih komponenti. U ovom Odjeljku će biti detaljnije razmotrene neke visoko povezane komponente PPI mreže *Schizosaccharomyces pombe - phys*. Grafički prikaz ove mreže dat je lijevo na Slici 4.6. Originalna mreža se sastoji od 1963 čvora i 4772 grane. Nakon primjene procedure redukcije grana, dobija se mreža sa 965 čvorova i 2854 grana, koja je prikazana desno na Slici 4.6.

Primjenom VNS algoritma dobija se particija od 598 visoko povezanih komponenti. Najveća od dobijenih visoko povezanih komponenti se sastoji od 17 čvorova i 96 grana, prikazna je na Slici 4.7. U ovoj komponenti proteini su na određeni način povezani sa U4/U6.U5 kompleksom malih nuklearnih RNK (engl. Small nuclear RNA, snRNA), koji čine mala nuklearna U5 RNK, bazama spojene U4/U6 snRNA i preko 30 proteina, u koje spadaju ključne komponente prp8, brr2 i Snu114 [95, 142]. Ovaj kompleks kombinuje supstrat prekursorske tRNK sa U1 i U2 snRNA (prp5, sap145, prp12, prp10) i prevodi ih u katalitički aktivne splajsozome nakon određenih kombinovanih i konformacionih promjena izazvanih relaksacijom na U4 i U6 snRNA [1]. Da bi došlo do sparivanje baza u U4/U6 kompleksu proteini moraju da budu fosforilisani čime se obezbjeđuje normalna funkcija katalaza koje prenose fosforne grupe [92]. Spp42 protein u ovom kompleksu je regulator transkripcije ribozomalnih proteina koji posjeduje hvatače za fosforilisane proteine [60, 127]. Ovaj protein samim tim mora direktno da bude povezan za proteine kinaze saf1, saf2, saf3 (koji su članovi puta degradacije adenin diaminaza, a nivo ovog gena je usko povezan za protein kinazni A put), cwf2, a preko njega i cwf10 proteine

Visoko povezane komponente



Slika 4.6: *Schizosaccharomyces pombe - phys* mreža - početna (lijevo), redukovana (desno). Sa slike desno se vidi da redukcija značajno smanjuje ukupan broj grana.

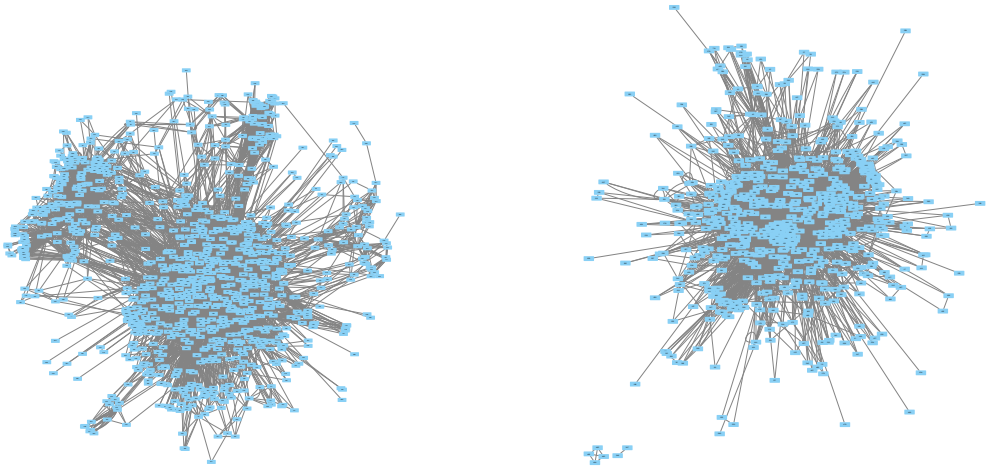


Slika 4.7: Visoko povezana komponenta *Schizosaccharomyces pombe - phys* mreže

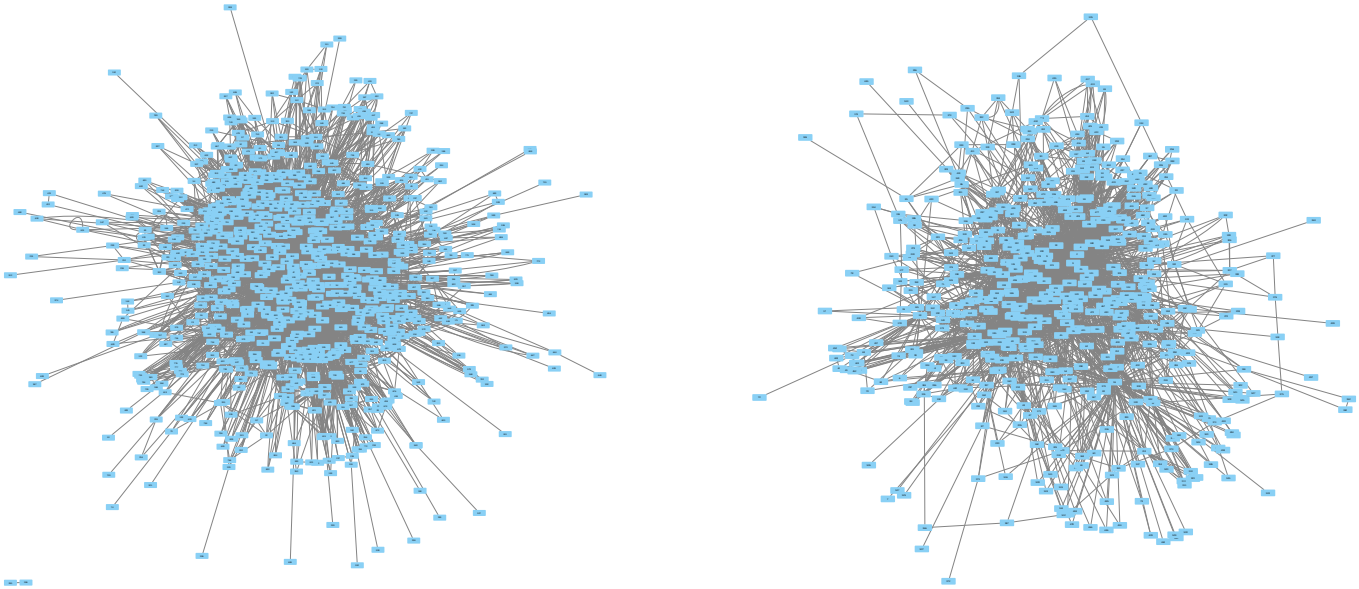
(GTP-aze, komponente U5 snRNA, proteini koji vežu direktno za U5 preko faktora vezivanja spf38 ili cdc8 gena koji takođe kodiraju za GTP-aze) [44, 143, 124]. Za normalnu funkciju U4/U6.U5 u [125] istaknuta je uloga dre4 gena koji direktno reaguje sa prp19 i neophodan je za prethodno prečišćavanje U2, U5 i U6 subjedinica snRNA, pri čemu sprečava višestruko uzastopno sparivanje baza što vodi do mutacija. Pored toga značajno se ističe i uloga smg1 gena koji kodira za protein čija je uloga antioksidans u ovom procesu pri čemu je njegov gen uključen u regulaciju raspadanja tRNK. Datom analizom ove visoko povezane komponente može se zaključiti da su grupisani proteini sa sličnim biološkim funkcijama.

Kompletne metaboličke mreže organizama *Saccharomyces cerevisiae* i *Staphylococcus aureus* su prikazane na Slici 4.8, lijevo, odnosno desno. Grafički prikazi metaboličkih mreža organizama *Tuberculosis -Mycobacterium tuberculosis* i *Escherichia coli* su dati lijevo, odnosno desno na Slici 4.9. Zbog velikog broja čvorova i grana na ovako velikim mrežama je teško uočiti značajna zapažanja i izvesti zaključke koji bi doveli do novih saznanja. U nastavku su prikazane neke visoko povezane komponente dobijene particionisanjem datih metaboličkih mreža i data je analiza procesa koje predstavljaju.

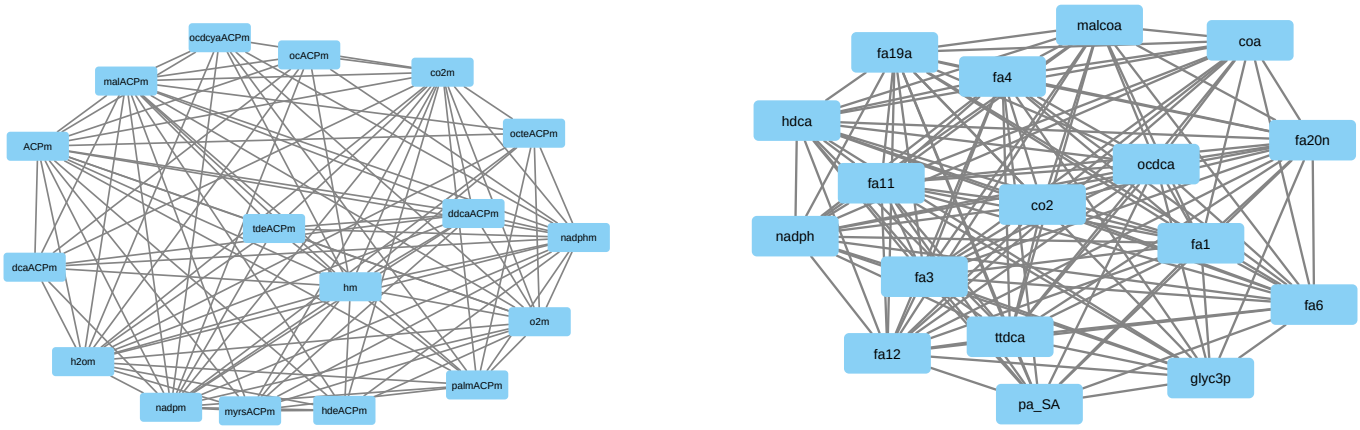
Proces sinteze masnih kiselina se sastoji iz tri uzastopne faze: aktivacija, elogancija i terminacija [8, 149]. U svakoj od faza se ponavljaju neki od intermedijera, pa su u nekim organizmima metaboliti koji učestvuju u ovim fazama gušće, a u nekima rjeđe povezani. Na osnovu metaboličkih mreža, za



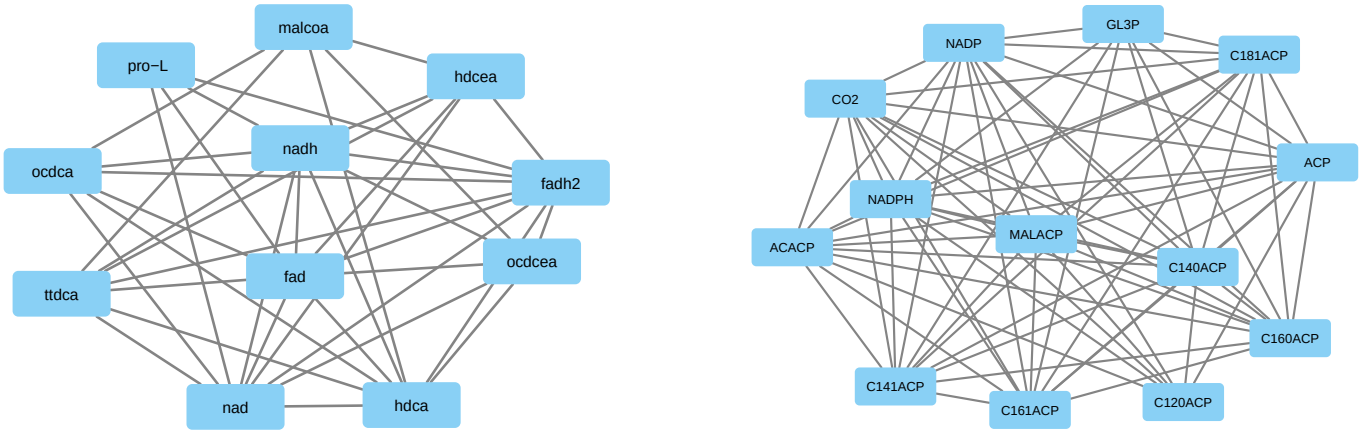
Slika 4.8: Metaboličke mreže organizama *Saccharomyces cerevisiae* (lijevo) i *Staphylococcus aureus* (desno)



Slika 4.9: Metaboličke mreže organizama *Tuberculosis -Mycobacterium tuberculosis* (lijevo) i *Escherichia coli* (desno)



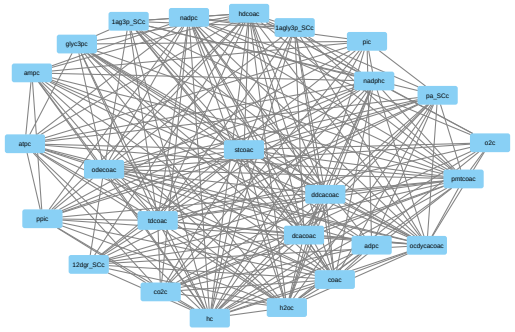
Slika 4.10: Faza terminacije procesa sinteze masnih kiselina u mreži *Saccharomyces cerevisiae*(lijevo) i faza aktivacije i elongacije procesa sinteze masnih kiselina u mreži *Staphylococcus aureus* (desno)



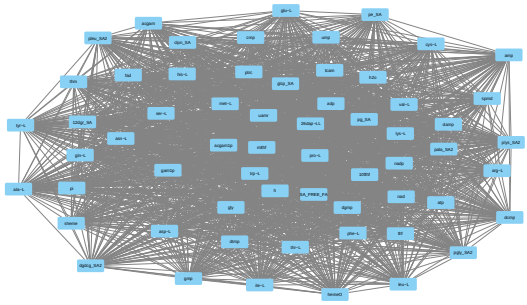
Slika 4.11: Faza aktivacije procesa sinteze masnih kiselina u mreži *Tuberculosis -Mycobacterium tuberculosis*(lijevo) i faza elongacije procesa sinteze masnih kiselina u mreži *Escherichia coli* (desno)

sva četiri organizma se kao visoko povezana komponenta dobija graf koji odgovara nekoj od faza sinteze masnih kiselina. Lijevo na Slici 4.10 je prikazana jedna od visoko povezanih komponenti organizma *Saccharomyces cerevisiae*, koja predstavlja fazu terminacije procesa sinteze masnih kiselina. S druge strane, kao visoko povezana komponenta metaboličke mreže organizma *Staphylococcus aureus* dobijena je struktura koja predstavlja prve dvije faze procesa sinteze masnih kiselina, odnosno fazu aktivacije i elongacije (Slika 4.10 desno). Za razliku od organizma *Staphylococcus aureus*, gdje su intermedijeri prve dvije faze procesa sinteze masnih kiselina povezani u jednu visoko povezanu komonentu, za organizam *Tuberculosis -Mycobacterium tuberculosis* je faza aktivacije izdvojena u posebnu visoko povezanu komponentu (Slika 4.11 lijevo), dok za organizam *Escherichia coli*, jedna od visoko povezanih komponenti predstavlja fazu elongacije procesa sinteze masnih kiselina (Slika 4.11 desno).

U ostalim visoko povezanim komponentama prepoznati su još neki važni metabolički procesi posmatranih organizama. Tako je na primjer, jedna od visoko povezanih komponenti koja je dobijena particionisanjem *Saccharomyces cerevisiae* mreže prikazana na Slici 4.12. Ova komonenta se sastoji od 25 čvorova i 222 grane, a posmatranjem metabolita koji je čine, može se primijetiti da predstavlja proces biosinteze fosfolipida. Najveća visoko povezana komponenta dobijena je za organizam *Staphylococcus aureus* i predstavlja kompletno povezan graf sa 61 čvorom (metabolitom). Grafički prikaz



Slika 4.12: Biosinteza fosfolipida u *Saccharomyces cerevisiae* mreži



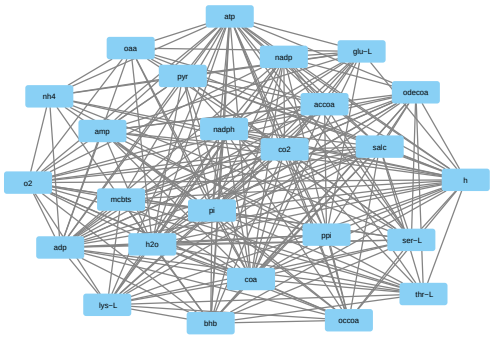
Slika 4.13: Sinteza i konverzija nekih aminokiselina u *Staphylococcus aureus* mreži

te komponente dat je na Slici 4.13, sa koje se može vidjeti da su ovdje grupisani intermedijeri koji učestvuju u procesima sinteze i konverzije nekih aminokiselina. Za razliku od drugih organizama, vidljivo je da ovaj organizam može na više različitih načina izvršiti međusobnu konverziju pojedinih aromatičnih aminokiselina, samim tim i lakšu sintezu proteina. Konverzija aminokiselina je predstavljena i visoko povezanom komponentom organizma *Tuberculosis -Mycobacterium tuberculosis* (Slika 4.14), s tim da je ovaj graf mnogo manje kompleksan i preko piruvata povezuje proces oksidativne fosforilacije i sintezu i konverziju aminokiselina. Oksidativna fosforilacija sa piruvatom kao kofaktorom predstavljena je jednom od visoko povezanih komponenti organizma *Escherichia coli*. Grafički prikaz te komponente dat je na Slici 4.15.

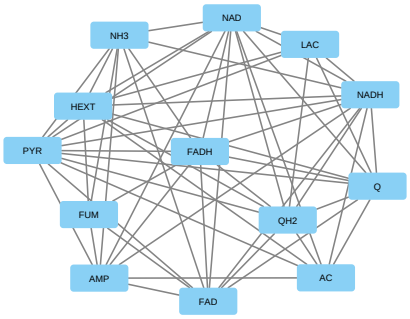
Na osnovu dobijenih visoko povezanih komponenti organizma *Staphylococcus aureus*, moguće je zaključiti da, za razliku od ostalih organizama, ovaj organizam može na više različitih načina sintetisati makromolekule, proteine i lipide, te samim tim favorizovati ove procese u odnosu na ostale. Dakle, dobijene visoko povezane komponente mogu predstavljati dobru polaznu osnovu za poređenje metaboličkih procesa u različitim organizmima.

4.5 Predviđanje GO termina za proteine sa neuređenom strukturom

U ovom odjeljku je predstavljena metoda za predviđanje novih GO anotacija proteina na osnovu analize PPI mreže, pomoću visoko povezanih komponenti. Osnovna ideja ove metode je da se na osnovu identifikacije visoko povezanih komponenti izdvoji spisak GO termina kojima su označeni geni koji odgovaraju proteinima iz visoko povezane komponente. Dalje se proteinima iz komponente, koji ranije nisu bili anotirani datim terminom koji se pojavljuje u spisku izdvojenih GO anotacija, dodjeljuje data oznaka.



Slika 4.14: Konverzija aminokiselina u *Tuberculosis -Mycobacterium tuberculosis* mreži



Slika 4.15: Oksidativna fosforilacija u *Escherichia coli* mreži

U ovom dijelu istraživanja, PPI mreža je formirana na osnovu spiska PPI preuzetih iz Hippie baze http://cbdm-01.zdv.uni-mainz.de/~mschaefer/hippie/ verzija 2.1 datum posljednjeg ažuriranja 18.07.2017). Prvo je izvršena filtracija preuzetih podataka, odnosno sve interakcije koje su autointerakcije ili koje nisu binarne interakcije su izostavljene. Autointerakcije su interakcije proteina samog sa sobom, dok se pod interakcijama koje nisu binarne podrazumijevaju interakcije u kojima učestvuje više od dva proteina. Konačan spisak interakcija sadrži 21326 interakcija u kojima učestvuje 7423 različita proteina. Proteini predstavljaju čvorove mreže, a grana između dva proteina postoji ako između njih postoji interakcija. Dakle, formirana mreža ima 7423 čvora i 21326 grana.

Primjenom VNS algoritma, koji je opisan u Odjeljku 4.2, mreža je particionisana na 446 visoko povezanih komponenti i 5876 singltona. Informacije o kardinalnosti visoko povezanih komponenti su date u Tabeli 4.4. Prva kolona sadrži informaciju o kardinalnosti komponente, dok se u drugoj koloni nalazi podatak koliko komponenti ima datu kardinalnost. Kao što se može vidjeti iz Tabele 4.4, postoji samo jedna komponenta najveće kardinalnosti (12 čvorova), dok su tri komponente kardinalnosti 11. Pored toga, najveći broj visoko povezanih komponenti, njih 360, su trouglovi.

Dobijene visoko povezane komponente su dalje predmet analize obogaćivanja informacijama. Na šest izabranih visoko povezanih komponenti, čije su kardinalnosti u opsegu od 9 do 12, primijenjena su dva alata za obogaćivanje informacijama, DAVID [66] i DinGO [37]. DAVID je veb-alat koji koristi modifikovan Fišerov test za određivanje statističke značajnosti, dok DinGO alat primjenjuje statističke metode BiNGO alata sa poboljšanim performansama. Oba alata kao rezultat vraćaju spisak GO termina kojima su anotirani proteini iz komponente i po kojima su grupisani. U Tabeli 4.5 su predstavljeni neki od rezultata analize. U prvoj koloni se nalazi oznaka visoko povezane komponente, dok je u drugoj koloni sadržana informacija o kardinalnosti visoko povezane komponente. Treća i četvrta kolona sadrže neke od zajedničkih identifikatora i termina za datu komponentu, koji su rezultat analize DAVID i DinGO alatima, respektivno. U posljednje dvije kolone prikazane su

Tabela 4.4: Kardinalnost visoko povezanih komponenti Hippie PPI mreže

Kardinalnost komponente	Broj komponenti
12	1
11	3
10	1
9	4
7	7
6	3
5	41
4	26
3	360

Tabela 4.5: Rezultati analize visoko povezanih komponenti DAVID i DinGO alatima

Komponenta	$\|V\|$	Identifikator	GO termin	DAVID: p-vrijednost	DinGO: p-vrijednost
component_13	12	GO:0034719	SMN-Sm protein complex	1.117E-25	1.963E-25
component_73	11	GO:0005669	transcription factor TFIID complex	4.119E-27	3.810E-16
component_15	11	GO:0004407	histone deacetylase activity	2.921E-16	6.700E-13
component_273	11	GO:0000178	exosome (RNase complex)	2.626E-27	5.389E-25
component_86	10	GO:0000715	nucleotide-excision repair, DNA damage recognition	6.022E-18	9.992E-21
component_136	9	GO:0003688	DNA replication origin binding	4.831E-18	7.067E-23

p-vrijednost, za date termine, koje su dobijene DAVID, odnosno DinGO alatom, respektivno.

S obzirom na jako veliki broj čvorova koji su nakon particionisanja ostali kao singltoni, razmatrana je ideja proširenja dobijenih visoko povezanih komponenti. Cilj ovog postupka je da se u proširenje dodaju proteini koji intereaguju sa proteinima iz visoko povezane komponente, pa su dobijene komponente proširene na sljedeća tri načina:

- Proširenje 1: svakoj visoko povezanoj komponenti su dodati proteini koji interaguju (u polaznoj mreži) sa bar jednim proteinom iz visoko povezane komponente;
- Proširenje 2: svakoj visoko povezanoj komponenti su dodati proteini koji interaguju (u polaznoj mreži) sa bar dva proteina iz visoko povezane komponente;
- Proširenje 3: svakoj visoko povezanoj komponenti su dodati proteini koji interaguju (u polaznoj mreži) sa bar tri proteina iz visoko povezane komponente.

Komponente proširene na opisani način su dalje predmet analize obogaćivanja informacijama DinGO alatom. DinGO alat kao rezultat vraća spisak GO termina koji su karakteristični za proteine iz visoko povezane komponente. Kao rezultat, lista GO termina će biti dodijeljena proteinima koji ranije nisu imali tu anotaciju, a nalaze se u istoj komponenti kojoj pripadaju proteini koji imaju datu anotaciju.

Izbor parametara

Testiranje performansi predložene metode za predviđanje GO termina, koja se zasniva na analizi PPI mreže i analizi obogaćivanja informacijama, je izvršeno na skupu podataka koji potiču iz

Tabela 4.6: Informacije o CAFA3 skupu podataka

Skup podataka	GO	$\|G\|$	$\|G_{Hippie}\|$
mfo_HUMAN_type2	MF	93	44
bpo_HUMAN_type2	BP	163	88
cco_HUMAN_type2	CC	68	42

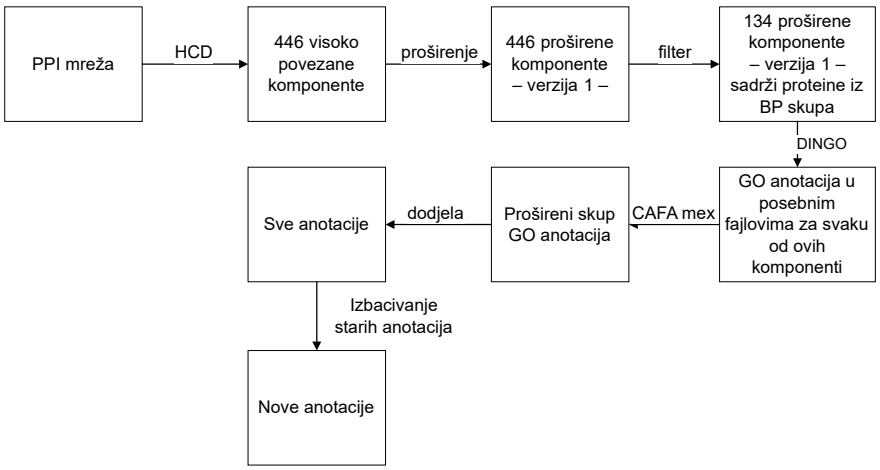
Tabela 4.7: Podaci o komponentama koje sadrže proteine iz CAFA3 skupa

Proširenje	GO	Broj komponenti	Broj proteina iz CAFA3 skupa
Version_1	CC	62	31
Version_2	CC	9	6
Version_3	CC	2	2
Version_1	BP	134	70
Version_2	BP	23	18
Version_3	BP	14	14
Version_1	MF	54	28

poznatog CAFA takimičenja (engl. Critical Assessment of protein Function Annotation algorithms - CAFA Challenge) [167]. Namjena CAFA takmičenja je procjena tačnosti računarskih metoda i algoritama koji predviđaju funkciju proteina. Za poređenje predloženih računarskih metoda koristi se referentni CAFA skup podataka (engl. CAFA benchmark dataset), koji se formira nekoliko mjeseci nakon završetka takmičenja, na osnovu rezultata eksperimenata, kojima se u međuvremenu odredi tačna funkcija proteina.

U ovom istraživanju su korišteni skupovi podataka CAFA3 takmičenja, koji su tipa 2, odnosno skupovi koji sadrže proteine koji već imaju neke GO oznake. Pregled korištenih podataka dat je u Tabeli 4.6. Prva kolona sadrži naziv skupa podataka, a druga oznaku ontologije. U trećoj koloni je prikazan broj proteina koji pripadaju tom skupu, a u četvrtoj koloni informacija koliko tih proteina se nalazi u Hippie bazi. Od svih proširenih komponenti izabrane su samo one koje sadrže proteine iz CAFA3 skupa. Kao što se može vidjeti iz Tabele 4.7, u odnosu na CC ontologiju, 62 komponente koje su proširene verzijom proširenja 1 sadrže proteine iz CAFA3 skupa i to ne sve proteine, već njih 31/42. Samo šest proteina iz ove ontologije je sadržano u 9 komponenti proširenih verzijom proširenja 2. Dok se u komponentama proširenim trećom verzijom proširenja nalaze samo 2 proteina iz CAFA3 skupa i to u 2 komponente.

Na izabrane komponente je primijenjen DinGO alat sa različitim vrijednostima parametra za odsijecanje (engl. cutoff parametar), odnosno za vrijednosti parametra iz skupa $\{0.002, 0.02, 0.03, 0.05\}$. U cilju pridruživanja svih GO anotacija urađena je propagacija CAFA *mex* softverom, koji je razvijen u Labaratoriji za bioinformatiku i računarsku hemiju, Instituta za nuklearne nauke, Vinča. Pod propagacijom se podrazumijeva proširenje skupa termina, kojima je protein prvobitno anotiran, svim roditeljskim terminima iz ontologije do korjena. U tom cilju korišten je GO obo fajl, koji predstavlja tekstualni zapis genske ontologije, iz 2016. godine (data-version: releases/2016-05-31). Lako se može zaključiti da će na opisan način broj dodijeljenih GO termina biti jako veliki, pa je iz tog razloga variran maksimalan broj dobijenih GO anotacija koji je dalje korišten za pridruživanje elementima određene komponente. Broj maksimalnih GO anotacija uzima vrijednosti iz skupa $\{30, 50, 100\}$. U posljednjem koraku ove procedure iz spiska dodjeljenih anotacija izbačene su anotacije koje su postojale ranije i za to je korišten anotacijski fajl iz 2016. godine (dostupan na http://release.geneontology.org/). Anotacijski fajl sadrži spisak svih termina kojima su do tog



Slika 4.16: Kompletna procedura predviđanja GO termina na osnovu visoko povezanih komponenti i analize obogaćivanja

trenutka proteini anotirani.

Kompletna procedura za verziju proširenja 1 i BP ontologiju je prikazana dijagramom na Slici 4.16. Na početnu PPI mrežu se primjenjuje HCD algoritam, čime je dobijeno 446 visoko povezanih komponenti. Nakon toga, vrši se proširenje tih dobijenih komponenti, verzijom proširenja 1. Zatim se izdvajaju samo proširene komponente koje sadrže proteine iz BP ontologije i takvih u ovom slučaju ima 134. Na svaku od ovih 134 komponenti se primjenjuje DinGO alat i za svaku komponentu se dobija posebna datoteka sa spiskom GO termina, koji su već postojeći termini proteina koji čine tu komponentu. Za svaki spisak GO termina, CAFA *mex* softverom se vrši propagacija datog skupa termina i time se dobija proširen skup GO termina. Svakom proteinu iz date komponente se dodjeljuju svi termini iz proširenog skupa. U posljednjem koraku ove procedure se iz spiska dodijeljenih termina izbacuju već postojeći termini, te se tako dobijaju nove anotacije. Za ostale kombinacije verzija proširenja i ontologija redoslijed koraka je isti, mijenjaju se samo skupovi podataka koji se koriste.

Dobijeni rezultati su evaluirani softverom CAFA *eval*, koji je kao i CAFA *mex* softver razvijen u istoj labaratoriji, uz upotrebu utvrđenog, polaznog CAFA referentnog skupa podataka https://www.biofunctionprediction.org/cafa/. Za evaluacionu mjeru je izabrana *F1* mjera, čija vrijednost se računa po formuli

$$F1 = \frac{2 * precision * recall}{precision + recall},$$

gdje se preciznost (engl. precision) definiše kao odnos ukupnog broja pogođenih anotacija i ukupnog broja anotacija u modelu, a odziv (engl. recall) kao odnos ukupnog broja pogođenih anotacija i ukupnog broja anotacija u testnom skupu podataka. Informacije o vrijednosti *F1* mjere za sve kombinacije parametara za izabrane komponente, koje sadrže proteine iz CC ontologije, su prikazane u Tabeli 4.8. Tabela 4.8 je organizovana na sljedeći način. Prva kolona sadrži informaciju o verziji proširenja visoko povezanih komponenti, a u drugoj koloni se nalazi vrijednost DinGO parametra za odsijecanje. U trećoj koloni je prikazan maksimalan broj GO termina koji se pridružuje, dok posljednja kolona sadrži vrijednost *F1* mjere. S obzirom da verzija proširenja 3 sadrži veoma mali broj ciljnih proteina iz CC ontologije, te komponente nisu dalje razmatrane i zato se u Tabeli 4.8 ne nalaze informacije o njima. Tabela 4.9 sadrži navedene informacije za izabrane komponente koje sadrže proteine iz BP ontologije i organizovana je na isti način kao i Tabela 4.8. Najbolji dobijeni

rezultati su označeni.

Tabela 4.8: Vrijednosti *F1* mjere za CC skup podataka

Verzija proširenja	DINGO cutoff	Maks. broj GO termina	*F1* mjera
1	**0.002**	**30**	**0.368317**
1	0.002	50	0.27214
1	0.002	100	0.206094
1	0.02	30	0.366124
1	0.02	50	0.257384
1	0.02	100	0.160902
1	0.03	30	0.366124
1	0.03	50	0.256538
1	0.03	100	0.154138
1	0.05	30	0.365996
1	0.05	50	0.256516
1	0.05	100	0.14725
2	0.002	30	0.265896
2	0.002	50	0.221154
2	0.002	100	0.184739
2	0.02	30	0.256158
2	0.02	50	0.221277
2	0.02	100	0.16
2	0.03	30	0.269231
2	0.03	50	0.221344
2	0.03	100	0.162791
2	0.05	30	0.323144
2	0.05	50	0.268116
2	0.05	100	0.197861

Pri analizi komponenti koje sadrže proteine iz MF ontologije su razmatrane samo sljedeće kombinacije parametara: verzija proširenja 1, DinGO parametar za odsijecanje 0.002 i maksimalan broj GO termina, koji se uzimaju u obzir nakon propagacije, uzima vrijednost iz skupa $\{30, 40, 50, 100\}$. Dobijeni rezultati su prikazani u Tabeli 4.10, koja je organizovana kao i prethodne.

Posmatrajući sve prikazane rezultate u Tabelama 4.8–4.10 može se primijetiti da se najbolji rezultati koji su dobijeni predloženom metodom kreću u okviru prosječnih vrijednosti rezultata dobijenih drugim metodama za podatke ovog tipa [27]. Posmatrajući dobijene rezultate u Tabeli 4.8 uočava se da su najbolji rezultati dobijeni ako se u obzir uzima maksimalno 30 GO termina. Takođe, upoređivanjem rezultata algoritama koji dodjeljuju maksimalno 50 i 100 novih anotacija, zaključuje se da su rezultati bolji ako je broj dodijeljenih termina manji. S druge strane, posmatrajući rezultate dobijene za BP ontologiju (Tabela 4.9) vidi se da ako je verzija proširenja 1, onda su najbolji rezultati dobijeni ako se u obzir uzima maksimalno 50 GO termina. Za verzije proširenja 2 i 3, bolji rezultati za ovu ontologiju su dobijeni uzimanjem u obzir maksimalno 100 GO termina. Za MF ontologiju (Tabela 4.10) su dobijeni nešto lošiji rezultati nego za druge dvije ontologije, ali slično kao i za CC ontologiju, bolji rezultati se dobijaju što je parametar koji određuje maksimalan broj GO termina manji. Na osnovu prethodne analize dolazi se do zaključka da se najbolji rezultati dobijaju za sljedeće kombinacije parametara:

- Verzija proširenja: 1;

Tabela 4.9: Vrijednosti *F1* mjere za BP skup podataka

Verzija proširenja	DINGO cutoff	Maks. broj GO termina	*F1* mjera
1	0.002	30	0.238731
1	0.002	50	0.269256
1	0.002	100	0.244866
1	0.02	30	0.240835
1	**0.02**	**50**	**0.269734**
1	0.02	100	0.246532
1	0.03	30	0.240776
1	0.03	50	0.267471
1	0.03	100	0.243928
1	0.05	30	0.240776
1	0.05	50	0.267471
1	0.05	100	0.243685
2	0.002	30	0.0831974
2	0.002	50	0.159401
2	0.002	100	0.205807
2	0.02	30	0.125191
2	0.02	50	0.186951
2	0.02	100	0.240479
2	0.03	30	0.12614
2	0.03	50	0.193069
2	0.03	100	0.243781
2	0.05	30	0.12614
2	0.05	50	0.199029
2	0.05	100	0.2416
3	0.002	30	0.0471698
3	0.002	50	0.0477742
3	0.002	100	0.131267
3	0.02	30	0.0688889
3	0.02	50	0.101791
3	0.02	100	0.148611
3	0.03	30	0.0706402
3	0.03	50	0.103093
3	0.03	100	0.197516
3	0.05	30	0.0706402
3	0.05	50	0.11465
3	0.05	100	0.20341

- DINGO parametar za odsijecanje: 0.002 (za MF i CC ontologije), 0.02 (za BP ontologiju);

- Maksimalan broj GO termina: 30 (za MF i CC ontologije), 50 (za BP ontologiju).

Izabrani parametri su dalje korišteni za primjenu predložene metode na proteine neuređenje strukture (engl. Intrinsically disordered proteins (IDP)).

Tabela 4.10: Vrijednosti *F1* mjere za MF skup podataka

Verzija proširenja	DINGO cutoff	Maks. broj GO termina	*F1* mjera
1	**0.002**	**30**	**0.199122**
1	0.002	40	0.172121
1	0.002	50	0.154258
1	0.002	100	0.117505

Tabela 4.11: Broj novih anotacija za IDPs

Ontologija	Broj gena	Broj novih GO termina
MF	96	3934
BP	97	10988
CC	95	4910

Primjena na IDP

Poznato je da su IDP uključeni u ključne ćelijske funkcije, uključujući regulaciju transkripcije, translaciju i ćelijski ciklus [152]. IDP nemaju stabilne sekundarne ili tercijarne strukture u nekoliko regiona ili duž cijele sekvence. Zbog svoje lake savitljivosti, IDP često djeluju kao čvorišta u mrežama proteinskih interakcija gdje imaju centralnu ulogu u regulaciji signalnih puteva. Upravo zbog te činjenice, osnovna ideja ovog odjeljka je da se prethodno opisani postupak primijeni na IDP, pa da se na osnovu informacija iz PPI mreže identifikuju nove GO oznake za IDP i proteine koji sa njima interaguju.

Za ovaj dio istraživanja korišten je skup inherentno neuređenih proteina čovjeka, preuzet iz DisProt baze podataka http://www.disprot.org/. Kompletan skup sadrži 229 proteina, od kojih je 109 sadržano u mreži formiranoj na osnovu PPI preuzetih iz Hippie baze podataka. Nakon particionisanja mreže, 60/109 proteina iz DisProt baze podataka su singltoni dok se ostatak 49/109 pojavljuje u nekoj visoko povezanoj komponenti. Nakon proširenja visoko povezanih komponenti verzijom proširenja 1, još 48/60 proteina se pojavi u nekoj od proširenih visoko povezanih komponenti. Konačno, 97/109 proteina iz DisProt baze podataka se nalazi u nekoj proširenoj visoko povezanoj komponenti mreže iz Hippie baze podataka. Na proširene visoko povezane komponente koje sadrže IDP, primjenjena je procedura sa Slike 4.16 sa vrijednostima parametara za koje su dobijeni najbolji rezultati.

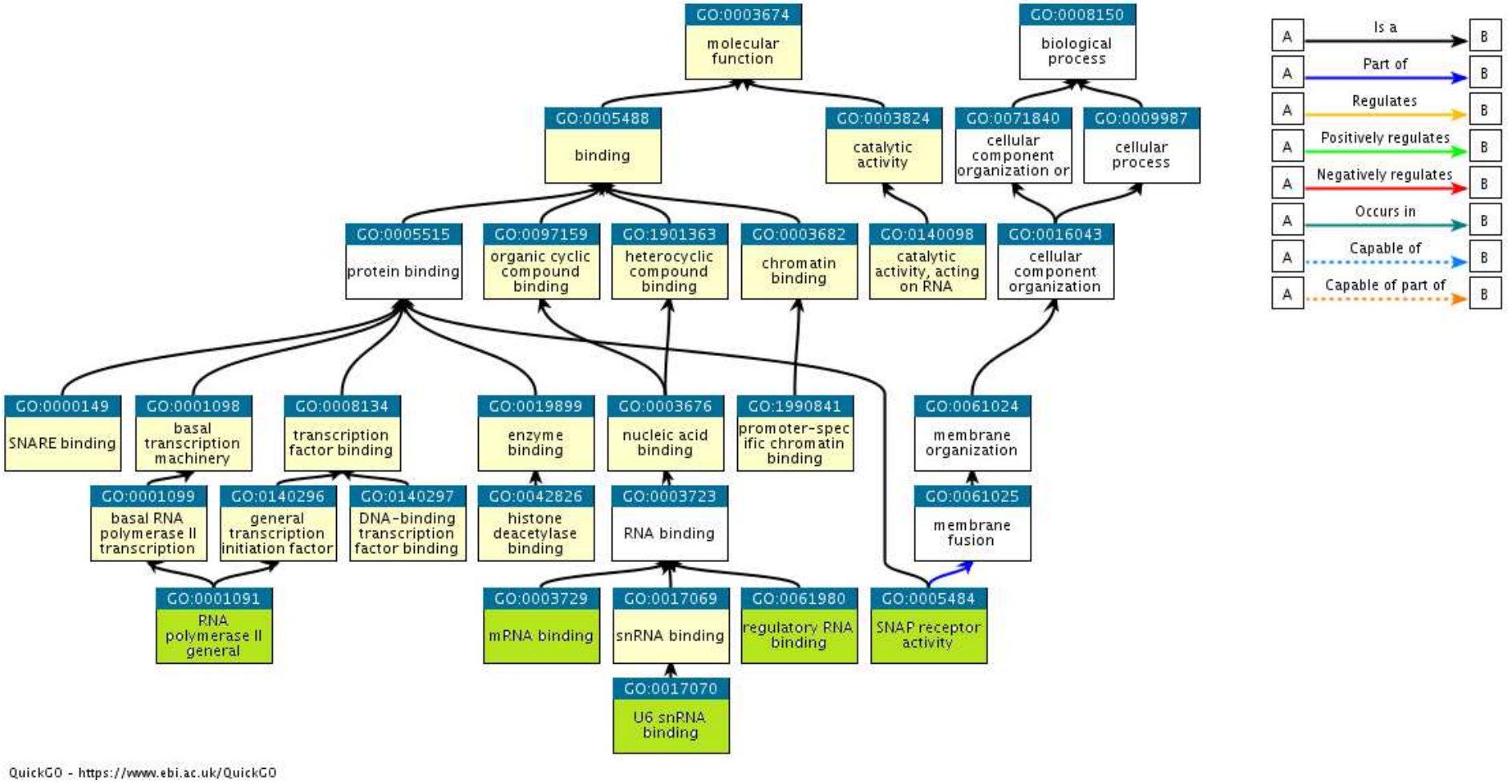
Za navedenu analizu korištena je genska ontologija (data-version: releases/2019-02-14) i anotacijski fajl iz 2019. godine (dostupan na http://release.geneontology.org/). Nakon primjene opisanog postupka dobija se veliki broj GO termina (reda veličine nekoliko hiljada) koji su pridruženi IDP. Preciznije informacije se nalaze u Tabeli 4.11. Veliki broj novih termina je posljedica toga što se vrši propagacija GO termina po ontologiji do korijena i što proširene visoko povezane komponente nisu disjunktne.

Zbog ilustracije dobijenih rezultata, detaljnije su posmatrani novi GO termini koji su dodjeljeni genu AGO2. AGO2 gen (DP00736) kodira za protein iz *Argonaute* familije proteina koji su uključeni u RNK interferenciju (engl. RNA interference) i predstavlja ključni dio RISC kompleksa (engl. RNA-induced silencing complex (RISC)). AGO2 sadrži dva neuređena regiona na N- i na C-kraju. Na Slici 4.17 su prikazani novi termini koji su dodijeljeni AGO2 u MF ontologiji. Kao što se može vidjeti, samo 5 od 29 novih termina za AGO2 u MF se nalazi u listovima. Na Slikama 4.18 i 4.19 su predstavljeni novi termini za AGO2 u CC i BP ontologijama, respektivno. Za CC ontologiju 6 od 35 novih termina se nalazi u listovima, dok je za BP ontologiju taj odnos 4/64.

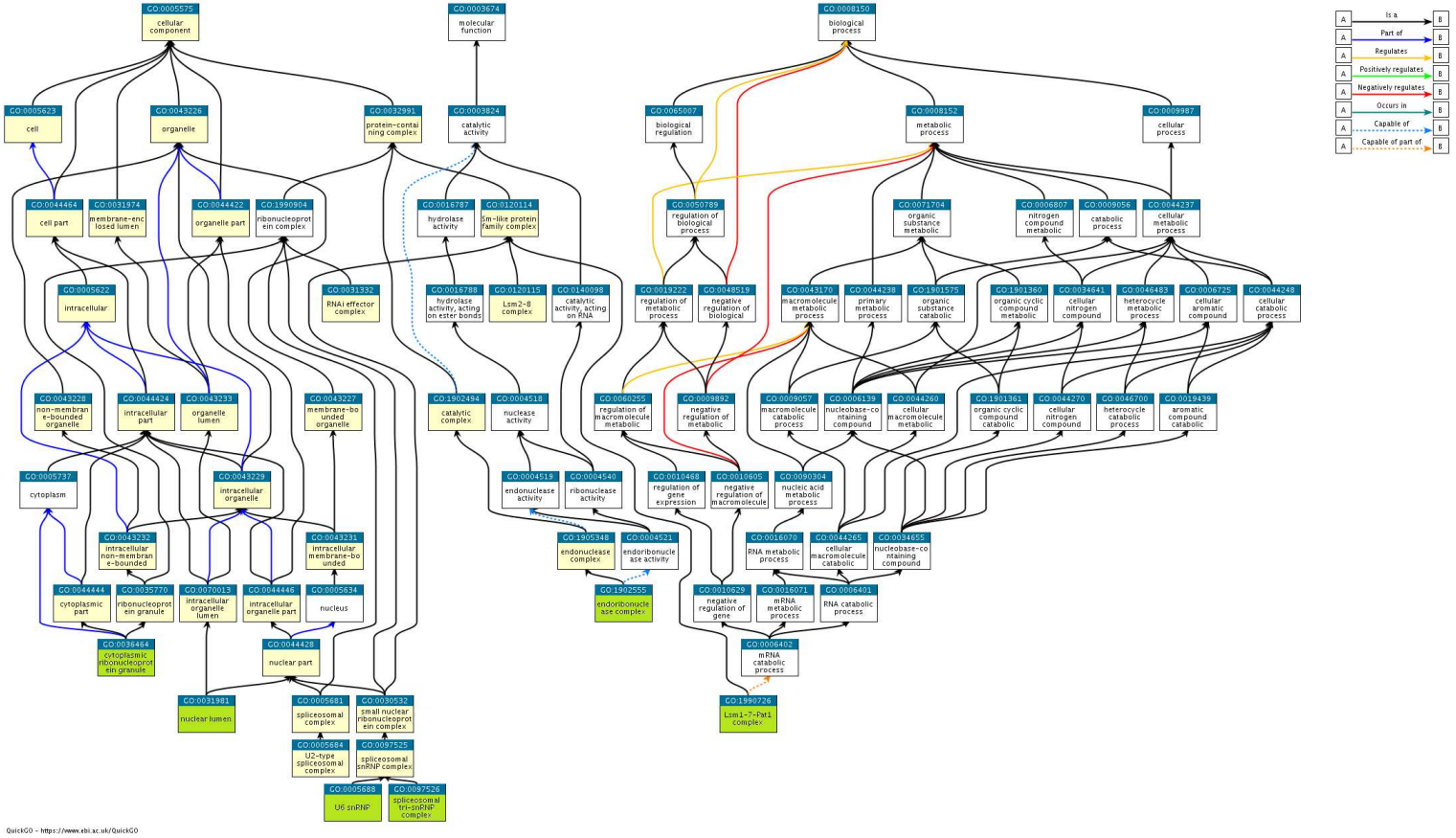
4.6 Završna razmatranja

U ovom poglavlju je razmatran NP težak problem particionisanja mreže na visoko povezane komponente. U literaturi je poznat egzaktan algoritam za rješavanje datog problema, ali za instance velikih dimenzija egzaktan algoritam ne pronalazi rješenje u razumnom vremenu, te je zato opravdan pristup razvijanja heurističkih metoda. Ovdje je predstavljen algoritam zasnovan na metoda promjenljivih okolina. Primjenom na PPI mreže, koje su i ranije razmatrane u literaturi, pokazano je da predloženi VNS algoritam pronalazi kvalitetna rješenja, koja su bliža optimalnim rješenjima nego rješenja dobijena drugim poznatim heurističkim metodama. Pored navedenih PPI mreža, algoritam je po prvi put primijenjen i na metaboličke mreže različitih organizama. Dobijene visoko povezane komponente metaboličkih mreža predstavljaju dobru osnovu za upoređivanje metaboličkih procesa u različitim organizmima. U posljednjem odjeljku ovog poglavlja je predstavljena metoda za predviđanje novih GO anotacija proteina na osnovu informacija dobijenih iz visoko povezanih komponenti PPI mreže.

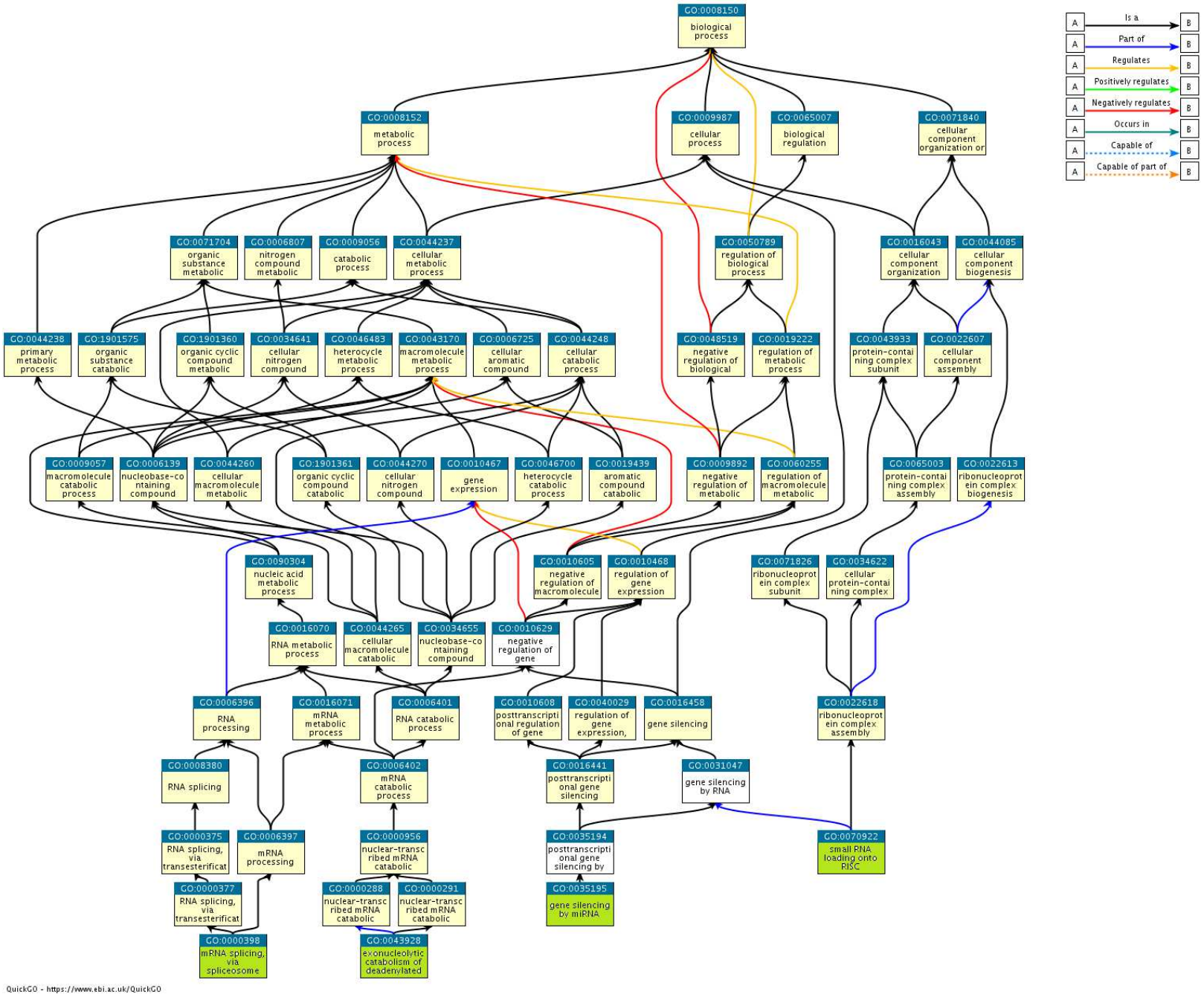
Brojni su pravci u kojima se mogu vršiti unapređenja u vezi problema razmatranih u ovom poglavlju. Hibridizacijom predloženog VNS algoritma sa nekom drugom metodom bi se eventualno omogućilo dobijanje kvalitetnijih rješenja koja su optimalna ili još bliža optimalnim rješenjima. Predložena metoda za predviđanje novih GO anotacija može dodatno da se poboljša, prije svega to poboljšanje treba da ide u pravcu smanjenja broja novih anotacija. Jedan od mogućih pristupa bi mogao da bude izbor druge polazne PPI mreže, čijim particionisanjem bi se dobilo više visoko povezanih komponenti, a manje singltona. Druga varijanta poboljšanja bi mogla da bude razmatranje "užih" varijanti proširenja, odnosno nešto striktniji izbor proteina koji čine proširenu visoko povezanu komponentu. Takođe, variranje predloženih parametara DinGO alata i/ili maksimalnog broja termina koji će se koristi nakon propagacije po genskoj ontologiji može da dovede do boljih rezultata.



Slika 4.17: Grafički prikaz novih termina za gen AGO2 u MF ontologiji, prikazan QuickGO (https://www.ebi.ac.uk/QuickGO/) alatom



Slika 4.18: Grafički prikaz novih termina za gen AGO2 u CC ontologiji, prikazan QuickGO (https://www.ebi.ac.uk/QuickGO/) alatom



Slika 4.19: Grafički prikaz novih termina za gen AGO2 u BP ontologiji, prikazan QuickGO (https://www.ebi.ac.uk/QuickGO/) alatom

Glava 5

Identifikacija značajnih grupa proteina dodavanjem novih grana u težinsku PPI mrežu

5.1 Uvod

Proteini su odgovorni za većinu važnih funkcija u ćeliji. U većini slučajeva, ćelijski procesi nisu regulisani samo jednim proteinom već čitavom grupom proteina koji ulaze u međusobne interakcije. Proteini mogu formirati proteinske komplekse, koje čine bar dva proteina u stabilnoj, dugotrajnoj interakciji, a dobar primjer proteinskih kompleksa su ribozomi i splajsozomi. S druge strane, najveći broj interakcija između proteina su nepostojane i one leže u osnovi ćelijske regulacije, signalne transdukcije, itd. U nekoliko istraživanja je pokazano da se u mrežama, koje reprezentuju biološke molekulske interakcije, mogu razlikovati proteinski kompleksi - grupe proteina koje formiraju stabilne proteinske komplekse i u samim biološkim sistemima, i funkcionalni moduli, koji sadrže proteine čije su interakcije nepostojane i dešavaju se na različitim mjestima i u različitom trenutku određenog ćelijskog procesa [91, 140].

U mnogim radovima (na primjer u [140, 49, 63]) je primijećeno da otkrivanje proteinskih kompleksa i funkcionalnih modula računarskim metodama ima visoku statističku značajnost i konzistentnu funkcionalnu anotaciju, koja se dobro poklapa sa eksperimentalno dobijenim grupama proteina. Prema tome, određivanje proteinskih kompleksa i funkcionalnih modula određenog organizma je važno zbog boljeg razumijevanja principa ćelijske organizacije. Sa druge strane, pojedinačne ćelije organizma sadrže hiljade proteina pa je broj potencijalnih grupa proteina koje imaju značajnost veoma veliki. Da bi se provjerile PPI u kompleksima koje su predviđene računarskim metodama, potrebno je imati referentan skup podataka koji sadrži validne interakcije (pozitivni slučajevi) i neintereagujuće (negativne) slučajeve. Takvi skupovi podataka su poznati pod nazivom zlatni standardi (engl. Gold standards). Pomenuti skupovi se koriste za formiranje modela predviđanja i za evaluaciju [22], tj. kao osnova za zaključivanje i validaciju predviđenih PPI. Zlatni standardi su obično rezultat sistemskih istraživanja. Za organizam kvasca (engl. yeast), čiji proteinski kompleksi su razmatrani u ovom poglavlju, se obično koristi MIPS (Munich Information Center of Protein Sequences database) standard [103] i CYC2008 standard [121]. CYC2008 standard je obiman katalog od 408 heteromernih proteinskih kompleksa koji su potvrđeni eksperimentima objavljenim u literaturi.

Proteini koji pripadaju istom kompleksu ili funkcionalnom modulu obično fizički intereaguju. Stoga je pristup za identifikaciju značajnih grupa proteina razmatranjem gustih regiona u PPI

mrežama opravdan [166]. Preciznije, te grupe se posmatraju kao gusto povezane podmreže [88]. Međutim, s obzirom na veliku količinu šuma, odnosno, lažno pozitivnih i lažno negativnih interakcija, metode koji identifikuju grupe proteina (poput kompleksa) na osnovu PPI mreže imaju ograničenu tačnost [91, 41, 165, 30]. Novija istraživanja ukazuju na to da značajne grupe proteina mogu biti vrlo rijetki regioni ili čak i nepovezani podgrafovi u mreži [109]. Dakle, opravdana je pretpostavka da PPI mreže nisu potpune, odnosno da još uvijek nedostaju mnoge interakcije između proteina koji čine značajne proteinske grupe.

Cilj istraživanja predstavljenog u ovom poglavlju je ispitivanje da li se na osnovu PPI mreže mogu identifikovati značajne grupe proteina koji su u toj PPI mreži rijetko povezani.

U nauci se integracija podataka iz više različitih izvora već pokazala kao uspješan pristup za dobijanje novih bioloških informacija o određenoj strukturi (na primjer [50, 155, 146]). Pored toga, istraživanja PPI mreža takođe ukazuju na to da metode bazirane samo na topološkim osobinama mreže ne koriste dovoljno informacija o važnim proteinskim grupama [41]. Dakle, kombinovanje informacija dobijenih iz topoloških osobina PPI mreže sa dodatnim biološkim informacijama dobijenim iz drugih izvora, bi moglo poboljšati pretragu za proteinima koji čine značajne grupe, a koji su u mreži slabo povezani.

U ovom poglavlju predstavljena je ideja integracije podataka iz nekoliko izvora. Osnovni izvor podataka su baze PPI mreža i zlatni standardi proteinskih kompleksa, a dodatne informacije o vezama između proteina su dobijene na osnovu podataka iz baza o genskoj ko-ekspresiji. Kao polazna tačka uzima se grupa slabo povezanih proteina koji pripadaju različitim kompleksima. Na početku pristup je sličan pristupu iz [109], odnosno rješava se problem dodavanja novih interakcija u netežinsku PPI mrežu, tako da podgrafovi indukovani proteinima koji čine jedan kompleks budu povezani. Poznato je da je navedeni problem u opštem slučaju NP težak, pa bi primjena približnih algoritama mogla biti od koristi za rješavanje ovog problema na velikim instancama. Ovdje su razmatrane dvije varijante ovog problema, netežinska i težinska, i za obje je predložen algoritam koji se zasniva na metodi promjenljivih okolina (VNS). U težinskoj varijanti problema, pri izboru interakcije koja će biti dodata veći prioritet se daje interakcijama između proteina sa većom genskom ko-ekspresijom. Vrijednost geneske ko-ekspresije između dva proteina je određena pomoću SPELL alata [61]. SPELL alat je veb baziran pretraživač zasnovan na kontekstu velike kolekcije informacija o organizmu *S. cerevisiae*. Metodologija za određivanje vrijednosti genske ko-ekspresije za parove proteina je opisana u Odjeljku 5.4.3.

U cilju formiranja biološki značajnih grupa proteina, u sljedećoj fazi predložene metode uključuju se dodatni proteini. Specifičnom strategijom, koja je opisana u Odjeljku 5.3.3, dodaju se proteini koji indirektno intereaguju sa proteinima iz malih polaznih grupa. Tako proširene grupe proteina, koje sadrže nekoliko stotina proteina, su dalje predmet analize u alatima za obogaćivanje informacijama, koji vrše procjenu statističke značajnosti. Rezultati dobijeni nad PPI mrežama, koji su prikazani u Odjeljku 5.4, pokazuju da većina takvih grupa proteina ima visoke skorove značajnosti. Tako dobijene grupe proteina su dalje razmatrane sa biološke tačke gledišta.

5.2 Raniji rezultati

Problem podržavanja poznatih proteinskih kompleksa u PPI mreži tako da svaki proteinski kompleks bude povezan definisan je u [109]. Za dati skup proteinskih kompleksa i PPI mrežu potrebno je dodati što je moguće manje grana tako da svaki kompleks bude povezan. Problem je označen

sa MinPPI. Specijalni slučaj ovog problema je MinPPI0 gdje je skup grana u polaznoj PPI mreži prazan.

Za rješavanje MinPPI problema u pomenutom radu je predložen model cjelobrojnog linearnog programiranja nazvan ILPMinPPI, kao i pohlepni heuristički pristup nazvan GreedyMinPPI. GreedyMinPPI je baziran na pohlepnom algoritmu iz [7], primijenjen je na nekoliko skupova proteinskih kompleksa i PPI mreža i vrši procjenu broja dodatnih PPI. Dobijena rješenja su posmatrana kao poboljšanje tačnosti postojećih metoda za predviđanje PPI.

MinPPI problem je u bliskoj vezi sa opštijim problemom rekonstrukcije mreže sa ograničenjem da podgrafovi trebaju biti povezani (engl. Network construction problem), koji je uveden u [74]. U problemu rekonstrukcije mreže dat je graf G sa skupom čvorova V i skupom grana E, te je svakoj grani pridružena cijena dodavanja. Pored toga dat je skup ograničenja $S = \{S_1, S_2, ..., S_r\}$, gdje je svaki S_i podskup skupa V. Cilj problema rekonstrukcije mreže je formiranje skupa grana E' između čvorova skupa V, tako da, za svako i, skup S_i indukuje povezan podgraf skupa $G = (V, E \cup E')$ i da je suma cijena svih grana iz E' minimalna. Sličnost između MinPPI problema i problema rekonstrukcije mreže je očigledna. Proteini iz mreže odgovaraju čvorovima, interakcije predstavljaju grane, težina svake PPI odgovara cijeni dodavanja grane, a proteinski kompleksi su elementi skupa S.

MinPPI problem je ekvivalentan problemu pronalaženja minimalnog preklapanja povezanih tema (engl. Minimum Topic-Connected Overlay), koji je definisan u [28]. U navedenom problemu dat je skup od t tema i skup od n korisnika. Za svaku temu je poznat skup korisnika koji su zainteresovani za nju. Potrebno je povezati korisnike u mrežu minimalnim brojem grana tako da je svaki graf, indukovan korisnicima zainteresovanim za istu temu, povezan. U pomenutom radu je pokazano da ne postoji polinomski algoritam kojim se može garantovati aproksimacija sa konstantnim faktorom, osim ako važi P=NP.

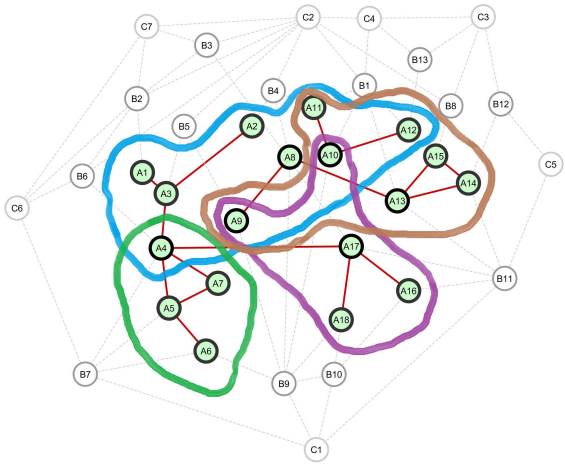
Iako pohlepni algoritam iz [7] rješava problem rekonstrukcije mreže sa cijenama na granama, oba rješenja predložena u [109] (ILPMinPPI i GreedyMinPPI) su formirana i primijenjena samo na netežinske PPI mreže. Kao što je već pomenuto, u ovom poglavlju je predstavljena metoda promjenljivih okolina koja može da se primjenjuje i nad netežinskim i nad težinskim PPI mrežama.

5.3 Trofazni metod za dodavanje PPI u mrežu

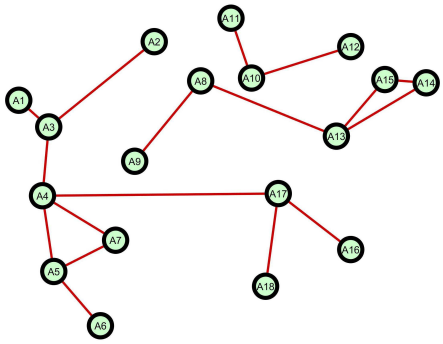
Predložena metoda se sastoji od tri faze:

- Faza I: podržavanje kompleksa dodavanjem PPI u postojeću težinsku PPI mrežu metodom promjenljivih okolina (rješavanje težinskog MinPPI problema);
- Faza II: spajanje određenih dijelova različitih kompleksa kombinacijom postojećih PPI i novih PPI dodatih u Fazi I;
- Faza III: dodavanje novih proteina grupama formiranim u Fazi II na osnovu indirektnih interakcija.

Primjer 5.1. *Na Slici 5.1 data je jednostavna mreža proteinskih interakcija. Mreža se sastoji od 38 proteina koji učestvuju u 76 interakcija i koji se nalaze u 4 kompleksa. Kompleksi su označeni sa K_1, K_2, K_3 i K_4 i redom oivičeni plavom, zelenom, braon i ljubičastom linijom. Grane koje postoje između proteina koji pripadaju nekom od kompleksa su predstavljene crvenom linijom, dok su ostale grane predstavljene tačkastim linijama. Identifikovana su tri tipa proteina:*



Slika 5.1: Početna mreža proteinskih interakcija



Slika 5.2: Protein-protein interakcije između A proteina

- *A proteini, odnosno proteini koji pripadaju nekom od kompleksa, označeni sa* $A_1 - A_{18}$*;*
- *B proteini, odnosno proteini koji imaju direktnu interakciju sa A proteinima, označeni sa* $B_1 - B_{13}$*;*
- *C proteini, odnosno proteini koji nemaju direktnu interakciju sa A proteinima, ali imaju direktnu interakciju sa B proteinima. Označeni sa* $C_1 - C_7$*.*

Proteini A su raspoređeni po kompleksima na sljedeći način:
$K_1 = \{A_1, A_2, A_3, A_4, A_8, A_9, A_{10}, A_{11}, A_{12}\}$
$K_2 = \{A_4, A_5, A_6, A_7\}$
$K_3 = \{A_9, A_{10}, A_{11}, A_{12}, A_{13}, A_{14}, A_{15}\}$
$K_4 = \{A_9, A_{10}, A_{16}, A_{17}, A_{18}\}$
Na Slici 5.2 su prikazani samo A proteini i njihove interakcije u polaznoj mreži proteinskih interakcija. Sa slike se može primijetiti da kompleksi K_1*,* K_3 *i* K_4 *nisu povezani.*

Cilj prve faze je da se doda što je moguće manje novih PPI tako da nepovezani kompleksi postanu povezani. Da bi se to postiglo primijenjen je algoritam baziran na metodi promjenljivih okolina.

```
1  VNS(v_cnt, complexes, fixedEdges, weights, k_min, k_max, it_max,
     itrep_max, t_max, p);
2  e_cnt ← v_cnt · (v_cnt − 1)/2;
3  sol.x ← fixedEdges;
4  sol.x ← fixGreedy(sol, v_cnt, complexes, weights);
5  k ← k_min;
6  it ← 1;
7  while it < it_max ∧ (it − it_lastimpr) < itrep_max ∧ t_run < t_max do
8  |  solNew ← shaking(sol,k, fixedEdges);
9  |  solNew ← fixGreedy(solNew,v_cnt, complexes, weights);
10 |  solNew ← localSearch(solNew, complexes, fixedEdges,
   |    weights);
11 |  if solNew.obj < sol.obj ∨ (solNew.obj = sol.obj ∧ U(0,1)_RV < p)
   |    then
12 |  |  sol ← solNew ;
13 |  |  k ← k_min;
14 |  else if k < k_max then
15 |  |  k ← k + 1;
16 |  else
17 |  |  k ← k_min;
18 |  it ← it + 1;
19 end
20 return sol;
```

Slika 5.3: Struktura VNS algoritma za MinPPI problem

5.3.1 Prva faza: podržavanje proteinskih kompleksa metodom promjenljivih okolina

Struktura VNS algoritma za rješavanje MinPPI problema prikazana je na Slici 5.3. Sljedeći parametri su argumenti algoritma:

- v_{cnt} je ukupan broj proteina u mreži;
- *complexes* je lista kompleksa, pri čemu je svaki kompleks lista proteina;
- $fixedEdges$ je lista postojećih PPI u mreži. Za MinPPI0 problem ova lista je prazna;
- *weights* je lista težina PPI. Za netežinsku varijantu problema sve grane su iste težine koja je jednaka 1.0;
- k_{min}, k_{max} su minimalna i maksimalna veličina VNS okoline;
- it_{max}, $itrep_{max}$ su maksimalan broj iteracija i maksimalan broj iteracija bez poboljšanja;
- t_{max} je maksimalno vrijeme izvršenja VNS algoritma u sekundama;
- p je vjerovatnoća prelaska iz jednog u drugo rješenje istog kvaliteta.

Inicijalno rješenje se formira pohlepnom strategijom na sljedeći način. Na početku je skup novih PPI koje se dodaju prazan, pa da bi se dobilo dopustivo rješenje koristi se procedura `fixGreedy`. Ova procedura lokalno popravlja svaki nepovezan kompleks dodavanjem grana kako bi se zadovoljio uslov povezanosti. U trenutku kada je formirano inicijalno rješenje, ono postaje najbolje rješenje. Dalja pretraga se nastavlja u glavnoj petlji algoritma, gdje se generišu kandidati za nova rješenja na osnovu najboljeg rješenja. Tokom faze generisanja novih mogućih rješenja primjenjuju se tri ključne procedure:

- `shaking`;

- `fixGreedy`;
- `localSearch`.

Kao što je uobičajno u VNS-u, procedura `shaking` kontroliše sistem okolina i na slučajan način bira nova rješenja iz trenutne okoline, sa ciljem da riješi situacije u kojima se lokalno optimalna rješenja ne mogu dalje poboljšati uzastopnim pozivima lokalne pretrage. Cilj `fixGreedy` procedure je brzo popravljanje rješenja, ako ono postane nedopustivo. Unutar `localSearch` procedure algoritam poboljšava prethodno popravljena rješenja sistematskim razmatranjem rješenja koja se nalaze u trenutnoj okolini. Procedure `shaking`, `fixGreedy` i `localSearch` su detaljno opisane u nastavku.

Nakon završetka faze lokalne pretrage upoređuju se trenutni kandidat za rješenje i trenutno najbolje rješenje. Ako je kandidatsko rješenje bolje od trenutno najboljeg rješenja, onda ono postaje najbolje rješenje. Ako su oba rješenja istog kvaliteta, onda kandidatsko rješenje postaje najbolje rješenje sa vjerovatnoćom p. Pri svakoj promjeni najboljeg rješenja, vrijednost varijable k, koja predstavlja veličinu okoline koja se razmatra, se vraća na početnu vrijednost k_{min}. Ako se ne desi promjena, k se povećava za 1, a svaki put kad dostigne maksimalnu vrijednost k_{max} vraća se na početnu vrijednost k_{min}.

Izvršenje VNS algoritma se zaustavlja ako je zadovoljen neki od sljedećih kriterijuma: dostignut je maksimalan broj dozvoljenih iteracija, dostignut je maksimalan broj iteracija bez poboljšanja trenutno najboljeg rješenja ili je dostignuto maksimalno vrijeme izvršenja.

Inicijalizacija i funkcija cilja

Neka je n ukupan broj proteina u mreži. Rješenje VNS algoritma je predstavljeno binarnom matricom $\mathbf{X}$ dimenzije n. Ako je $\mathbf{X}[i,j] = 1$, za neke $1 \leqslant i, j \leqslant n$, onda je PPI (grana u mreži) između proteina i i j uključena u rješenje, u suprotnom nije. Ako se rješava MinPPI0 problem, onda je početni skup PPI prazan i svaka PPI je kandidat za uključivanje u rješenje. Sa druge strane, za MinPPI problem postoji skup fiksiranih PPI koje moraju biti uključene u svako rješenje. Dakle, u slučaju MinPPI problema za svaku postojeću PPI između dva proteina i i j, odgovarajući element matrice $\mathbf{X}$ se postavlja na 1 i ne može se mijenjati tokom procesa pretrage.

Može se primijetiti da ovakva reprezentacija rješenja implicitno dozvoljava nedopustiva rješenja. Zbog toga, svaki put kada postoji mogućnost da je nakon uključivanja neke PPI rješenje postalo nedopustivo, algoritam poziva `fixGreedy` proceduru, koja popravlja trenutno rješenje tako da postane dopustivo.

Neka je sa $PPI_{i,j}$ označena protein-protein interakcija između proteina i i j. Neka je **Weights** matrica težina PPI, pri čemu je $\mathbf{Weights}[i,j]$ težina $PPI_{i,j}$, za neke $1 \leqslant i, j \leqslant n$. Algoritam računa funkciju cilja tako što sumira težine PPI po svim dodatim PPI na sljedeći način

$$Obj_{weighted}(Sol) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (1 - \mathbf{1}_{fixedEdges}(PPI_{i,j}))\mathbf{X}[i,j] \cdot \mathbf{Weights}[i,j] \tag{5.1}$$

gdje je $\mathbf{1}_{fixedEdges}$ indikatorska funkcija skupa $fixedEdges$, definisana sa

$$\mathbf{1}_{fixedEdges}(PPI_{i,j}) = \begin{cases} 1, & \text{ako postoji grana između proteina } i \text{ i } j \\ & \text{u polaznoj mreži ;} \\ 0, & \text{u suprotnom.} \end{cases}$$

Kao što je već napomenuto, u slučaju netežinske varijante problema, sve težine PPI su postavljene na 1.0, pa je u tom slučaju funkcija cilja

$$Obj_{unweighted}(Sol) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (1 - \mathbf{1}_{fixedEdges}(PPI_{i,j}))\mathbf{X}[i,j] \tag{5.2}$$

što je u suštini jednako broju dodatih grana. U obje formule, faktorom
$(1 - \mathbf{1}_{fixedEdges}(PPI_{i,j}))$ je kontrolisano da se samo novododate grane razmatraju u funkciji cilja.

Cilj predloženog VNS algoritma je rješavanje problema minimizacije navedene funkcije cilja.

Procedura razmrdavanja - `shaking`

Neka je *Sol* rješenje predstavljeno binarnom matricom $\mathbf{X}$ dimenzije n. Kao što je već navedeno, varijabla $fixedEdges$ označava skup PPI koje postoje u mreži. Za definisanje k-te okoline koristi se sljedeća procedura. Uklanja se nekih k PPI iz rješenja *Sol*, koje su prethodno dodate algoritmom. Preciznije, algoritam na slučajan način bira nekih k PPI tako što identifikuje 1-elemente u matrici $\mathbf{X}$, uz uslov da su odgovarajuće PPI novododate, tj. ne pripadaju skupu $fixedEdges$. Svih k izabranih elementa matrice se postavlja na 0, što znači da odgovarajuće PPI sada nisu uključene u rješenje. Tako formirano novo rješenje je dalje predmet analize u proceduri `fixGreedy`.

FixGreedy procedura

Može se primijetiti da uklanjanje PPI u proceduri razmrdavanja može da dovede do nedopustivih rješenja. Da bi se takva rješenja popravila i postala dopustiva, formirana je brza `fixGreedy` procedura (pseudokod dat na Slici 5.4) koja se primjenjuje na rješenje predloženo procedurom razmrdavanja, prije nego što se ono dalje proslijedi proceduri lokalne pretrage. Glavna petlja (6. linija pseudokoda) kontroliše kompletan proces korekcije koristeći varijablu *ce*, koja je jednaka razlici između ukupnog broja kompleksa i ukupnog broja povezanih komponenti po svim kompleksima. U slučaju da je vrijednost ove varijable manja od 0, onda postoji bar jedan nepovezan kompleks. Procedura popravlja svaki takav kompleks na sljedeći način: za svaka dva nepovezana proteina iz kompleksa, algoritam računa korist od dodavanja PPI koja bi povezala ta dva proteina. Navedena korist se računa kao odnos između:

(i) razlike između novog i starog broja povezanih komponenti;

(ii) težine PPI.

Nakon razmatranja svih takvih PPI, algoritam pohlepno dodaje onu PPI koja donosi maksimalnu korist, tj. PPI koja maksimalno smanjuje broj nepovezanih komponenti uzimajući u obzir i težinu te interakcije.

Opisani pristup izbora PPI, posmatran lokalno za trenutni kompleks, je sličan pristupu iz [7]. Za razliku od globalnog pohlepnog pristupa iz [7], `fixGreedy` procedura primjenjuje popravke samo na proteinima iz trenutno posmatranog kompleksa, što je mnogo efikasnije. Dakle, `fixGreedy` procedura se koristi kao međukorak za popravku rješenja između procedure razmrdavanja i procedure lokalne pretrage. Uključivanje ove procedure je dobar kompromis između potrebnog dodatnog vremena za izvršenje algoritma i koristi koja se postiže time što se procedura lokalne pretrage poziva isključivo na dopustiva rješenja.

```
1  fixGreedy(sol, complexes, weights);
2  solFixed ← sol;
3  cmpl_cnt ← |complexes|;
4  ce ← cmpl_cnt -
     connectedComponentsInComplexes(solFixed, complexes);
5  checkedEdges = ∅;
6  while ce < 0 do
7      diff_best ← 0;
8      e_best ← none;
9      foreach complex ∈ complexes do
10         if connectedComponentsInComplexes(sol, complex) = 1 then
11             continue;
12         foreach e ∈ {{u, v}|u, v ∈ complex.vertices} do
13             if e ∈ solFixed.x ∨ e ∈ checkedEdges then
14                 continue;
15             solFixed.x ← solFixed.x ∪ {e};
16             ceNew ← cmpl_cnt −
                 connectedComponentsInComplexes(solFixed, complexes);
17             ceDiffWeighted ← (ce − ceNew)/weights[e];
18             if ceDiffWeighted < diff_best then
19                 diff_best ← ceDiffWeighted;
20                 e_best ← e;
21             solFixed.x ← solFixed.x \ {e};
22             checkedEdges ← checkedEdges ∪ {e};
23         end
24     end
25     solFixed.x ← solFixed.x ∪ {e_best};
26     ce ← cmpl_cnt −
         connectedComponentsInComplexes(solFixed, complexes);
27 end
28 solFixed.obj ← obj(solFixed, complexes, weights);
29 return solFixed;
```

Slika 5.4: Procedura FixGreedy

Procedura lokalne pretrage

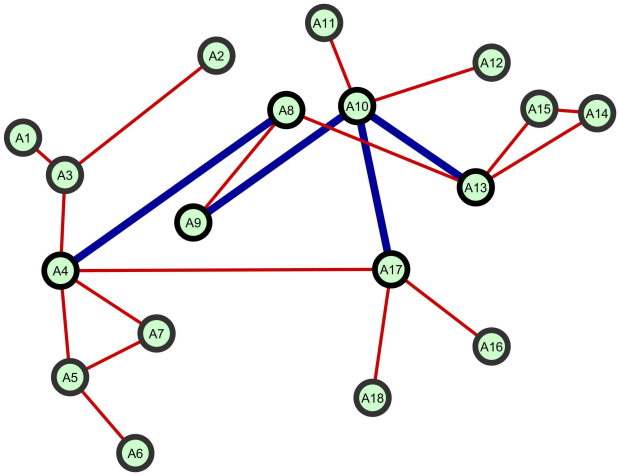
Rješenje formirano u proceduri razmrdavanja, i eventualno, popravljeno procedurom `fixGreedy` je dalje predmet poboljšanja u proceduri lokalne pretrage. Pseudokod za proceduru lokalne pretrage je prikazan na Slici 5.5. Glavni dio lokalne pretrage je petlja (`foreach` petlja koja se nalazi u 8. liniji pseudokoda), u okviru koje se ispituje potencijalna korist od uklanjanja PPI iz rješenja. Unutar petlje, algoritam za svako takvo uklanjanje PPI (označena sa e u pseudokodu) poziva proceduru parcijalnog računanja funkcije cilja. Da bi se to postiglo, formirana je brza procedura parcijalnog računanja funkcije cilja, označena sa `objDiff` (pseudokod je dat na Slici 5.6). Unutar procedure `objDiff` algoritam računa razliku između vrijednosti funkcije cilja u slučaju kada se PPI e pojavljuje

```
1  localSearch(sol, v_cnt, complexes, fixedEdges, weights);
2  solImpr ← sol;
3  impr ← true;
4  while impr do
5      impr ← false;
6      diff_best ← 0;
7      e_best ← none;
8      foreach e ∈ {{u, v}|u, v ∈ {1, 2, ..., v_cnt}} do
9          if e ∉ solImpr.x ∨ e ∈ fixedEdges then
10             continue;
11         diff ← objDiff(solImpr, e, complexes, weights);
12         if diff < diff_best then
13             diff_best ← diff;
14             e_best ← e;
15             impr ← true;
16     end
17     if impr then
18         solImpr.x ← solImpr.x \ {e_best};
19         solImpr.obj ← obj(solImpr, complexes, weights);
20 end
21 return solImpr;
```

Slika 5.5: Procedura lokalne pretrage

```
1  objDiff(sol, e, complexes, weights);
2  sol.x ← sol.x \ {e};
3  diff ← −weights[e];
4  foreach complex ∈ complexes do
5  |  related ← false;
6  |  foreach w ∈ complex.vertices do
7  |  |  if w ∈ e then
8  |  |  |  related = true;
9  |  |  |  break;
10 |  end
11 |  if related then
12 |  |  diff ← diff+edgeWeightsToFix(sol, e, complex, weights);
13 end
14 sol.x ← sol.x ∪ {e};
15 return diff;
```

Slika 5.6: Parcijalno računanje funkcije cilja



Slika 5.7: Izdvojeni A proteini nakon prve faze

u rješenju i vrijednosti funkcije cilja u slučaju da je ta PPI izostavljena, a, umjesto te interakcije, dodate su neke druge PPI u cilju popravljanja rješenja. Negativna vrijednost ove razlike znači da je rješenje dopustivo čak i bez PPI e ili da je algoritam uspio da pronađe alternativne PPI koje mogu biti uključene u rješenje umjesto PPI e da bi rješenje ostalo dopustivo ali je ukupna vrijednost funkcije cilja manja.

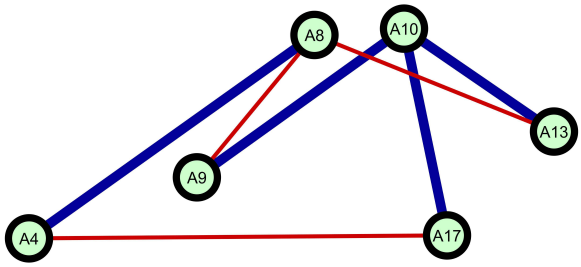
Algoritam koristi strategiju najboljeg unapređenja (engl. best improvement strategy), što znači da se trenutno najbolje rješenje zamjenjuje najboljim alternativnim rješenjem u trenutnoj iteraciji lokalne pretrage. Ako se desi poboljšanje, ono se primjenjuje na rješenje, računa se nova vrijednost funkcije cilja i lokalna pretraga prelazi u sljedeću iteraciju.

Procedura lokalne pretrage se zaustavlja kada se više ne može postići poboljšanje.

Rezultat primjene algoritma promjenljivih okolina na mrežu iz primjera 5.1 je prikazan na Slici 5.7. Dodate grane A_4A_8, A_9A_{10},$A_{10}A_{13}$ i $A_{10}A_{17}$ su označene plavom bojom.

5.3.2 Druga faza: spajanje

Nakon što je VNS algoritmom dodat minimalan broj novih PPI koje podržavaju proteinske komplekse, algoritam ulazi u drugu fazu. Cilj druge faze je grupisanje proteina iz različitih kompleksa koji su u početnoj PPI mreži slabo povezani. Procedura druge faze je sljedeća. Fokus je stavljen na proteine koji su krajevi PPI koje su dodate VNS algoritmom. Dalje se razmatraju podmreže koje sadrže te proteine i sve PPI koje su incidentne sa njima. Pod svim PPI misli se PPI koje postoje u



Slika 5.8: Izdvojeni A proteini nakon druge faze

mreži i PPI koje su dodate u fazi I. Na taj način se povezuju određeni dijelovi različitih kompleksa, a proteini se grupišu u veće povezane grupe, koje se u daljem tekstu nazivaju *bazne grupe*.

Za mrežu iz primjera 5.1 izgled strukture nakon druge faze je prikazan na Slici 5.8. Kao što se može vidjeti sa slike, ovako formirana podmreža sadrži proteine iz različitih kompleksa. Bazne grupe imaju mali broj PPI, ali su sada povezane strukture i predstavljaju dobru osnovu za treću fazu.

5.3.3 Treća faza: dodavanje novih proteina

Nakon što su proteini iz nekoliko kompleksa grupisani u povezanu strukturu, svaka bazna grupa proteina je dalje proširena dodavanjem novih proteina. Za proširenje se razmatraju proteini koji nemaju direktne interakcije sa proteinima iz bazne grupe. Da bi bio dodat, protein treba da zadovoljava sljedeća dva uslova:

- nalazi se na rastojanju 2 od svakog proteina iz bazne grupe;
- prosječna vrijednost genske ko-ekspresije sa svakim od proteina iz bazne grupe je iznad datog praga.

Prvim uslovom je obezbjeđeno da:

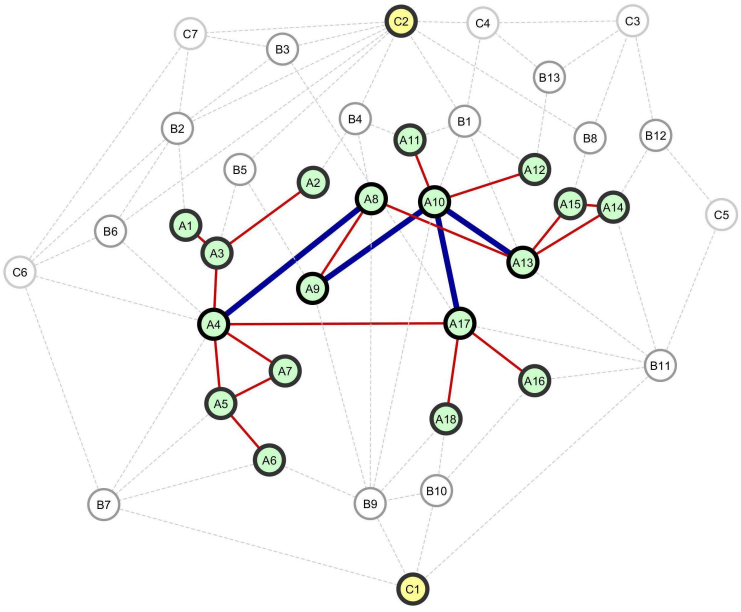
(a) ne postoji direktna grana između bazne grupe proteina i proteina koji se dodaju;

(b) za svaki par koji čini protein koji se dodaje i protein iz bazne grupe postoji bar jedan protein sa kojim oba imaju interakciju (u primjeru 5.1 to su B proteini).

Drugim riječima, razmatraju se samo indirektne interakcije između dva proteina, za koje je već dokazano da obično dijele zajedničku biološku funkciju [30]. Drugi uslov, koji je detaljnije objašnjen u Odjeljku 5.5, poboljšava preciznost ovog pristupa tako što eliminiše "slabe" indirektne interakcije.

Na Slici 5.9 su označena dva C proteina, odnosno proteini C_1 i C_2, koji zadovoljavaju prvi navedeni uslov. Za mrežu iz primjera 5.1 značajnu grupu proteina čine: A_4, A_8, A_9, A_{10},A_{13}, A_{17}, C_1 i C_2.

Posljedica upotrebe drugog kriterijuma je to što će u veću grupu proteina biti uključeni samo oni proteini koji imaju jače funkcionalne veze sa proteinima iz bazne grupe. Jačina funkcionalne veze između dva proteina se može procijeniti različitim kriterijumima [29, 88]. U ovom istraživanju se kao parametar koristi genska ko-ekspresija. Za svaki protein koji je kandidat za uključivanje u grupu (tj. koji zadovoljava prvi uslov) računa se prosječna genska ko-ekspresija sa svim proteinima iz bazne grupe. Drugim riječima, od svih proteina koji se nalaze na rastojanju 2 od svakog proteina iz bazne grupe, biraju se samo oni za koje je vrijednost prosječne genske ko-ekspresije iznad određenog praga.



Slika 5.9: Označeni proteini koji se razmatraju u trećoj fazi

Na taj način se izostavljaju indirektni proteini čija je prosječna genska ko-ekspresija sa baznom grupom proteina niska, a sa druge strane opet većina proteina koji indirektno interaguju ostaje uključena.

Kao što je i pokazano u Odjeljku 5.4, grupe proteina formirane opisanom metodom imaju izuzetno visoke skorove obogaćivanja informacijama, što ukazuje na njihovu značajnu semantičku povezanost.

5.4 Rezultati testiranja

U ovom odjeljku su prikazani rezultati testiranja predloženog VNS algoritma. Sva testiranja su vršena na računaru Intel i7-4770 CPU@3.40GHz sa 8 GB RAM i Windows 7 64Bit operativnim sistemom. Za svako izvršenje korištena je jedna nit/jedno jezgro. VNS algoritam je implementiran u programskom jeziku C i kompajliran Visual Studio 2019 kompajlerom.

Za svaku instancu, VNS je izvršen 10 puta sa različitim inicijalnim podešavanjem generatora pseudoslučajnih brojeva. Izvršavanje se zaustavlja kada je zadovoljen neki od sljedećih uslova:

- dostignut je maksimalan broj iteracija, $it_{max} = 20000$;
- dostignut je maksimalan broj iteracija bez poboljšanja, $itrep_{max} = 5000$;
- dostignuto maksimalno vrijeme izvršenja, $t_{max} = 1200$ sekundi.

Ostali kontrolni parametri su:

- minimalna veličina VNS okoline, $k_{min} = 1$;
- maksimalna veličina VNS okoline, $k_{max} = 5$;
- vjerovatnoća prelaska iz jednog rješenja u drugo rješenja istog kvaliteta, $prob = 0.5$.

Da bi poređenje performansi predloženog VNS algoritma u odnosu na druge metode iz literature bilo izvršeno pod istim uslovima, implementirani su ILP model i pohlepni algoritam iz [109] i testirani na istom računaru. Pored toga, prateći princip iz [7], pohlepni algoritam iz [109] je prilagođen da radi sa težinskim instancama. Pohlepni algoritam je implementiran u programskom jeziku C i kompajliran Visual Studio 2019 kompajlerom, a ILP model u programskom jeziku Python 2.7 i pokrenut CPLEX 12.1 rješavačem.

VNS algoritam je testiran na dostupnim vještačkim i realnim PPI skupovima podataka koji su korišteni u [109]. Pored toga, testiranje je vršeno i na još 6 realnih PPI instanci, a formiran je jedan skup slučajnih instanci. Korišteni skupovi podataka su opisani u narednom odjeljku.

5.4.1 Skupovi podataka

Za testiranje performansi korištene su vještačke i realne instance. Razmatrani su sljedeći skupovi podataka dostupni u literaturi:

- SYNDATA , koji su takođe korišteni u [109] i sadrže dva skupa koji imaju po 10 proteinskih kompleksa koji su vještački formirani:
  - syndata1 - (s1data1-s1data10), u kojima je maksimalan broj ukupnog broja proteina, kompleksa i maksimalna kardinalnost kompleksa 10, 20 i 5, respektivno;
  - syndata2 - (s2data1-s2data10), u kojima je maksimalan broj ukupnog broja proteina, kompleksa i maksimalna kardinalnost kompleksa 100, 100 i 4, respektivno.
- CYCDATA, koji sadrže 32 vještačke instance proteinskih kompleksa koje su podjeljene u 4 grupe:
  - cyc4, maksimalna kardinalnost kompleksa je 4;
  - cyc5, maksimalna kardinalnost kompleksa je 5;
  - cyc6, maksimalna kardinalnost kompleksa je 6;
  - cyc7, maksimalna kardinalnost kompleksa je 7;

  Može se primijetiti da je skup podataka CYCDATA sličan skupovima podataka koji su predstavljeni na Slici 3 (Fig. 3) u [109]. Međutim, kao što će biti pokazano kasnije, rezultati dobijeni nad CYCDATA skupovima podataka ILP algoritmom koji je implementiran za ovo istraživanje se razlikuju od rezultata koji su predstavljeni u [109]. Ova činjenica ukazuje na to da postoji razlika između skupova podataka ili da postoji razlika u implementaciji ILP-a.
- Skupovi proteinskih kompleksa, takođe korišteni u [109]:
  - CYC2008, koji sadrži 1627 proteina i 408 kompleksa;
  - MIPS, koji sadrži 1189 proteina i 203 kompleksa.
- Dvije grupe realnih PPI mreža:
  - Prvu grupu čine PPI mreže koje su korištene i u [109]: String [47], BioGRID [112], WI-PHI [72], iRefIndex [123], MINT [87]. S obzirom da se pomenute PPI mreže često ažuriraju, nije bilo moguće pristupti istim verzijama mreža koje su korištene u [109]. Zbog navedenih

razloga, u ovom istraživanju korištene su posljednje verzije javno dostupnih podataka o PPI mrežama i proteinskim kompleksima. Prilikom upotrebe ovih mreža primijećeno je da je možda samo BioGRID mreža ista kao u [109].

– Drugu grupu čine 4 PPI mreže koje su prvi put u ovom istraživanju korištene pri rješavanju MinPPI problema: collins2007 PPI mreža koja sadrži prvih 9,074 interakcija u odnosu na njihov skor za obogaćivanje informacija [33], gavin2006 PPI mreža koja sadrži PPI čiji je indeks društvene sklonosti (engl. socio-affinity index) veći od 5 [49], krogan2006-core PPI mreža koja sadrži samo veoma pouzdane interakcije (vjerovatnoća > 0.273) i krogan2006-extended PPI mreža koja sadrži više interakcija čija je ukupna pouzdanost manja (vjerovatnoća > 0.101) [76].

- Novi skup od 12 slučajno generisanih instanci, koje sadrže do 100 proteina raspoređenih u maksimalno 1000 kompleksa.

5.4.2 Poređenje performansi VNS sa ILP i pohlepnim algoritmom

U nastavku su predstavljeni rezultati upoređivanja ILP i pohlepnog algoritma iz [109] sa predloženim VNS algoritmom. Kao što je već navedeno, da bi poređenje bilo realizovano pod istim uslovima ILP i pohlepni algoritam su implementirani i testirani na istom računaru na kojem je testiran VNS algoritam.

Rezultati dobijeni rješavanjem MinPPI0 problema

Algoritam za rješavanje MinPPI0 problema je testiran na dva skupa vještačkih instanci (SYNDATA skupovi) i na dvije realne instance proteinskih kompleksa (CYC2008 i MIPS). Rezultati su prikazani u Tabeli 5.1. Tabela je organizovana na sljedeći način. Prve 4 kolone sadrže ime instance, ukupan broj proteina, ukupan broj kompleksa i informaciju o maksimalnoj kardinalosti kompleksa, respektivno. Sljedeće 4 kolone sadrže podatke o rezultatima dobijenim ILP i pohlepnim algoritmom, odnosno broj dodatih grana (tj. broj dodatih PPI) i vrijeme izvršenja u sekundama. Posljednje tri kolone sadrže najbolji rezultat, prosječan rezultat i prosječno vrijeme izvršenja (u sekundama) koji su dobijeni u 10 pokretanja VNS algoritma.

S obzirom da vještačke instance iz SYNDATA skupova nisu velike, ILP je pronašao sva optimalna rješenja za kratko vrijeme. Vrijeme izvršenja ILP algoritma i rezultati dobijeni pohlepnim algoritmom se razlikuju od vremena i rezultata iz [109], zbog upotrebe različite verzije ILP rješavača i implementacije pohlepnog algoritma. Iz Tabele 5.1 se može primijetiti da VNS pronalazi sva optimalna rješenja u svih 10 izvršenja (posljedica toga je da su najbolje i prosječno rješenje jednaki), dok pohlepni algoritam za dvije instance s1data4 i s1data7 ne dostiže optimalno rješenje. Implementacija pohlepnog algoritma, koja je urađena za ovo istraživanje iz već pomenutih razloga testiranja pod istim uslovima, za instancu s1data7 pronalazi rješenje 17, koje nije optimalno, ali je bolje od rješenja pohlenog algoritma iz [109] koje iznosi 25 (navedni rezultat je prikazan u dodatnom materijalu rada [109]).

Instance prikazane u posljednja dva reda Tabele 5.1 predstavljaju stvarne proteinske komplekse i velikih su dimenzija, pa zbog memorijskih ograničenja nisu rješavane ILP rješavačem, te se ne može garantovati optimalnost rješenja koja su dobijena pohlepnim i VNS algoritmom. Može se primijetiti da oba algoritma dostižu ista rješenja, ali da VNS algoritam pronalazi rješenje u značajno kraćem vremenu. Za instancu CYC2008 vrijeme izvršenja VNS algoritma je oko 20 puta manje nego vrijeme

Tabela 5.1: Rezulatati za MinPPI0 problem

instanca	#vert	#compl	max card.	ILP		Greedy		VNS		
				res	t[s]	res	t[s]	best	avg	t[s]
s1data1	3	1	3	2	0.156	2	0	2	2	0.2037
s1data2	10	7	5	13	30.997	13	0.003	13	13	0.2844
s1data3	10	12	4	16	0.203	16	0.004	16	16	0.2694
s1data4	10	14	5	15	4.478	16	0.004	15	15	0.6584
s1data5	10	13	5	14	1.201	14	0.004	14	14	0.4133
s1data6	5	1	5	4	0.374	4	0.001	4	4	0.1436
s1data7	10	19	5	16	0.312	17	0.004	16	16	0.548
s1data8	10	12	5	13	19.344	13	0.001	13	13	0.6034
s1data9	6	2	4	5	0.046	5	< 0.001	5	5	0.1486
s1data10	10	20	5	19	0.515	19	0.005	19	19	0.5845
s2data1	100	86	4	166	4.321	166	0.747	166	166	1.7429
s2data2	100	68	4	139	1.857	139	0.544	139	139	1.3749
s2data3	100	93	4	180	4.587	180	0.773	180	180	1.9032
s2data4	100	55	4	111	1.294	111	0.272	111	111	1.1191
s2data5	100	50	4	92	0.812	92	0.187	92	92	0.9656
s2data6	100	71	4	136	1.638	136	0.414	136	136	1.3192
s2data7	100	37	4	74	0.936	74	0.127	74	74	0.8543
s2data8	100	79	4	150	1.513	150	0.541	150	150	1.4703
s2data9	100	11	4	20	0.203	20	0.011	20	20	0.5555
s2data10	100	34	4	67	0.78	67	0.115	67	67	0.7811
CYC2008	1627	408	57	*	*	1344	10614.38	1344	1344	507.5904
MIPS	1189	203	95	*	*	1154	31499.91	1154	1154	1200.232

izvršenja pohlepnog algoritma, dok je, za MIPS instancu, VNS brži od pohlepnog algoritma oko 26 puta.

Rezultati dobijeni nad CYCDATA podacima su prikazani u Tabelama 5.2-5.5. Tabele su organizovane na isti način kao i Tabela 5.1. VNS i pohlepni algoritam pronalaze sva poznata optimalna rješenja. Međutim, zbog veličine instance i memorijskih ograničenja, za većinu instanci iz skupa cyc6 (Tabela 5.4) i sve instance iz skupa cyc7 (Tabela 5.5) ILP rješavač ne uspijeva da u razumnom vremenu pronađe rješenje, tako da se za te rezultate ne može garantovati optimalnost. Kao i u prethodnom slučaju, VNS je u svih 10 izvršenja pronašao iste rezultate. Takođe, za sve instance VNS-om i pohlepnim algoritmom se dobijaju jednaka rješenja.

Tabela 5.2: Rezultati nad cyc4 podacima

instanca	#vert	#compl	max card.	ILP		Greedy		VNS		
				res.	t[s]	res.	t[s]	best	avg.	t[s]
4CYC2008_1	132	50	4	84	0.748	84	0.274	84	84	1.4634
4CYC2008_2	258	100	4	167	1.669	167	3.476	167	167	4.1221
4CYC2008_3	378	150	4	243	2.121	243	13.302	243	243	8.8593
4CYC2008_4	488	200	4	314	2.09	314	37.73	314	314	16.7105
4CYC2008_5	589	250	4	384	2.855	384	79.926	384	384	24.32
4CYC2008_6	698	300	4	459	3.151	459	150.298	459	459	36.8358
4CYC2008_7	703	303	4	464	3.307	464	154.302	464	464	37.3338
4CYC2008_8	703	303	4	464	2.917	464	175.323	464	464	38.3801

Tabela 5.3: Rezultati nad cyc5 podacima

instanca	#vert	#compl	max card.	ILP		Greedy		VNS		
				res.	t[s]	res.	t[s]	best	avg.	t[s]
5CYC2008_1	139	50	5	91	1.856	91	0.46	91	91	1.3718
5CYC2008_2	272	100	5	182	15.709	182	5.367	182	182	4.375
5CYC2008_3	396	150	5	263	18.346	263	20.88	263	263	9.3879
5CYC2008_4	511	200	5	338	24.446	338	51.072	338	338	17.828
5CYC2008_5	626	250	5	422	44.866	422	120.585	422	422	28.2189
5CYC2008_6	732	300	5	492	59.515	492	219.131	492	492	38.1944
5CYC2008_7	786	324	5	534	63.648	534	289.495	534	534	45.2417
5CYC2008_8	786	324	5	534	54.834	534	296.062	534	534	45.9095

Tabela 5.4: Rezultati nad cyc6 podacima

instanca	#vert	#compl	max card.	ILP		Greedy		VNS		
				res.	t[s]	res.	t[s]	best	avg.	t[s]
6CYC2008_1	149	50	6	101	40.248	101	0.732	101	101	1.4482
6CYC2008_2	285	100	6	193	162.193	193	5.429	193	193	4.6653
6CYC2008_3	414	150	6	o.m.	o.m.	281	23.495	281	281	9.7483
6CYC2008_4	537	200	6	o.m.	o.m.	362	64.267	362	362	17.5
6CYC2008_5	656	250	6	o.m.	o.m.	451	152.752	451	451	26.4603
6CYC2008_6	786	300	6	o.m.	o.m.	543	278.445	543	543	40.7311
6CYC2008_7	900	345	6	o.m.	o.m.	630	498.661	630	630	57.3483
6CYC2008_8	900	345	6	o.m.	o.m.	630	425.392	630	630	56.6603

Rezultati dobijeni rješavanjem MinPPI problema nad biološkim mrežama

U ovom odjeljku su prikazani ekeperimentalni rezultati dobijeni pri rješavanju MinPPI problema VNS algoritmom i pohlepnim algoritmom, koji je implementiran za potrebe ovog istraživanja. Za testiranje korišten je skup CYC2008 proteinskih kompleksa i 9 PPI mreža. Pet PPI mreža (String, BioGRID, WI-PHI, iRefIndex, MINT) je testirano i u [109], ali zbog različitih verzija mreža koje su javno dostupne, ne može se garantovati da su to potpuno iste instance. Preostale četiri instance (collins2007, Gavin2006, Krogan2006_core, Krogan2006_extended) nisu korištene u pomenutnom radu i prvi put su u ovom istraživanju upotrebljene za MinPPI problem. Prateći metod iz [109], PPI mreže su redukovane na mreže koje sadrže samo 1627 proteina iz CYC2008 standarda.

Tabela 5.6 sadrži rezultate dobijene nad netežinskim PPI mrežama. U prve tri kolone su prikazani naziv instance, ukupan broj PPI u originalnoj mreži i ukupan broj PPI nakon redukcije na proteine iz CYC2008 standarda. Ostatak tabele sadrži rezultate dobijene pohlepnim i VNS algoritmom, koji su organizovani na isti način kao u Tabeli 5.1.

Iz Tabele 5.6 se može vidjeti da je broj dodatih PPI u opsegu od 0 (za String mrežu) do 625 (za Gavin2006 mrežu). Iz činjenice da se u String mrežu ne dodaje nijedna nova grana slijedi da je skup proteinskih kompleksa iz CYC2008 standarda potpuno podržan u originalnoj String mreži, odnosno da su svi kompleksi povezani u String mreži. BioGRID, WI-PHI i iRefIndex mreže nakon redukcije sadrže više od 10000 PPI, pa je i broj dodatih PPI znatno manji nego za ostale mreže. Ako se posmatra kvalitet rezultata, može se primijetiti da se za oba algoritma dobijaju isti rezultati tj. dodaje se jednak broj PPI za sve instance, s tim da je VNS algoritam znatno brži, u većini slučajeva do jednog reda veličine. Takođe, može se primijetiti da je vrijeme izvršenja pohlepnog algoritma znatno veće za mreže u kojima proteinski kompleksi nisu tako dobro podržani (posljednjih

Tabela 5.5: Rezultati nad cyc7 podacima

instanca	#vert	#compl	max card.	ILP		Greedy		VNS		
				res.	t[s]	res.	t[s]	best	avg.	t[s]
7CYC2008_1	152	50	7	o.m.	o.m.	104	0.432	104	104	1.4969
7CYC2008_2	292	100	7	o.m.	o.m.	200	5.817	200	200	4.6452
7CYC2008_3	429	150	7	o.m.	o.m.	296	21.309	296	296	10.6471
7CYC2008_4	555	200	7	o.m.	o.m.	380	68.739	380	380	18.3524
7CYC2008_5	681	250	7	o.m.	o.m.	474	134.681	474	474	28.927
7CYC2008_6	821	300	7	o.m.	o.m.	578	319.914	578	578	43.1374
7CYC2008_7	954	350	7	o.m.	o.m.	678	525.337	678	678	61.2066
7CYC2008_8	963	355	7	o.m.	o.m.	687	575.537	687	687	62.6751

Tabela 5.6: Rezultati dobijeni nad netežinskim PPI mrežama

instanca	#PPIs prije	#PPIs poslije	Greedy		VNS		
			res.	t[s]	best	avg.	t[s]
String	777589	145765	0	7.708	0	0	
BioGRID	59748	16180	67	1957.189	67	67	1200.221
WI-PHI	50000	12685	93	2070.795	93	93	943.6255
iRefIndex	259645	22429	86	1883.417	86	86	1198.017
MINT	40619	5053	427	7976.703	427	427	786.8882
Collins2007	9074	6392	466	8590.775	466	466	486.8771
Gavin2006	7669	4031	625	7109.146	625	625	658.9797
Krogan2006_core	7123	3169	587	5984.598	587	587	722.4588
Krogan2006_extended	14317	4714	563	5673.091	563	563	738.5703

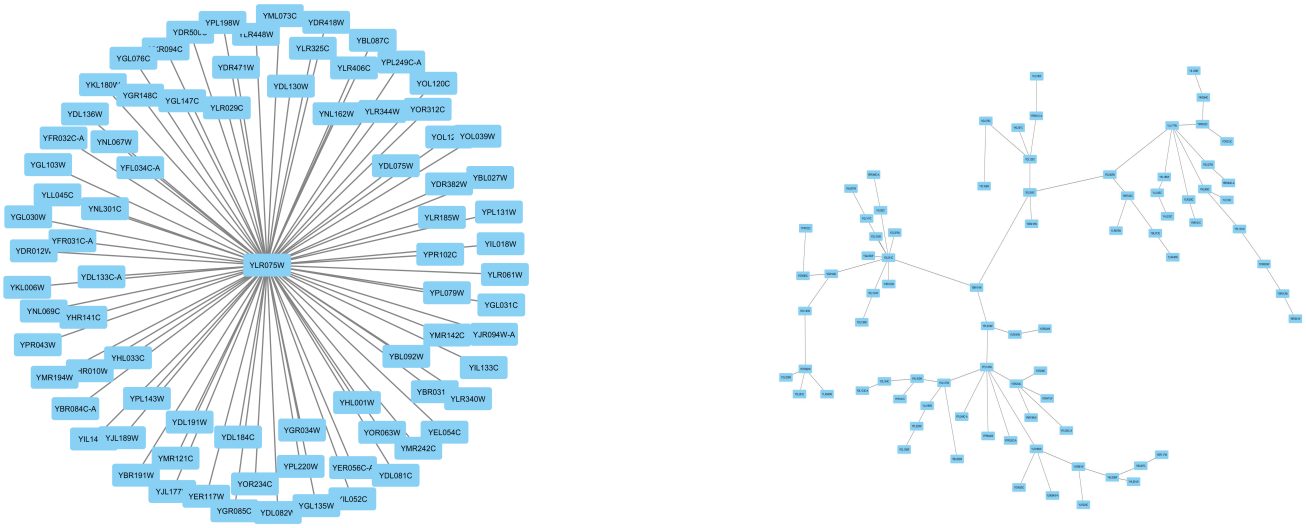
pet redova Tabele 5.6). S druge strane, vrijeme izvršenja VNS algoritma je proporcionalno veličini instance i ne zavisi od broja dodatih PPI.

5.4.3 Rezultati na težinskim instancama

U ovom odjeljku su uvedene težine u PPI mrežu i riješena je težinska varijanta problema, koja do sada nije razmatrana u literaturi.

Kao što je već pomenuto u Odjeljku 5.1, u ovom istraživanju težine PPI su bazirane na vrijednost genske ko-ekspresije. Za određivanje vrijednosti genske ko-ekspresije između dva proteina korišten je poznati SPELL alat [61]. Za svaki par (P, Q) proteina iz PPI mreže računa se prilagođeni koeficijent korelacije (engl. Adjusted Correlation Score (ACS)), što je mjera težinske korelacije gena koji odgovaraju proteinima P i Q. ACS se može odrediti za bilo koji par proteina, bez obzira da li su povezani u PPI mreži. SPELL alat za zadati protein (npr. zadato sistemsko ime proteina, poput YHR002W) vraća rangiranu listu proteina (gena) sa ACS vrijednostima između proteina koji je upit i svih proteina iz baze podataka. Opisani postupak je primijenjen na 1627 proteina iz CYC2008 standarda i određena je ACS vrijednost za svaki par proteina. Za svaku PPI u mreži dobijena ACS vrijednost predstavlja težinu grane koja spaja te proteine. Na taj način formirana je polazna težinska PPI mreža.

Kao što je već rečeno u Odjeljku 5.3.1, predloženi VNS algoritam rješava optimizacioni problem minimizacije, odnosno minimizuje funkciju (5.1). S obzirom da je ideja da se favorizuju interakcije čija je genska ko-ekspresija veća, formirane su "inverzne težine" pomoću sljedeće formule:



Slika 5.10: Povezan kompleks cytoplasmic ribosomal large subunit - netežinskom verzijom algoritma (lijevo) i težinskom verzijom algoritma (desno)

$$w_{new}(PPI) = CoExpression_{max} - ACS(PPI) \tag{5.3}$$

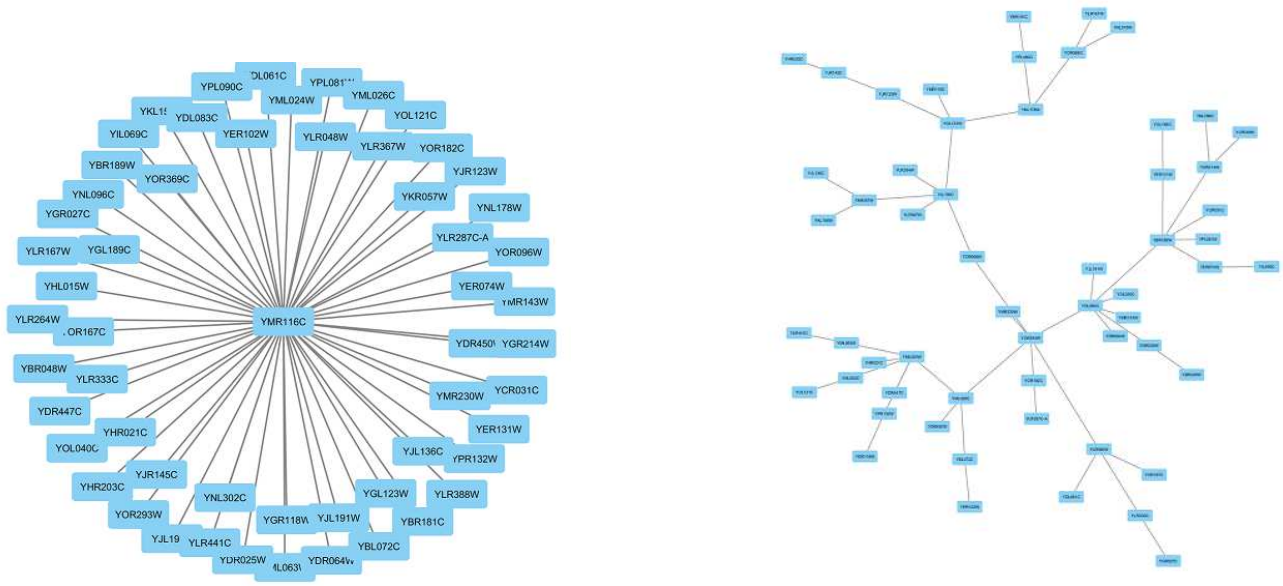
gdje je $CoExpression_{max}$ maksimalna ko-ekspresija (maksimalna vrijednost ACS) između dva proteina u cijeloj PPI mreži, a $ACS(PPI)$ prilagođeni koeficijent korelacije posmatrane interakcije. Formula (5.3) je primijenjena na sve grane mreže i tako je formirana "inverzna" mreža. Ova jednostavna transformacija početne PPI mreže omogućava da predloženi VNS algoritam, koji rješava problem minimizacije, favorizuje interakcije sa većom genskom ko-ekspresijom.

Rješavanje MinPPI0 problema na težinskim instancama

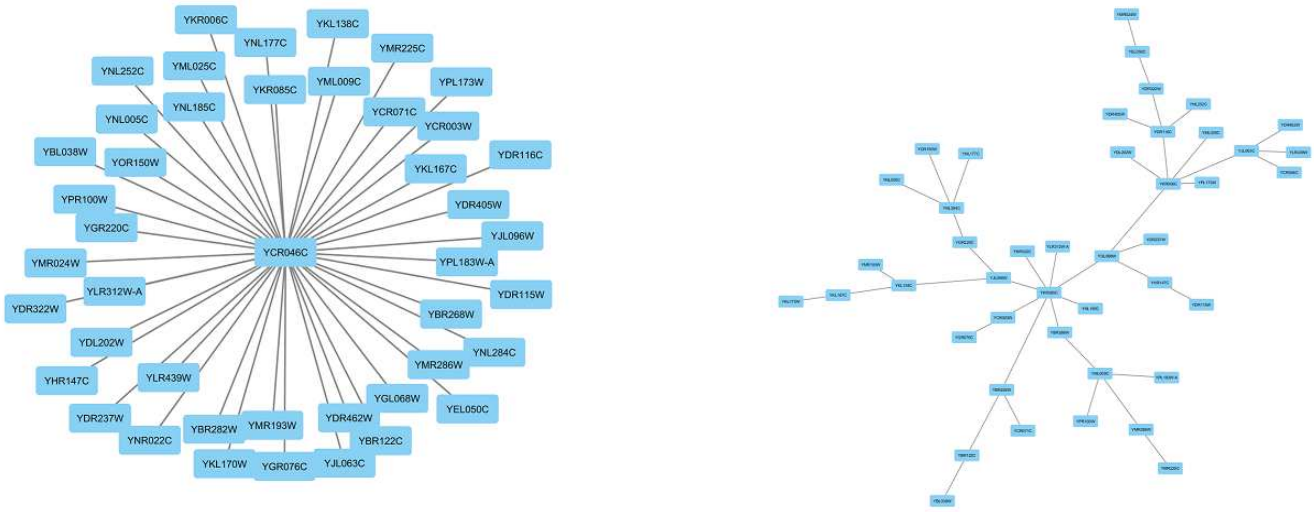
Da bi se opravdala upotreba vrijednosti genske ko-ekspresije kao težina PPI, upoređena su rješenja netežinske i težinske varijante MinPPI0 problema na instanci CYC2008. Strukture dobijene nakon izvršenja netežinskog i težinskog algoritma za nekoliko kompleksa iz ovog standarda, odnosno za *cytoplasmic ribosomal large subunit* kompleks (Slika 5.10), *cytoplasmic ribosomal small subunit* kompleks (Slika 5.11), *mitochondrial ribosomal large subunit* kompleks (Slika 5.12) i *Kornberg's mediator (SRB)* kompleks (Slika 5.13), te za kompleks 19S (Slike 5.14 i 5.15), koji će biti detaljnije analiziran u nastavku, su prikazane na navedenim slikama. Na Slici 5.14 prikazan je podgraf indukovan granama koje podržavaju 19S kompleks u slučaju kad se ne koriste težine, dok je na Slici 5.15 prikazan podgraf dobijen ako se u obzir uzimaju težine grana.

Analiziranjem strukture sa Slike 5.14 može se zaključiti da je izabran jedan protein (vjerovatno prvi nepovezan protein iz liste proteina koji čine kompleks) i da algoritam dalje nastoji da taj protein poveže sa ostalima u cilju smanjenja nivoa nepovezanosti. Na taj način algoritam formira strukture koje liče na zvijezda grafove. Na Slikama 5.10-5.13 se može uočiti da se slična situacija javlja i za ostale komplekse, ako se koristi netežinska varijanta algoritma. Sa druge strane, ako su pri dodavanju novih grana uzete u obzir vrijednosti genske ko-ekspresije kao težine na granama, algoritam teži da formira biološki smislenija rješenja, što se može potvrditi sljedećim razmatranjem.

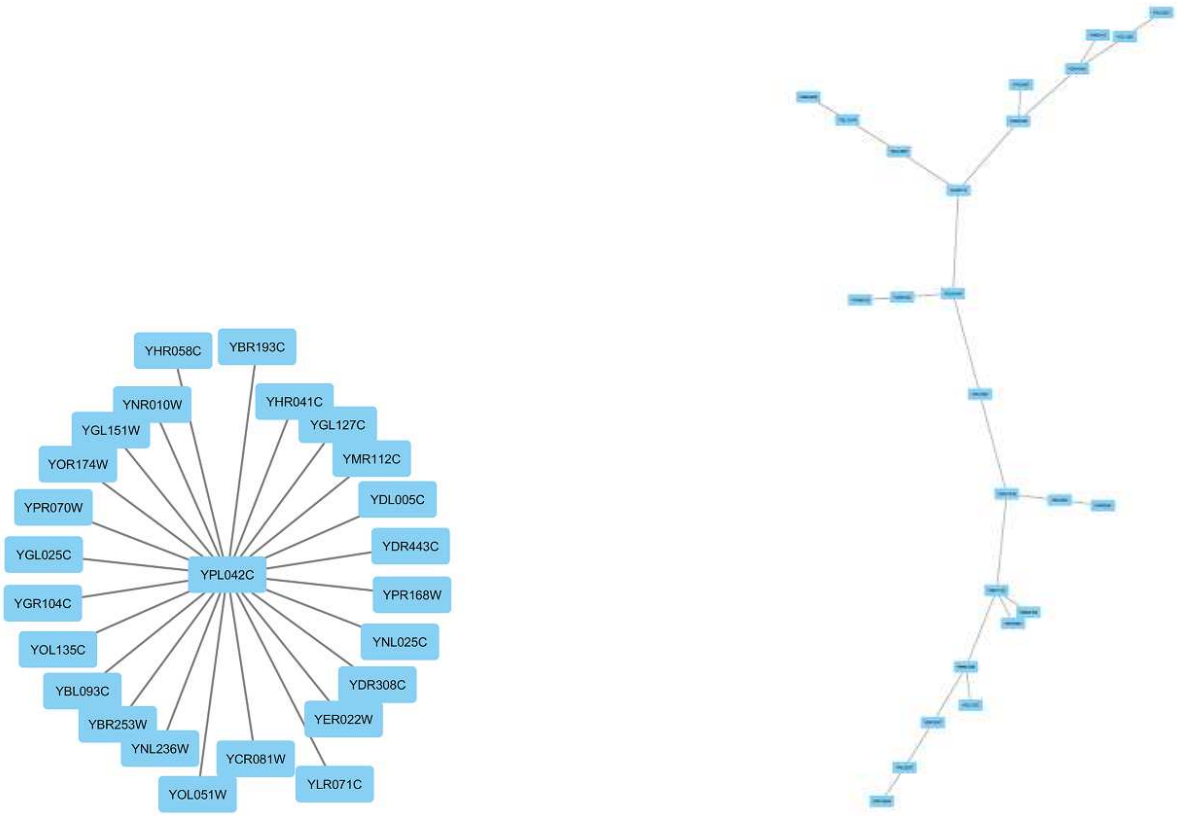
Poznato je da je za normalan rast i razvoj živog organizma veoma važno održavanje homeostaze između procesa sinteze i razgradnje proteina. Do poremećaja homeostaze može doći uslijed pojave



Slika 5.11: Povezan kompleks cytoplasmic ribosomal small subunit - netežinskom verzijom algoritma (lijevo) i težinskom verzijom algoritma (desno)



Slika 5.12: Povezan kompleks mitochondrial ribosomal large subunit - netežinskom verzijom algoritma (lijevo) i težinskom verzijom algoritma (desno)



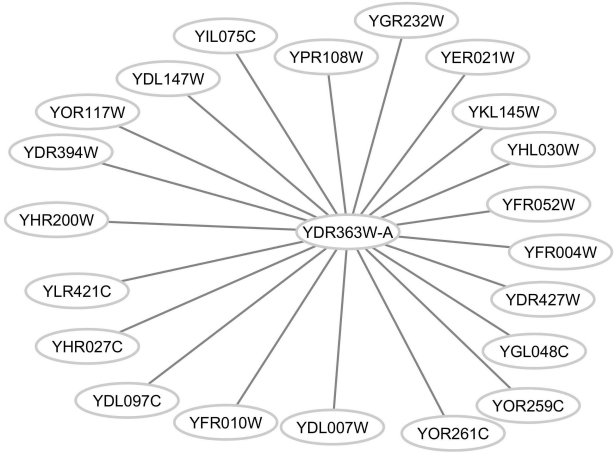
Slika 5.13: Povezan kompleks Kornberg's mediator (SRB) - netežinskom verzijom algoritma (lijevo) i težinskom verzijom algoritma (desno)

različitih bolesti, stresa, starosti ćelije itd. Na Slici 5.15 se mogu vidjeti proteinske interakcije tokom procesa razgradnje proteina ubikvitinskim putem. Ubikvitin je protein koji se nalazi u svim eukariotskim ćelijama i sastoji iz 76 aminokiselinskih ostataka. Uloga ubikvitina je da se kroz proces ubikvitacije veže za druge proteine i pri tom utiče na njihovu funkciju, lokaciju i promet ili ih potpuno degradira pomoću 26 proteozoma [24]. Proces razgradnje proteina ubikvitinskim putem se sastoji od tri koraka: (i) prepoznavanje ciljnog proteina preko specifičnih signala, (ii) modifikacija ciljnog proteina i vezivanje za ubikvintin i (iii) isporuka ciljnog proteina 26S proteozomu [45]. 26S proteozom je složen proteinski kompleks koji ima funkciju razradnje proteinubikvitin konjugata. Sastoji se od proteolitičkog jezgra -20S subjedinice (engl. proteolytic core particle (20S-CP)) i 19S subjedinice koja ima funkciju regulatorne čestice (engl. regulatory particles (19S-RPs))[82].

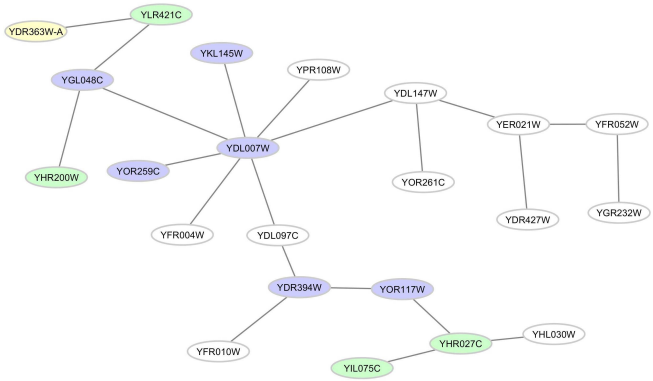
Struktura, lokalizacija i uređenost 20S-CP subjedinice je već decenijama razjašnjena. Međutim, o načinu organizacije 19S subjedinice se još uvijek malo toga zna [157].

Sa Slike 5.15 se vidi da je u osnovi 19S regulatornog kompleksa heteroheksamerni prsten koji čini 6 ATP-aza (RPT 1-6) obojenih ljubičastom bojom: YKL145W, YDL007W, YDR394W, YOR259C, YOR117W, YGL048C. RPT2 gen (YDL007W) ima centralnu ulogu, dok su ostalih pet ATP-aza povezani direktno (YKL145W, YOR259C i YGL048C) ili indirektno (YDR394W i YOR117W) preko vezujućeg proteina YDL097C (RPN6). Ostale osnovne komponente koje omogućavaju ovaj proces su vezujući proteini RPN1 (YHR027C) i RPN2 (YIL075C), obojeni svijetlo zelenom bojom, i ubikvitinski receptori RPN 10 (YHR200W) i RPN 13 (YLR421C).

Ovaj dio 19S subjedinice, u kojoj su proteini povezani prema težinskoj varijanti VNS algoritma, je u direktnoj vezi sa 20S-CP i reguliše pravilnu razgradnju proteina koji su obilježeni ubikvitinom,



Slika 5.14: Povezan 19S kompleks u netežinskom slučaju



Slika 5.15: Povezan 19S kompleks u težinskom slučaju

te na taj način sprečava nagomilavanje nerazgrađenih i pogrešno vezanih proteina [31]. Akumulacija nerazgrađenih i pogrešno spojenih proteina može dovesti do različitih neurodegenerativnih oboljenja [114, 156, 19], poput Alchajmerove, Hantingtonove, Parkinsonove bolesti. Međutim, još uvijek nije potpuno razjašnjeno kako PPI utiču na pojavu ovih disfunkcionalnosti i oboljenja [12, 161], te bi nove informacije o tim interakcijama mogle biti od velike koristi za buduća istraživanja.

Kao rezultat netežinskog algoritma dobijena je struktura prikazana na Slici 5.14, gdje SEM1 gen (protein YDR363W-A) ima centralnu ulogu i veže preostale proteine, njih 21, za sebe. Ova struktura se teško može objasniti jer još uvijek nije dokazano da protein za koji kodira SEM1 gen ima tako jake veze sa ostalim genima iz kompleksa. Međutim, poznato je da ovaj protein ima ulogu u ubikvitinskoj proteolizi i da je uključen u održavanje stabilnosti proteazoma, ali o povezanosti sa ostalim proteinima u kompleksu 19-20S još uvijek nema dovoljno informacija. Na Slici 5.15 se može vidjeti da je protein YDR363W-A, koji je obojen svijetlo žutom bojom, povezan sa ATP-aznim proteinom YGL048C preko RPN13 (YLR421C), što može da predstavlja mnogo korisniju informaciju za objašnjenje strukture 19S subjedinice. Ovo saznanje je u skladu sa rezultatima iz [18]. Preciznije, u [18] je navedeno da SEM1 može da formira podkompleks sa RPN 3 (YER021W), RPN 13 (YLR421C) ili RPN 7 (YPR108W) proteozomskim podjedinicama, ali precizna funkcija u 26S proteozomu još uvijek nije poznata. Ova analiza ukazuje na neke od uočenih asocijacija SEM1 gena sa drugim genima, međutim zbog malog broja informacija o ovom genu ne može se sa sigurnošću

Tabela 5.7: Rezultati nad realnim težinskim PPI mrežama

instanca	#PPI	Greedy			VNS				
		res.	GObj	t[s]	best	avg	bestObj.	avgObj.	t[s]
BioGRID	16180	67	205	1791.905	67	67	205	205	1182.0763
wiphi	12685	93	363.9	1728.269	93	93	363.9	363.9	624.6206
iRefIndex	22429	86	191.6	1354.977	86	86	191.6	191.6	1200.3126
MINT	5053	427	1631.8	4932.602	427	427	1631.8	1631.8	614.3754
Collins2007	6392	466	2172.4	8167.677	466	466	2172.4	2172.4	335.4952
Gavin2006	4031	625	2714.9	7498.768	625	625	2714.9	2714.9	475.2087
Krogan2006_core	3169	587	2434.8	6556.267	587	587	2434.8	2434.8	540.1228
Krogan2006_extended	4714	563	2312.2	6145.011	563	563	2312.2	2312.2	553.0909
CYC	0	1344	5938.1	12159.075	1344	1344.3	5938.1	5939.66	549.7351

tvrditi da li se radi o direktnim ili indirektnim korelacijama.

Na osnovu svega navedenog, dolazimo do zaključka da nove interakcije koje su rezultat VNS algoritma povezuju proteine na način koji je biološki opravdan. Sve to može poslužiti kao osnova za predviđanje interakcija između proteina iz različitih kompleksa sa proteinima iz 19S subjednince. Na taj način se mogu dobiti nove informacije o organizaciji i funkciji različitih proteinskih grupa unutar 19S subjedinici.

Rješavanje MinPPI problema na težinskim instancama

U skladu sa pristupom opisanim na početku Odjeljka 5.4.3, za svaki par proteina koji se pojavljuju u PPI mrežama izračunata je težina (ACS vrijednost), zatim određena i inverzna težina na osnovu formule 5.3 i primijenjen VNS algoritam. Pored toga, pohlepni algoritam iz [7, 109] je prilagođen da radi sa težinskim instancama. Tabela 5.7 sadrži rezultate dobijene nad težinskim instancama i organizovana je slično kao i Tabela 5.6, s tim da ima tri dodatne kolone: kolonu GObj koja sadrži vrijednost funkcije cilja za pohlepni algoritam, dok su u kolonama bestObj i avgObj prikazane najbolja i prosječna vrijednost funkcije cilja VNS algoritma.

Upoređivanjem rezultata iz Tabele 5.7 sa rezultatima dobijenim netežinskom varijantom algoritma (Tabela 5.6) može se doći do nekoliko zaključaka. Oba algoritma, VNS i pohlepni algoritam, u obe varijante problema (težinskoj i netežinskoj) kao rezultat vraćaju isti broj dodatih grana, odnosno dodatih PPI. Dakle, uključivanje informacije o težinama ne utiče na broj dodatih grana, što ukazuje na stabilnost predloženog pristupa. Ako se porede vremena izvršenja netežinskog i težinskog VNS algoritma, može se primijetiti da je za 6/8 instanci vrijeme izvršenja u težinskoj varijanti do 50% manje nego u netežinskom slučaju. Na osnovu toga možemo zaključiti da uključivanje ACS vrijednosti kao težina PPI pozitivno utiče na kompletan proces, olakšavajući algoritmu pretragu za granama koje treba uključiti i pri tome vodi ka bržoj konvergenciji algoritma ka boljem rješenju. Razmatranja i upoređivanja vremena izvršenja pohlepnog i VNS algoritma vode ka sličnim zaključcima kao i u netežinskoj varijanti algoritma - VNS je opet brži od pohlepnog algoritma do jednog reda veličine. S obzirom da su najbolji i prosječni rezultati VNS algoritma isti za svih osam instanci, koje su PPI mreže, zaključujemo da je VNS stabilan nad ovim instancama jer dolazi do istih rješenja u svih 10 izvršenja.

5.4.4 Rezultati testiranja na slučajno generisanim instancama

Iako je VNS algoritam brži od pohlepnog algoritma, činjenica je da su rješenja (tj. broj dodatih grana) koja su dobijena ovim algoritmima vrlo slična i za vještačke i za realne instance. Razlika se pojavljuje jedino na dvije vještačke instance, s1data4 i s1data7, gdje je VNS algoritam nešto bolji.

Tabela 5.8: Rezultati nad slučajnim instancama

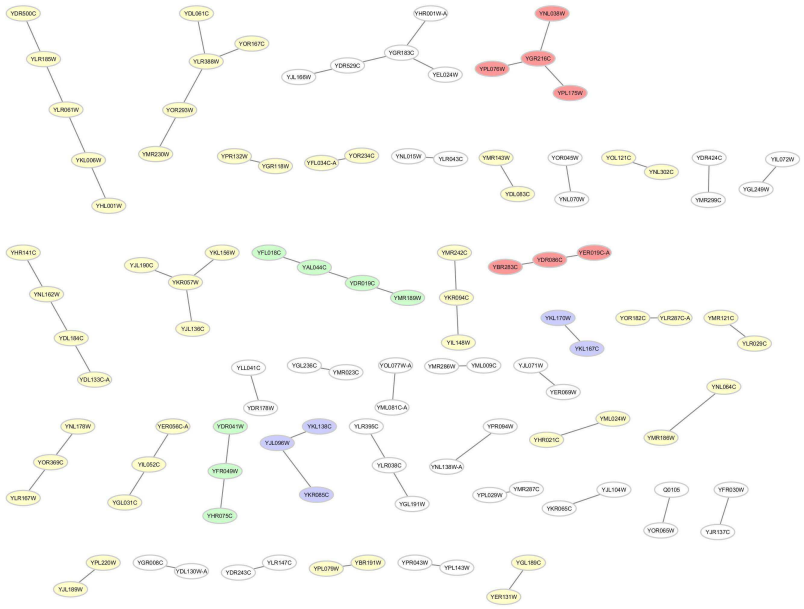
instanca	#vert	#compl	max card.	Greedy		VNS		
				res.	t[s]	best	avg	t[s]
rand01	10	10	10	11	0.002	11	11	1.3952
rand02	10	50	10	21	0.015	20	20	6.8462
rand03	10	100	10	27	0.035	25	25	14.7322
rand04	10	200	10	31	0.086	30	30	20.322
rand05	10	500	10	38	0.229	36	36	33.9923
rand06	50	50	50	110	23.255	90	92	1200.3233
rand07	50	250	50	206	296.087	177	180.3	1201.3391
rand08	50	500	50	249	791.966	227	232.3	1201.4706
rand09	50	1000	50	330	1655.871	305	311.3	1203.1761
rand10	100	100	100	281	1264.348	254	258.7	7212.2528
rand11	100	500	100	487	14283.221	480	492.6	7212.3395
rand12	100	1000	100	636	40607.598	619	629.6	21626.8131

Ova činjenica je inspirisala dalje istraživanje, u cilju pronalaska klase instanci na kojima bi VNS algoritam bio bolji od pohlepnog algoritma, ne samo po vremenu izvršenja, nego i po minimalnom broju dodatih PPI takvih da kompleksi budu povezani. U prethodnim testiranjima se može primijetiti da pohlepni algoritam ima dobre performanse na instancama u kojima je kardinalnost kompleksa relativno mala. U takvim slučajevima je kardinalnost presjeka između kompleksa relativno mala i pohlepni algoritam uglavnom uspješno pronalazi kvalitetne rezultate. Jednostavno, u svakom koraku, pohlepni algoritam bira PPI koja će najviše uticati na smanjenje broja nepovezanih komponenti. Ako je broj takvih PPI mali, onda su šanse pohlepnog algoritma da dostigne optimalno rješenje veće. Sa druge strane, veća kardinalnost presjeka između kompleksa otežava ovu strategiju i vodi pohlepni algoritam ka suboptimalnom rješenju.

Da bi se opravdala ova pretpostavka konstruisan je novi skup slučajnih instanci, variranjem tri parametra: ukupnog broja proteina, ukupnog broja kompleksa i maksimalne kardinalnosti kompleksa. Na tako generisanim instancama razmatrana je netežinska varijanta MinPPI0 problema i primijenjena su oba algoritma, VNS i pohlepni algoritam. Za manje i srednje instance korišteni su isti parametri za VNS algoritam koji su navedeni na početku Odjeljka 5.4. Za veće instance povećano je ukupno vrijeme izvršenja i postavljeno na dva sata (za instance rand10 i rand11), odnosno šest sati za najveću instancu (rand12). Dodatno, za instance koje imaju po 100 proteina uvećana je i kardinalnost okoline koja se razmatra, odnosno k_{max} je postavljeno na 20.

Rezultati su prikazana u Tabeli 5.8. Prve četiri kolone sadrže naziv instance, ukupan broj proteina, ukupan broj kompleksa i maksimalnu kardinalnost kompleksa, respektivno. Rezultati i vrijeme izvršenja (u sekundama) dobijeni pohlepnim algoritmom su prikazani u sljedeće dvije kolone. Ostatak tabele sadrži informacije o najboljem i prosječnom rezultatu, kao i prosječnom vremenu izvršenja (u sekundama) VNS algoritma u 10 izvršenja.

Iz Tabele 5.8 se može vidjeti da je VNS algoritam bolji od pohlepnog algoritma po kvalitetu rezultata. Ukupan broj dodatih PPI VNS algoritmom je u 11/12 slučajeva manji nego broj dodatih PPI pohlepnim algoritmom. Iako je vrijeme izvršenja pohlepnog algoritma za instance sa manjim brojem čvorova i kompleksa manje nego vrijeme izvršenja VNS algoritma, može se primijetiti da su rezultati dobijeni VNS algoritmom bolji u smislu broja dodatih grana. Za veće instance, posebno instance rand11 i rand12, VNS algoritam je bolji i u pogledu rezultata i vremena izvršenja.



Slika 5.16: Grane dodate u BioGRID mrežu VNS algoritmom

5.5 Identifikacija značajnih grupa proteina

Kroz ovaj odjeljak biće validirana korisnost predloženog pristupa i data biološka interpretacija dobijenih rezultata algoritma u tri faze nad BioGRID PPI mrežom.

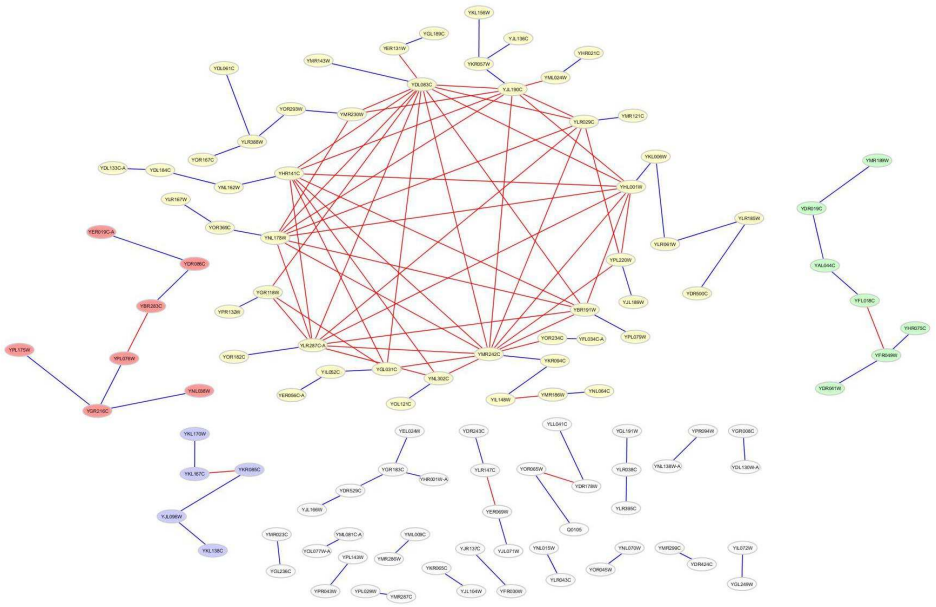
Faza I

U prethodnom odjeljku su prikazani rezultati dobijeni primjenom VNS algoritma na različite PPI mreže, što je prva faza predložene metodologije. Dalje se nastavlja formiranje značajne grupe proteina dodavanjem novih proteina baznim grupama. Da bi se pokazala korisnost predloženog pristupa, kompletna metodologija je primjenjena na BioGRID PPI mrežu. Originalna BioGRID mreža sadrži 5641 protein i 59748 PPI. Kao što je i navedeno u Odjeljku 5.4.2, originalna mreža je redukovana tako da sadrži samo proteine koji pripadaju CYC2008 standardu. Na redukovanu BioGRID mrežu je primjenjen VNS algoritam.

Kao što je i prikazano u Tabelama 5.6 i 5.7, da bi povezao proteinske komplekse iz CYC2008 standarda VNS algoritam u redukovanu BioGRID mrežu dodaje novih 67 interakcija. Tih 67 interakcija, odnosno grana, povezuje 110 proteina. Na Slici 5.16, prikazano je tih 110 proteina povezanih novim PPI. Proteini koji će formirati veće grupe proteina nakon dodavanja postojećih PPI su obojeni istom bojom. Proteini koji nisu obojeni dalje ne formiraju veće proteinske grupe, pa neće biti ni razmatrani. Nakon primjene VNS algoritma, da bi se dobile nove interakcije, u nastavku istraživanja se ponovo koristi originalna mreža.

Faza II

Svi ovi proteini se dalje razmatraju u drugoj fazi algoritma, u kojoj se dodaju grane koje postoje između tih 110 proteina u originalnoj BioGRID mreži. Na Slici 5.17 je prikazana ista grupa proteina kao i na Slici 5.16, ali sada povezna sa dva različita tipa PPI: nove PPI (obojene plavom bojom) dodate VNS algoritmom i PPI (obojene crvenom bojom) koje postoje u originalnoj BioGRID mreži. Kao što se može primijetiti sa Slike 5.17, nakon što su u razmatranje uključene obje vrste PPI, dobijeno je nekoliko većih i nekoliko manjih grupa proteina. Jedna od većih grupa proteina se sastoji od 49 proteina (obojenih svijetlo žutom bojom) u čijem jezgru je 14 proteina koji pripadaju



Slika 5.17: Bazne grupe BioGRID mreže koje su formirane nakon druge faze

ili kompleksu "Mala citoplazmatska ribozomalna subjedinica" (engl. cytoplasmic ribosomal small subunit complex) ili kompleksu "Velika citoplazmatska ribozomalna subjedinica" (engl. cytoplasmic ribosomal large subunit complex). Preostali proteini iz ove grupe pripadaju *High-molecular-weight complex - HMC* kompleksu. U daljem tekstu ova grupa proteina će biti označena sa $G1$. Pored ove grupe proteina dalje će biti razmtrane još tri manje grupe proteina: $G2$ grupa, koja se sastoji od 7 proteina (obojenih ljubičastom bojom) koji pripadaju *Ssh1p translocon* kompleksu ili *glycosylphosphatidylinositol-N-acetylglucosaminyltransferase (GPI-GnT)* kompleksu, $G3$ grupa koja se sastoji od 7 proteina (obojenih svijetlo zelenom bojom) koji pripadaju *glycine cleavage* kompleksu ili kompleksu "Mala mitohondrijalna ribozomalna subjedinica" (engl. mitochondrial ribosomal small subunit complex), $G4$ grupa od 5 proteina (obojenih svjetlo plavom bojom) koji pripadaju kompleksu "Velika mitohondrijalna ribozomalna subjedinica" (engl. mitochondrial ribosomal large subunit complex).

Faza III

Četiri grupe proteina identifikovane u drugoj fazi se dodatno proširuju u trećoj fazi na sljedeći način. Svakoj grupi se dodaju proteini koji se nalaze u originalnoj BioGRID PPI mreži, ali ne pripadaju nijednom kompleksu iz CYC2008 standarda i pri tome zadovoljavaju sljedeća dva uslova:

(i) imaju indirektnu interakciju sa svim proteinima iz bazne grupe, tj. nalaze se na rastojanju 2 od svakog proteina iz grupe;

(ii) prosječna vrijednost genske ko-ekspresije (ACS vrijednost) sa proteinima iz bazne grupe je veća od praga, čija je vrijednost 1.8.

Ovakav izbor vrijednosti za prag je motivisan činjenicom da bi ta vrijednost trebala biti veća od prosječne vrijednosti genske ko-ekspresije u cijeloj mreži (što je oko 1.4), ali opet dovoljno mala da omogući dodavanje većeg broja proteina u grupu.

Na ovaj način formirane su četiri proširene grupe (Proširene grupe 1-4) koje sadrže nekoliko stotina proteina, koji su rijetko povezani u originalnoj BioGRID PPI mreži. Da bi se pokazala

značajnost dobijenih proširenih grupa, razmatrane su alatom za obogaćivanje informacijama DAVID [40]. DAVID (the Database for Annotation, Visualization and Integrated Discovery) alat omogućava izvlačenje bioloških karakteristika ili biološkog značenja povezanih sa datom listom gena. Od nekoliko modula koji su dostupni unutar DAVID alata, korišteno je funkcionalno anotacijsko klasterovanje (engl. Functional Annotation Clustering), koje klasteruje funkcionalno slične termine povezane sa ulaznom listom gena u grupe, primjenjujući analizu obogaćivanja "koja je usmjerena na termine" (engl. term centric modular enrichment analysis) [66].

U Tabeli 5.9 su prikazani rezultati dobijeni DAVID alatom za obogaćivanje informacijama. Za svaku od četiri proširene grupe proteina prikazane su sljedeće informacije. Kolone #Prot., #ePPI, #nPPI, respektivno, sadrže informaciju o broju proteina u grupama G1-G4 nakon Faze II, broj postojećih PPI između ovih proteina u originalnoj BioGRID mreži i broj dodatih PPI VNS algoritmom. Kolona Kompleksi sadrži listu kompleksa kojima pripadaju proteini grupisani u drugoj fazi. Kolone #aProt. i #tProt. sadrže broj proteina dodatih nakon Faze III i ukupan broj proteina u svakoj od proširenih grupa, respektivno. Za svaku od proširenih grupa, prikazan je anotacijski klaster sa najvećim skorom obogaćenja, kategorija termina i određen termin na osnovu kojeg su proteini grupisani u isti anotacijski klaster. Dodatno je prikazana i informacija o ukupnom broju proteina koji imaju sličan termin, kao i informacija o p-vrijednosti.

Kao što se i može vidjeti iz Tabele 5.9, vrijednosti skorova obogaćenja su prilično visoki, dok su p-vrijednosti niske za svaki termin. Visoke vrijednosti skorova za obogaćivanje, nad dobijenim anotacijskim klasterima, ukazuju na to da postoji značajna funkcionalna sličnost između proteina grupisanih u isti klaster. Ova činjenica bi mogla biti dobra polazna osnova za dalju identifikaciju proteina sa sličnim anotacijama.

Zbog dalje validacije predloženog metoda sa biološke tačke gledišta, razmatra se antotacijski klaster koji je dobijen za Proširenu grupu 2 i koji ima visok skor obogaćenja. Posmatranjem dobijenih rezultata može se zaključiti da su proteini klasterovani na osnovu funkcije, ćelijskog kompartmenta i ključnih riječi koje ih opisuju. U svakom klasteru postoji više međusobno povezanih podklastera, pa se većina gena pojavljuje u više od jednog podklastera. Na primjer, protein YDR091C se pojavljuje u podklasteru koji je određen na osnovu ćelijskog kompartmenta - *preribosome, large subunit precursor*, podklasteru u kojem su geni grupisani na osnovu ključnih riječi - *Ribosome biogenesis* i još jednom podklasteru koji je takođe nastao grupisanjem na osnovu ključnih riječi - *rRNA processing*. Sa biološke tačke gledišta, ovo ima smisla, zato što je velika ribozomalna subjedinica sastavni dio ribozoma.

5.6 Završna razmatranja

U ovom poglavlju je predstavljena metoda koja u tri faze identifikuje značajne grupe proteina u PPI mrežama. Počevši od slabo povezanih proteina iz proteinskih kompleksa, u prvoj fazi dodaju se nove PPI tako da svaki proteinski kompleks bude povezana struktura. VNS algoritam, formiran za rješavanje odgovarajućeg matematičkog optimizacionog problema, dodaje što je moguće manje grana da podrži svaki proteinski kompleks i po prvi put u literaturi je razmatran nad težinskom PPI mrežom. Pristup uvođenja PPI težina, na osnovu vrijednosti genske ko-ekspresije za odgovarajuće gene, usmjerava pretragu u oblasti koje su više obećavajuće, poboljšava performanse algoritma i vodi ka biološki smislenijim rješenjima. U drugoj fazi, povezuju se proteini iz različitih kompleksa koristeći novododate PPI i postojeće PPI iz razmatrane PPI mreže i na taj način formira veće grupe proteina.

I konačno, u trećoj fazi algoritam traži nove proteine koji su u indirektnoj vezi sa proteinima iz bazne grupe i imaju relativno visoke vrijednosti genske ko-ekspresije sa tim proteinima. Na taj način identifikovane su velike grupe koje imaju nekoliko stotina proteina i koje se dalje analiziraju alatima za obogaćivanje informacijama. Prva faza, koja je računarski najzahtjevnija, je riješena je pomoću VNS metaheurističkog metoda. U predloženom VNS-u su implemenitrane nove procedure koje omogućavaju efikasno izvršenje VNS metode. Rezultati testiranja, dobijeni nad sintetičkim i stvarnim PPI mrežama, jasno pokazuju da predložena metoda ima bolje performanse od postojećih metoda iz literature kako u pogledu potrebnog vremena za izvršenje, tako i u pogledu kvaliteta dobijenih rezultata.

Sa biološke tačke gledišta, urađena je analiza dobijenih rezultata pomoću DAVID alata za obogaćivanje informacijama. Rezultati analize pokazuju da ovako identifikovane grupe proteina imaju visoku statističku značajnost, sa skorom obogaćenja većim od 17. Ta činjenica ukazuje da postoji značajna funkcionalna sličnost između grupisanih proteina.

Ovo istraživanje se može proširiti na nekoliko načina. Visoki skorovi obogaćenja za identifikovane grupe proteina ukazuju da se predložena metoda može dalje koristiti za razvijanje metoda koje se bave predviđanjem PPI. Kao poboljšanje predložene metode, u slučaju težinskog MinPPI problema težine u PPI mreži mogu biti neke druge biološke informacije, poput sličnosti GO termina. Pored toga, predloženi VNS algoritam se može primijeniti na druge biološke mreže, kao i na mreže iz drugih oblasti.

Tabela 5.9: Rezultati dobijeni DAVID alatom za obogaćivanje informacijama

Grupa proteina	#Prot.	#ePPI	#nPPI	Kompleksi	#aProt.	#tProt.	Rezultati DAVID analize obogaćivanja informacijama			
							Anotacijski klaster	**Skor obogaćivanja: 17.020**	br. prot.	p-vrijednost
Proširena grupa 1	49	49	31	cytoplasmic ribosomal small subunit, cytoplasmic ribosomal large subunit, HMC Complex	481	530	GOTERM_BP_DIRECT	GO:0042254 ~ribosome biogenesis	63	2.4E-19
							UP_KEYWORDS	Ribosome biogenesis	58	4.6E-19
							GOTERM_CC_DIRECT	GO:0005730 ~nucleolus	72	2.6E-18
							GOTERM_BP_DIRECT	GO:0006364~rRNA processing	60	1.6E-16
							UP_KEYWORDS	rRNA processing	52	1.7E-15
							Anotacijski klaster	**Skor obogaćivanja: 19.007**	br. prot.	p-vrijednost
Proširena grupa 2	7	1	5	glycosylphosphatidy-linositol-N-acetylglucosaminyl-transferase (GPI-GnT) complex, Ssh1p translocon complex	219	226	GOTERM_CC_DIRECT	GO:0030687~preribosome, large subunit precursor	35	8.3E-26
							GOTERM_CC_DIRECT	GO:0005730~nucleolus	55	1.7E-25
							UP_KEYWORDS	Ribosome biogenesis	45	9.6E-25
							GOTERM_BP_DIRECT	GO:0042254~ribosome biogenesis	46	4.2E-23
							GOTERM_BP_DIRECT	GO:0006364~rRNA processing	41	5.1E-18
							UP_KEYWORDS	rRNA processing	34	4.4E-15
							UP_KEYWORDS	Nucleus	97	7.2E-6
							Anotacijski klaster	**Skor obogaćivanja: 17.456**	br. prot.	p-vrijednost
Proširena grupa 3	7	1	5	glycine cleavage complex, mitochondrial ribosomal small subunit	485	492	UP_KEYWORDS	Mitochondrion	147	4.6E-25
							UP_SEQ_FEATURE	transit peptide:Mitochondrion	72	5.9E-15
							UP_KEYWORDS	Transit peptide	77	1.6E-14
							Anotacijski klaster	**Skor obogaćivanja: 32.228**	br. prot.	p-vrijednost
Proširena grupa 4	5	1	3	mitochondrial ribosomal large subunit	279	284	UP_KEYWORDS	Mitochondrion	130	5.6E-44
							GOTERM_CC_DIRECT	GO:0005739~mitochondrion	147	1.3E-34
							UP_KEYWORDS	Transit peptide	75	1.8E-28
							UP_SEQ_FEATURE	transit peptide:Mitochondrion	66	8.9E-25

Glava 6

Zaključak

Problemi koji se rješavaju u ovoj disertaciji pripadaju trenutno vrlo aktuelnim oblastima bioinformatike i računarske biologije. U ovoj disertaciji su razmatrani i rješavani problemi: particionisanje rijetkih bioloških mreža na *k-plex* podmreže (Max-EkP problem), predviđanje uloge metabolita u metaboličkim reakcijama, particionisanje bioloških mreža na visoko povezane komponente (HCD problem) i problem identifikacije značajnih grupa proteina dodavanjem novih grana u težinsku PPI mrežu. Dio posljednjeg navedenog problema je NP težak problem dodavanja minimalnog broja grana da bi određeni podgrafovi postali povezani (MinPPI problem).

Sa računarskog aspekta, problemi Max-EkP, HCD i MinPPI pripadaju klasi grafovskih NP teških problema i rješavani su metodama kombinatorne optimizacije, preciznije metodom promjenljivih okolina. Za problem predviđanja uloge metabolita u metaboličkim reakcijama predloženo rješenje koje se zasniva na metodi statističkog modeliranja uslovnim slučajnim poljima.

Metoda promjenljivih okolina za Max-EkP problem koristi cjelobrojno kodiranje po čvorovima. U sklopu ove metode, kreirana je nova funkcije cilja i njeno parcijalno izračunavanje zbog ubrzanja algoritma. Funkcija cilja uzima u obzir stepen svakog čvora u svakom *k-plex*-u i favorizuje dopustiva rešenja, uz dozvoljavanje postepenog porasta vrijednosti same funkcije u slučaju blago nedopustivih rješenja. *k-plex* strukture dobijene pri rješavanju Max-EkP problema predstavljaju važne metaboličke procese organizma, poput sinteze masnih kiselina, procesa degradacije aminokiselina, sinteze vitamina B6 i sl.

Metoda zasnovana na uslovnim slučajnim poljima je primijenjena za sekvencijalno predviđanje uloga metabolita u metaboličkim reakcijama. Za primjenu ove metode formirana su tri različita skupa karakterističnih funkcija koje uključuju informacije o elementima u najbližem okruženju i informacije o njihovim oznakama.

Za rješavanje HCD problema prije primjene metode promjenljivih okolina primijenjena je faza pretprocesiranja. U fazi pretprocesiranja se brišu grane koje zadovoljavaju pravilo da povezuju čvorove koji nemaju zajedničkih susjeda. Slično kao i za Max-EkP problem, metoda promjenljivih okolina koja je predložena za HCD problem koristi cjelobrojno kodiranje po čvorovima. Takođe, funkcija cilja za HCD problem favorizuje dopustiva rješenja, po sličnom principu kao i za Max-EkP problem. Pored standardnih procedura razmrdavanja i lokalne pretrage, koje se ponavljaju u cilju dobijanja što kvalitetnijih rješenja, za HCD problem je formirana dodatna procedura spajanja komponenti koja za cilj ima zadržavanje što većeg broja grana. Visoko povezane komponente PPI mreža, koje su rezulatat rješavanja HCD problema, grupišu proteine sa sličnim biološkim funkcijama, a visoko povezane komponente metaboličkih mreža, slično kao i *k-plex* strukture, predstavljaju važne

metaboličke procese.

Metodom u tri faze, koja je predložena u petom poglavlju ove disertacije, identifikuju se značajne grupe proteina koji u postojećim PPI mrežama nisu povezani ili su povezani malim brojem grana. Prva faza je zasnovana na metodi promjenljivih okolina, dok se u drugoj i trećoj fazi koriste dodatne informacije o postojećim PPI u početnoj mreži, kao i informacije o genskoj ko-ekspresiji i indirektnim interakcijama. Metoda promjenljivih okolina za rješavanje MinPPI problema koristi binarno kodiranje po granama, kao i parcijalno računanje funkcije cilja zbog smanjenja vremena potrebnog za izvršenje algoritma. Formirana je dodatna procedura, nazvana `FixGreedy`, koja pokušava da popravi nedopustiva rješenja koja su dobijena procedurom razmrdavanja. Pored rješavanja problema dodavanja minimalnog broja grana u netežinsku PPI mrežu tako da poznati proteinski kompleksi postanu povezani, u ovom istraživanju je razmatran i problem dodavanja grana u težinsku PPI mrežu sa istim ciljem. Dobijeni rezultati su pokazali da uključivanje informacija o težinama na realnim biološkim mrežama usmjerava pretragu u perspektivnije oblasti, poboljšava performanse algoritma i daje biološki smislenija rješenja.

Particionisanje i grupisanje elemenata u biološkim mrežama, na način prikazan u ovoj disertaciji, predstavljaju novi pristup za potvrdu postojećih i dobijanje novih informacija o nekim biološkim strukturama i njihovim međusobnim vezama, te daju značajan naučni doprinos u oblasti bioinformatike i računarske biologije.

6.1 Naučni doprinos rada

Najvažniji rezultati koji predstavljaju naučni doprinos ovog rada su:

- Rješavanje NP teškog problema particionisanja grafa u *k-plex* strukture, koji je primjenjen na particionisanje bioloških mreža. Razvijeni algoritam je zasnovan na metodi promjenljivih okolina. Predložena metoda je primijenjena na metaboličke biološke mreže i predstavlja novi pristup u otkrivanju novih informacija u biološkim strukturama.

- Razvoj metode za predviđanje uloge metabolita u metaboličkim reakcijama. Metoda je zasnovana na uslovnim slučajnim poljima i predstavlja novi pristup za rješavanje problema predviđanja u oblasti bioloških nauka.

- Rješavanje NP teškog problema brisanja grana uz očuvanje visoke povezanosti. Predložena metoda je primijenjena na proteinske i metaboličke mreže i predstavlja novi pristup u analizi odnosa u biološkim strukturama.

- Razvoj novog pristupa za identifikovanje značajnih grupa proteina u PPI mreži. Proučavani problem je ekvivalentan matematičkom NP teškom problemu rekonstrukcije mreže. Pristup uključuje dodavanje novih protein-protein interakcija u postojeću mrežu i grupisanje proteina kombinovanjem bioloških informacija iz različitih izvora (PPI mreže, genska ko-ekspresija i alati za obogaćivanje informacijama). Identifikovanje grupa proteina na opisani način predstavlja novi pristup u razumijevanju unutrašnjih struktura i funkcija bioloških podataka.

Literatura

[1] N. Abovich, P. Legrain, and M. Rosbash. "The yeast PRP6 gene encodes a U4/U6 small nuclear ribonucleoprotein particle (snRNP) protein, and the PRP9 gene encodes a protein required for U2 snRNP binding." *Molecular and Cellular Biology* 10(12) (1990), pp. 6417–6425.

[2] T. Akutsu, S. Miyano, and S. Kuhara. "Algorithms for identifying Boolean networks and related biological networks based on matrix multiplication and fingerprint function". *Journal of Computational Biology* 7(3-4) (2000), pp. 331–343.

[3] T. Akutsu, M. Hayashida, D. Bahadur KC, E. Tomita, J. Suzuki, and K. Horimoto. "Dynamic programming and clique based approaches for protein threading with profiles and constraints". *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* 89(5) (2006), pp. 1215–1222.

[4] G. Alanis-Lobato, M. A. Andrade-Navarro, and M. H. Schaefer. "HIPPIE v2. 0: enhancing meaningfulness and reliability of protein–protein interaction networks". *Nucleic Acids Research* 45(1) (2016), pp. 408–414.

[5] R. Albert. "Scale-free networks in cell biology". *Journal of Cell Science* 118(21) (2005), pp. 4947–4957.

[6] U. Alon. "Biological networks: the tinkerer as an engineer". *Science* 301(5641) (2003), pp. 1866–1867.

[7] D. Angluin, J. Aspnes, and L. Reyzin. "Network construction with subgraph connectivity constraints". *Journal of Combinatorial Optimization* 29(2) (2015), pp. 418–432.

[8] V. M. Anoop, U. Basu, M. T. McCammon, L. McAlister-Henn, and G. J. Taylor. "Modulation of citrate metabolism alters aluminum tolerance in yeast and transgenic canola overexpressing a mitochondrial citrate synthase". *Plant Physiology* 132(4) (2003), pp. 2205–2217.

[9] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, et al. "Gene ontology: tool for the unification of biology". *Nature Genetics* 25(1) (2000), pp. 25–29.

[10] B. Balasundaram. "Cohesive subgroup model for graph-based text mining" in *IEEE International Conference on Automation Science and Engineering, 2008. CASE 2008.* IEEE. 2008, pp. 989–994.

[11] B. Balasundaram, S. Butenko, and I. V. Hicks. "Clique relaxations in social network analysis: The maximum k-plex problem". *Operations Research* 59(1) (2011), pp. 133–142.

[12] N. F. Bence, R. M. Sampat, and R. R. Kopito. "Impairment of the ubiquitin-proteasome system by protein aggregation". *Science* 292(5521) (2001), pp. 1552–1555.

[13] J. M. Berg, J. L. Tymoczko, and L. Stryer. *Biochemistry.* W. H. Freeman; 2002.

[14] A. Bernal, K. Crammer, A. Hatzigeorgiou, and F. Pereira. "Global discriminative learning for higher-accuracy computational gene prediction". *PLoS Computational Biology* 3(3) (2007), pp. 488–497.

[15] D. Binns, E. Dimmer, R. Huntley, D. Barrell, C. O'donovan, and R. Apweiler. "QuickGO: a web-based tool for Gene Ontology searching". *Bioinformatics* 25(22) (2009), pp. 3045–3046.

[16] P. Blunsom and T. Cohn. "Discriminative word alignment with conditional random fields" in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics. 2006, pp. 65–72.

[17] V. Boginski, S. Butenko, O. Shirokikh, S. Trukhanov, and J. G. Lafuente. "A network-based data mining approach to portfolio selection via weighted clique relaxations". *Annals of Operations Research* 216(1) (2014), pp. 23–34.

[18] S. Bohn, E. Sakata, F. Beck, G. R. Pathare, J. Schnitger, I. Nágy, W. Baumeister, and F. Förster. "Localization of the regulatory particle subunit Sem1 in the 26S proteasome". *Biochemical and Biophysical Research Communications* 435(2) (2013), pp. 250–254.

[19] E. Bossy-Wetzel, R. Schwarzenbacher, and S. A. Lipton. "Molecular pathways to neurodegeneration". *Nature Medicine* 10(7s) (2004), pp. 2–9.

[20] J. Brimberg, S. Janićijević, N. Mladenović, and D. Urošević. "Solving the clique partitioning problem as a maximally diverse grouping problem". *Optimization Letters* 11(6) (2017), pp. 1123–1135.

[21] J. Brown, D. K. Bahadur, E. Tomita, and T. Akutsu. "Multiple methods for protein side chain packing using maximum weight cliques". *Genome Informatics* 17(1) (2006), pp. 3–12.

[22] F. Browne, H. Wang, H. Zheng, and F. Azuaje. "GRIP: A web-based system for constructing Gold Standard datasets for protein-protein interaction prediction". *Source Code for Biology and Medicine* 4(1) (2009). DOI: 10.1186/1751-0473-4-2.

[23] S. Bruckner, F. Hüffner, R. M. Karp, R. Shamir, and R. Sharan. "Topology-free querying of protein interaction networks". *Journal of Computational Biology* 17(3) (2010), pp. 237–252.

[24] A. M. Burger and A. K. Seth. "The ubiquitin-mediated protein degradation pathway in cancer: therapeutic implications". *European Journal of Cancer* 40(15) (2004), pp. 2217–2229.

[25] S. Carbon, A. Ireland, C. J. Mungall, S. Shu, B. Marshall, S. Lewis, A. Hub, and W. P. W. Group. "AmiGO: online access to ontology and annotation data". *Bioinformatics* 25(2) (2008), pp. 288–289.

[26] H. Chen, W. Chung, J. J. Xu, G. Wang, Y. Qin, and M. Chau. "Crime data mining: a general framework and some examples". *Computer* 37(4) (2004), pp. 50–56.

[27] M. Chitale, I. K. Khan, and D. Kihara. "In-depth performance evaluation of PFP and ESG sequence-based function prediction methods in CAFA 2011 experiment". *BMC bioinformatics* 14(3) (2013), pp. 2–12.

[28] G. Chockler, R. Melamed, Y. Tock, and R. Vitenberg. "Constructing scalable overlays for pub-sub with many topics" in *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*. ACM. 2007, pp. 109–118.

[29] H. N. Chua, W.-K. Sung, and L. Wong. "Exploiting indirect neighbours and topological weight to predict protein function from protein–protein interactions". *Bioinformatics* 22(13) (2006), pp. 1623–1630.

[30] H. N. Chua, K. Ning, W.-K. Sung, H. W. Leong, and L. Wong. "Using indirect protein–protein interactions for protein complex prediction". *Journal of Bioinformatics and Computational Biology* 6(03) (2008), pp. 435–466.

[31] A. Ciechanover and K. Iwai. "The ubiquitin system: from basic mechanisms to the patient bed". *IUBMB life* 56(4) (2004), pp. 193–201.

[32] T. Cohn and P. Blunsom. "Semantic role labelling with tree conditional random fields" in *Proceedings of the Ninth Conference on Computational Natural Language Learning*. Association for Computational Linguistics. 2005, pp. 169–172.

[33] S. R. Collins, P. Kemmeren, X.-C. Zhao, J. F. Greenblatt, F. Spencer, F. C. Holstege, J. S. Weissman, and N. J. Krogan. "Toward a comprehensive atlas of the physical interactome of Saccharomyces cerevisiae". *Molecular & Cellular Proteomics* 6(3) (2007), pp. 439–450.

[34] G. O. Consortium. "The gene ontology project in 2008". *Nucleic Acids Research* 36(suppl_1) (2007), pp. D440–D444.

[35] G. O. Consortium. "The gene ontology resource: 20 years and still GOing strong". *Nucleic Acids Research* 47(D1) (2018), pp. D330–D338.

[36] L. Cowen, W. Goddard, and C. E. Jesurum. "Defective coloring revisited". *Journal of Graph Theory* 24(3) (1997), pp. 205–219.

[37] R. Davidović, B. Gemović, N. Veljković, and V. Perović. "DiNGO: stand-alone application for GO and HPO term enrichment" in *Belgrade BioInformatics Conference 2018, Biologia Serbica*. 2018, p. 70.

[38] J. De Las Rivas and C. Fontanillo. "Protein–protein interactions essentials: key concepts to building and analyzing interactome networks". *PLoS Computational Biology* 6(6) (2010), pp. 1–8.

[39] D. DeCaprio, J. P. Vinson, M. D. Pearson, P. Montgomery, M. Doherty, and J. E. Galagan. "Conrad: gene prediction using conditional random fields". *Genome Research* 17(9) (2007), pp. 1389–1398.

[40] G. Dennis, B. T. Sherman, D. A. Hosack, J. Yang, W. Gao, H. C. Lane, and R. A. Lempicki. "DAVID: database for annotation, visualization, and integrated discovery". *Genome Biology* 4(9) (2003). DOI: 10.1186/gb-2003-4-9-r60.

[41] Y. Dong, Y. Sun, and C. Qin. "Predicting protein complexes using a supervised learning method combined with local structural information". *PloS one* 13(3) (2018), pp. 1–23.

[42] U. Dorndorf and E. Pesch. "Fast clustering algorithms". *ORSA Journal on Computing* 6(2) (1994), pp. 141–153.

[43] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

[44] P. Fabrizio, J. Dannenberg, P. Dube, B. Kastner, H. Stark, H. Urlaub, and R. Lührmann. "The evolutionarily conserved core design of the catalytic activation step of the yeast spliceosome". *Molecular Cell* 36(4) (2009), pp. 593–608.

[45] D. Finley. "Recognition and processing of ubiquitin-protein conjugates by the proteasome". *Annual Review of Biochemistry* 78 (2009), pp. 477–513.

[46] J. Förster, I. Famili, P. Fu, B. Ø. Palsson, and J. Nielsen. "Genome-scale reconstruction of the Saccharomyces cerevisiae metabolic network". *Genome Research* 13(2) (2003), pp. 244–253.

[47] A. Franceschini, D. Szklarczyk, S. Frankild, M. Kuhn, M. Simonović, A. Roth, J. Lin, P. Minguez, P. Bork, C. Von Mering, et al. "STRING v9. 1: protein-protein interaction networks, with increased coverage and integration". *Nucleic Acids Research* 41(D1) (2012), pp. D808–D815.

[48] J. Gagneur, R. Krause, T. Bouwmeester, and G. Casari. "Modular decomposition of protein-protein interaction networks". *Genome Biology* 5(8) (2004). DOI: 10.1186/gb-2004-5-8-r57.

[49] A.-C. Gavin, P. Aloy, P. Grandi, R. Krause, M. Boesche, M. Marzioch, C. Rau, L. J. Jensen, S. Bastuck, B. Dümpelfeld, et al. "Proteome survey reveals modularity of the yeast cell machinery". *Nature* 440(7084) (2006), pp. 631–636.

[50] D. Gomez-Cabrero, I. Abugessaisa, D. Maier, A. Teschendorff, M. Merkenschlager, A. Gisel, E. Ballestar, E. Bongcam-Rudloff, A. Conesa, and J. Tegnér. "Data integration in the era of omics: current and future challenges". *BMC Systems Biology* 8(l1) (2014). DOI: 10.1186/1752-0509-8-S2-I1.

[51] L. Gouveia and P. Martins. "Solving the maximum edge-weight clique problem in sparse graphs with compact formulations". *EURO Journal on Computational Optimization* 3(1) (2015), pp. 1–30.

[52] M. Grbić. "Conditional Random Fields-based Aproach to Classification: Application to Life Sciences". *IPSI BgD Transactions on Advanced Research (TAR)* 15(1) (2019), pp. 1–9.

[53] M. Grbić, A. Kartelj, S. Janković, D. Matić, and V. Filipović. "Variable neighborhood search for partitioning sparse biological networks into the maximum edge-weighted k-plexes". *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2019). DOI: 10.1109/TCBB.2019.2898189.

[54] M. Grötschel and Y. Wakabayashi. "A cutting plane algorithm for a clustering problem". *Mathematical Programming* 45(1) (1989), pp. 59–96.

[55] P. Hansen, N. Mladenović, and J. A. M. Pérez. "Variable neighbourhood search: methods and applications". *4OR* 6(4) (2008), pp. 319–360.

[56] P. Hansen, N. Mladenović, R. Todosijević, and S. Hanafi. "Variable neighborhood search: basics and variants". *EURO Journal on Computational Optimization* 5(3) (2017), pp. 423–454.

[57] E. Hartuv, A. O. Schmitt, J. Lange, S. Meier-Ewert, H. Lehrach, and R. Shamir. "An algorithm for clustering cDNA fingerprints". *Genomics* 66(3) (2000), pp. 249–256.

[58] J. Hastings, P. de Matos, A. Dekker, M. Ennis, B. Harsha, N. Kale, V. Muthukrishnan, G. Owen, S. Turner, M. Williams, et al. "The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013". *Nucleic Acids Research* 41(D1) (2012), pp. D456–D463.

[59] X. He, R. S. Zemel, and M. Á. Carreira-Perpiñán. "Multiscale conditional random fields for image labeling" in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* Vol. 2. IEEE. 2004, pp. II–II.

[60] S. Hermann-Le Denmat, M. Werner, A. Sentenac, and P. Thuriaux. "Suppression of yeast RNA polymerase III mutations by FHL1, a gene coding for a fork head protein involved in rRNA processing." *Molecular and Cellular Biology* 14(5) (1994), pp. 2905–2913.

[61] M. A. Hibbs, D. C. Hess, C. L. Myers, C. Huttenhower, K. Li, and O. G. Troyanskaya. "Exploring the functional landscape of gene expression: directed search of large microarray compendia". *Bioinformatics* 23(20) (2007), pp. 2692–2699.

[62] A. Hickl and S. Harabagiu. "Enhanced interactive question-answering with conditional random fields" in *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006*. 2006, pp. 25–32.

[63] Y. Ho, A. Gruhler, A. Heilbut, G. D. Bader, L. Moore, S.-L. Adams, A. Millar, P. Taylor, K. Bennett, K. Boutilier, et al. "Systematic identification of protein complexes in Saccharomyces cerevisiae by mass spectrometry". *Nature* 415(6868) (2002), pp. 180–183.

[64] L.-L. Hu, C. Chen, T. Huang, Y.-D. Cai, and K.-C. Chou. "Predicting biological functions of compounds based on chemical-chemical interactions". *PloS one* 6(12) (2011), pp. 1–9.

[65] D. W. Huang, B. T. Sherman, and R. A. Lempicki. "Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists". *Nucleic acids research* 37(1) (2008), pp. 1–13.

[66] D. W. Huang, B. T. Sherman, and R. A. Lempicki. "Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources". *Nature Protocols* 4(1) (2009), pp. 44–57.

[67] F. Hüffner, C. Komusiewicz, A. Liebtrau, and R. Niedermeier. "Partitioning biological networks into highly connected clusters with maximum edge coverage". *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* 11(3) (2014), pp. 455–467.

[68] T. Jebara. *Machine learning: discriminative and generative*.

[69] D. Jiang and J. Pei. "Mining frequent cross-graph quasi-cliques". *ACM Transactions on Knowledge Discovery from Data (TKDD)* 2(4) (2009), 16:1–16:42.

[70] S. A. Kauffman. "Metabolic stability and epigenesis in randomly constructed genetic nets". *Journal of Theoretical Biology* 22(3) (1969), pp. 437–467.

[71] Z. Khuhro, F. Memon, M. Maree, and A. Harrison. "Cliq: A clique finding algorithm". *Sindh University Research Journal-SURJ (Science Series)* 44(2) (2012), pp. 313–318.

[72] L. Kiemer, S. Costa, M. Ueffing, and G. Cesareni. "WI-PHI: a weighted yeast interactome enriched for direct physical interactions". *Proteomics* 7(6) (2007), pp. 932–943.

[73] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[74] E. Korach and M. Stern. "The clustering matroid and the optimal clustering tree". *Mathematical Programming* 98(1-3) (2003), pp. 385–414.

[75] P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan. "A cluster-based approach for routing in dynamic networks". *ACM SIGCOMM Computer Communication Review* 27(2) (1997), pp. 49–64.

[76] N. J. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A. P. Tikuisis, et al. "Global landscape of protein complexes in the yeast Saccharomyces cerevisiae". *Nature* 440(7084) (2006), pp. 637–643.

[77] T. Kudo. "CRF++: Yet another CRF toolkit (2005)". *Available under LGPL from the following URL: http://crfpp.sourceforge.net* (2015).

[78] S. R. Kulkarni, D. Vaneechoutte, J. Van de Velde, and K. Vandepoele. "TF2Network: predicting transcription factor regulators and gene regulatory networks in Arabidopsis using publicly available binding site information". *Nucleic Acids Research* 46(6) (2017), e31. DOI: /10.1093/nar/gkx1279.

[79] S. Kumar and M. Hebert. "Discriminative random fields: A discriminative framework for contextual interaction in classification" in *Proceedings Ninth IEEE International Conference on Computer Vision*. 2003, pp. 1150–1157.

[80] J. Lafferty, A. McCallum, and F. C. Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data" in *Proceedings of the 18th International Conference on Machine Learning ICML*. 2001, pp. 282–289.

[81] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. "Neural architectures for named entity recognition". *arXiv preprint arXiv:1603.01360* (2016).

[82] B. Le Tallec, M.-B. Barrault, R. Guérois, T. Carré, and A. Peyroche. "Hsm3/S5b participates in the assembly pathway of the 19S regulatory particle of the proteasome". *Molecular Cell* 33(3) (2009), pp. 389–399.

[83] R. Leaman and G. Gonzalez. "BANNER: an executable survey of advances in biomedical named entity recognition" in *Pacific Symposium on Biocomputing*. World Scientific, 2008, pp. 652–663.

[84] V. E. Lee, N. Ruan, R. Jin, and C. Aggarwal. "A survey of algorithms for dense subgraph discovery" in *Managing and Mining Graph Data*. Springer, 2010, pp. 303–336.

[85] Z. Lei and Y. Dai. "Assessing protein similarity with Gene Ontology and its use in subnuclear localization prediction". *BMC bioinformatics* 7(1) (2006). DOI: 10.1186/1471-2105-7-491.

[86] T. Leifeld, Z. Zhang, and P. Zhang. "Identification of Boolean network models from time series data incorporating prior knowledge". *Frontiers in Physiology* 9 (2018). DOI: 10.3389/fphys.2018.00695.

[87] L. Licata, L. Briganti, D. Peluso, L. Perfetto, M. Iannuccelli, E. Galeota, F. Sacco, A. Palma, A. P. Nardozza, E. Santonico, et al. "MINT, the molecular interaction database: 2012 update". *Nucleic Acids Research* 40(D1) (2011), pp. D857–D861.

[88] G. Liu, L. Wong, and H. N. Chua. "Complex discovery from weighted PPI networks". *Bioinformatics* 25(15) (2009), pp. 1891–1897.

[89] H. Liu, P. Zhang, and D. Zhu. "On editing graphs into 2-club clusters" in *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management*. Springer, 2012, pp. 235–246.

[90] Y. Liu, J. Carbonell, P. Weigele, and V. Gopalakrishnan. "Protein fold recognition using segmentation conditional random fields (SCRFs)". *Journal of Computational Biology* 13(2) (2006), pp. 394–406.

[91] A. Maddi and C. Eslahchi. "Discovering overlapped protein complexes from weighted PPI networks by removing inter-module hubs". *Scientific Reports* 7(1) (2017).

[92] J. Maddock. "Genetic interactions among yeast gene products required for messenger-RNA processing." PhD thesis. Carnegie Mellon University, 1991.

[93] S. Maere, K. Heymans, and M. Kuiper. "BiNGO: a Cytoscape plugin to assess overrepresentation of gene ontology categories in biological networks". *Bioinformatics* 21(16) (2005), pp. 3448–3449.

[94] B. Magasanik. "Ammonia assimilation by Saccharomyces cerevisiae". *Eukaryotic Cell* 2(5) (2003), pp. 827–829.

[95] E. M. Makarov, O. V. Makarova, H. Urlaub, M. Gentzel, C. L. Will, M. Wilm, and R. Lührmann. "Small nuclear ribonucleoprotein remodeling during catalytic activation of the spliceosome". *Science* 298(5601) (2002), pp. 2205–2208.

[96] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[97] P. Martins. "Extended and discretized formulations for the maximum clique problem". *Computers & Operations Research* 37(7) (2010), pp. 1348–1358.

[98] P. Martins. "Modeling the maximum edge-weight k-plex partitioning problem". *arXiv preprint arXiv:1612.06243* (2016).

[99] A. McCallum and W. Li. "Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons" in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics. 2003, pp. 188–191.

[100] B. McClosky and I. V. Hicks. "Combinatorial algorithms for the maximum k-plex problem". *Journal of Combinatorial Optimization* 23(1) (2012), pp. 29–49.

[101] R. McDonald and F. Pereira. "Identifying gene and protein mentions in text using conditional random fields". *BMC bioinformatics* 6(1) (2005). DOI: 10.1186/1471-2105-6-S1-S6.

[102] C. v. Mering, M. Huynen, D. Jaeggi, S. Schmidt, P. Bork, and B. Snel. "STRING: a database of predicted functional associations between proteins". *Nucleic Acids Research* 31(1) (2003), pp. 258–261.

[103] H.-W. Mewes, D. Frishman, U. Güldener, G. Mannhaupt, K. Mayer, M. Mokrejs, B. Morgenstern, M. Münsterkötter, S. Rudd, and B Weil. "MIPS: a database for genomes and protein sequences". *Nucleic Acids Research* 30(1) (2002), pp. 31–34.

[104] N. Mladenović and P. Hansen. "Variable neighbourhood search". *Computers & Operations Research* 24 (1997), pp. 1097–1100.

[105] P. Moreno, S. Beisken, B. Harsha, V. Muthukrishnan, I. Tudose, A. Dekker, S. Dornfeldt, F. Taruttis, I. Grosse, J. Hastings, et al. "BiNChE: a web tool and library for chemical enrichment analysis based on the ChEBI ontology". *BMC Bioinformatics* 16(1) (2015). DOI: 10.1186/s12859-015-0486-3.

[106] H. Moser, R. Niedermeier, and M. Sorge. "Algorithms and experiments for clique relaxationsfinding maximum s-plexes" in *International Symposium on Experimental Algorithms*. Springer. 2009, pp. 233–244.

[107] M. Mukherjee and L. B. Holder. "Graph-based data mining on social networks". PhD thesis. University of Texas at Arlington, 2004.

[108] J. C. Nacher and T. Akutsu. "Minimum dominating set-based methods for analyzing biological networks". *Methods* 102 (2016), pp. 57–63.

[109] N. Nakajima, M. Hayashida, J. Jansson, O. Maruyama, and T. Akutsu. "Determining the minimum number of protein-protein interactions required to support known protein complexes". *PloS one* 13(4) (2018), pp. 1–17.

[110] S. Navlakha and C. Kingsford. "Exploring biological network dynamics with ensembles of graph partitions." in *Pacific Symposium on Biocomputing*. Vol. 15. 2010, pp. 166–177.

[111] D. L. Nelson, A. L. Lehninger, and M. M. Cox. *Lehninger principles of biochemistry*. Macmillan, 2008.

[112] T. Nepusz, H. Yu, and A. Paccanaro. "Detecting overlapping protein complexes in protein-protein interaction networks". *Nature Methods* 9(5) (2012). DOI: 10.1038/nmeth.1938.

[113] A. Y. Ng and M. I. Jordan. "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes" in *Advances in Neural Information Processing Systems*. 2002, pp. 841–848.

[114] D. A. T. Nijholt, L. De Kimpe, H. L. Elfrink, J. J. M Hoozemans, and W. Scheper. "Removing protein aggregates: the role of proteolysis in neurodegeneration". *Current medicinal chemistry* 18(16) (2011), pp. 2459–2476.

[115] N. Okazaki. "Crfsuite: a fast implementation of conditional random fields (crfs)". *Tehnical report* (2007).

[116] M. Oosten, J. H. Rutten, and F. C. Spieksma. "The clique partitioning problem: facets and patching facets". *Networks* 38(4) (2001), pp. 209–226.

[117] S. Orchard, M. Ammari, B. Aranda, L. Breuza, L. Briganti, F. Broackes-Carter, N. H. Campbell, G. Chavali, C. Chen, N. Del-Toro, et al. "The MIntAct projectIntAct as a common curation platform for 11 molecular interaction databases". *Nucleic Acids Research* 42(D1) (2013), pp. D358–D363.

[118] I. H. Osman and G. Laporte. "Metaheuristics: A bibliography". *Annals of Operations Research* (1996). DOI: 10.1007/BF02125421.

[119] J. Pattillo, N. Youssef, and S. Butenko. "Clique relaxation models in social network analysis" in Springer, 2012, pp. 143–162.

[120] D. Pinto, A. McCallum, X. Wei, and W. B. Croft. "Table extraction using conditional random fields" in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM. 2003, pp. 235–242.

[121] S. Pu, J. Wong, B. Turner, E. Cho, and S. J. Wodak. "Up-to-date catalogues of yeast protein complexes". *Nucleic Acids Research* 37(3) (2008), pp. 825–831.

[122] W. Pullan. "Approximating the maximum vertex/edge weighted clique using local search". *Journal of Heuristics* 14(2) (2008), pp. 117–134.

[123] S. Razick, G. Magklaras, and I. M. Donaldson. "iRefIndex: a consolidated protein interaction database with provenance". *BMC bioinformatics* 9(1) (2008). DOI: 10.1186/1471-2105-9-405.

[124] L. Ren, J. R. McLean, T. R. Hazbun, S. Fields, C. Vander Kooi, M. D. Ohi, and K. L. Gould. "Systematic two-hybrid and comparative proteomic analyses reveal novel yeast pre-mRNA splicing factors connected to Prp19". *PloS one* 6(2) (2011), pp. 1–16.

[125] X.-Z. Ren, L. H. Qiao, and Y. Qin. "A three-dimensional imaging algorithm for tomography SAR based on improved interpolated array transform1". *Progress In Electromagnetics Research* 120 (2011), pp. 181–193.

[126] A. V. Rudik, A. V. Dmitriev, A. A. Lagunin, D. A. Filimonov, and V. V. Poroikov. "Prediction of reacting atoms for the major biotransformation reactions of organic xenobiotics". *Journal of Cheminformatics* 8(1) (2016). DOI: 10.1186/s13321-016-0183-x.

[127] D. Rudra, Y. Zhao, and J. R. Warner. "Central role of Ifh1p–Fhl1p interaction in the synthesis of yeast ribosomal proteins". *The EMBO journal* 24(3) (2005), pp. 533–542.

[128] K. Sato and Y. Sakakibara. "RNA secondary structural alignment with conditional random fields". *Bioinformatics* 21(suppl_2) (2005), pp. ii237–ii242.

[129] D. Scharstein and C. Pal. "Learning conditional random fields for stereo" in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2007, pp. 1–8.

[130] S. B. Seidman and B. L. Foster. "A graph-theoretic generalization of the clique concept". *Journal of Mathematical Sociology* 6(1) (1978), pp. 139–154.

[131] B. Settles. "Biomedical named entity recognition using conditional random fields and rich feature sets" in *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)*. 2004, pp. 107–110.

[132] F. Sha and F. Pereira. "Shallow parsing with conditional random fields" in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics. 2003, pp. 134–141.

[133] R. Shamir, R. Sharan, and D. Tsur. "Cluster graph modification problems". *Discrete Applied Mathematics* 144(1-2) (2004), pp. 173–182.

[134] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. "Cytoscape: a software environment for integrated models of biomolecular interaction networks". *Genome Research* 13(11) (2003), pp. 2498–2504.

[135] R. Sharan, I. Ulitsky, and R. Shamir. "Network-based prediction of protein function". *Molecular Systems Biology* 3(1) (2007). DOI: 10.1038/msb4100129.

[136] A. K. Sharma, S. K. Jaiswal, N. Chaudhary, and V. K. Sharma. "A novel approach for the prediction of species-specific biotransformation of xenobiotic/drug molecules by the human gut microbiota". *Scientific Reports* 7(1) (2017). DOI: 10.1038/s41598-017-10203-6.

[137] D. Shen, J.-T. Sun, H. Li, Q. Yang, and Z. Chen. "Document summarization using conditional random fields." in *Proceedings of the 20th international joint conference on Artifical intelligence IJCAI*. vol. 7. 2007, pp. 2862–2867.

[138] J. Shotton, J. Winn, C. Rother, and A. Criminisi. "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation" in *European conference on computer vision*. Springer. 2006, pp. 1–15.

[139] N. Sokolovska, T. Lavergne, O. Cappé, and F. Yvon. "Efficient learning of sparse conditional random fields for supervised sequence labeling". *IEEE Journal of Selected Topics in Signal Processing* 4(6) (2010), pp. 953–964.

[140] V. Spirin and L. A. Mirny. "Protein complexes and functional modules in molecular networks". *Proceedings of the National Academy of Sciences* 100(21) (2003), pp. 12123–12128.

[141] C. Stark, B.-J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers. "BioGRID: a general repository for interaction datasets". *Nucleic Acids Research* 34(suppl_1) (2006), pp. D535–D539.

[142] S. W. Stevens, I. Barta, H. Y. Ge, R. E. Moore, M. K. Young, T. D. Lee, and J. Abelson. "Biochemical and genetic analyses of the U5, U6, and U4/U6 U5 small nuclear ribonucleoproteins from Saccharomyces cerevisiae". *RNA* 7(11) (2001), pp. 1543–1553.

[143] S. W. Stevens, D. E. Ryan, Y. G. Helen, R. E. Moore, M. K. Young, T. D. Lee, and J. Abelson. "Composition and functional characterization of the yeast spliceosomal penta-snRNP". *Molecular Cell* 9(1) (2002), pp. 31–44.

[144] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, et al. "Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles" in vol. 102. 43. National Acad Sciences, 2005, pp. 15545–15550.

[145] M. Suderman and M. Hallett. "Tools for visually exploring biological networks". *Bioinformatics* 23(20) (2007), pp. 2651–2659.

[146] J. Sun, P. Jia, A. H. Fanous, B. T. Webb, E. J. Van den Oord, X. Chen, J. Bukszar, K. S. Kendler, and Z. Zhao. "A multi-dimensional evidence-based candidate gene prioritization approach for complex diseases–schizophrenia as a case". *Bioinformatics* 25(19) (2009), pp. 2595–6602.

[147] C. Sutton, A. McCallum, et al. "An introduction to conditional random fields". *Foundations and Trends® in Machine Learning* 4(4) (2012), pp. 267–373.

[148] M. F. Tappen, C. Liu, E. H. Adelson, and W. T. Freeman. "Learning gaussian conditional random fields for low-level vision" in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2007, pp. 1–8.

[149] O. Tehlivets, K. Scheuringer, and S. D. Kohlwein. "Fatty acid synthesis and elongation in yeast". *Biochimica et Biophysica Acta (BBA)-Molecular and Cell Biology of Lipids* 1771(3) (2007), pp. 255–270.

[150] L. Terveen, W. Hill, and B. Amento. "Constructing, organizing, and visualizing collections of topically related web resources". *ACM Transactions on Computer-Human Interaction (TOCHI)* 6(1) (1999), pp. 67–94.

[151] H. Tipney and L. Hunter. "An introduction to effective use of enrichment analysis software". *Human Genomics* 4(3) (2010), p. 202.

[152] P. Tompa. "Intrinsically disordered proteins: a 10-year recap". *Trends in Biochemical Sciences* 37(12) (2012), pp. 509–516.

[153] S. Trukhanov, C. Balasubramaniam, B. Balasundaram, and S. Butenko. "Algorithms for detecting optimal hereditary structures in graphs, with application to clique relaxations". *Computational Optimization and Applications* 56(1) (2013), pp. 113–130.

[154] B. M. VandenBrink and N. Isoherranen. "The role of metabolites in predicting drug-drug interactions: Focus on irreversible P450 inhibition". *Current opinion in drug discovery & development* 13(1) (2010), pp. 66–77.

[155] D. Vella, I. Zoppis, G. Mauri, P. Mauri, and D. Di Silvestre. "From protein-protein interactions to protein co-expression networks: a new perspective to evaluate large-scale proteomic data". *EURASIP Journal on Bioinformatics and Systems Biology* 2017(1) (2017). DOI: 10.1186/s13637-017-0059-z.

[156] D. Vilchez, I. Saez, and A. Dillin. "The role of protein clearance mechanisms in organismal ageing and age-related diseases". *Nature Communications* 5 (2014). DOI: 10.1038/ncomms6659.

[157] D Voges, P Zwickl, and W Baumeister. "The 26S proteasome: a molecular machine designed for controlled proteolysis". *Annual Review of Biochemistry* 68(1) (1999), pp. 1015–1068.

[158] C. Von Mering, L. J. Jensen, B. Snel, S. D. Hooper, M. Krupp, M. Foglierini, N. Jouffre, M. A. Huynen, and P. Bork. "STRING: known and predicted protein–protein associations, integrated and transferred across organisms". *Nucleic Acids Research* 33(suppl_1) (2005), pp. D433–D437.

[159] T. Vujičić, J. Glass, F. Zhou, and Z. Obradović. "Gaussian conditional random fields extended for directed graphs". *Machine Learning* 106(9-10) (2017), pp. 1271–1288.

[160] H. Wang, B. Alidaee, F. Glover, and G. Kochenberger. "Solving group technology problems via clique partitioning". *International Journal of Flexible Manufacturing Systems* 18(2) (2006), pp. 77–97.

[161] J. Wang, C.-E. Wang, A. Orr, S. Tydlacka, S.-H. Li, and X.-J. Li. "Impaired ubiquitin–proteasome system activity in the synapses of Huntington's disease mice". *The Journal of Cell Biology* 180(6) (2008), pp. 1177–1189.

[162] Y. Wang, K.-F. Loe, and J.-K. Wu. "A dynamic conditional random field model for foreground and shadow segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28(2) (2005), pp. 279–289.

[163] I. Xenarios, L. Salwinski, X. J. Duan, P. Higney, S.-M. Kim, and D. Eisenberg. "DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions". *Nucleic Acids Research* 30(1) (2002), pp. 303–305.

[164] *Yeast Pathways Database*. https://pathway.yeastgenome.org/. Accessed: 2017-11-11.

[165] H. Yu, P. Braun, M. A. Yıldırım, I. Lemmens, K. Venkatesan, J. Sahalie, T. Hirozane-Kishikawa, F. Gebreab, N. Li, N. Simonis, et al. "High-quality binary protein interaction map of the yeast interactome network". *Science* 322(5898) (2008), pp. 104–110.

[166] Z. Zhang, J. Song, J. Tang, X. Xu, and F. Guo. "Detecting complexes from edge-weighted PPI networks via genes expression analysis". *BMC systems biology* 12(4) (2018). DOI: 10.1186/s12918-018-0565-y.

[167] N. Zhou, Y. Jiang, T. R. Bergquist, A. J. Lee, B. Z. Kacsoh, A. W. Crocker, K. A. Lewis, G. Georghiou, H. N. Nguyen, M. N. Hamid, et al. "The CAFA challenge reports improved protein function prediction and new functional annotations for hundreds of genes through experimental screens". *bioRxiv* (2019).

[168] Y. Zhou, J.-K. Hao, and A. Goëffon. "A three-phased local search approach for the clique partitioning problem". *Journal of Combinatorial Optimization* 32(2) (2016), pp. 469–491.

[169] X. Zhu, M. Gerstein, and M. Snyder. "Getting connected: analysis and principles of biological networks". *Genes & development* 21(9) (2007), pp. 1010–1024.

BIOGRAFIJA

Milana Grbić je rođena u Banjoj Luci 15. avgusta 1989. godine, gdje je završila osnovnu školu i Gimnaziju. Osnovne studije je završila na Prirodno-matematičkom fakultetu Univerziteta u Banjoj Luci 2012. godine, stekavši zvanje Diplomirani matematičar i informatičar. Master studije studijskog programa Računarstvo i informatika Matematičkog fakulteta Univerziteta u Beogradu završila je 2016. godine, odbranivši master rad pod nazivom „Grupisanje organizama pomoću različitih metoda klasifikacije u zavisnosti od genotipskih i fenotipskih karakteristika“ pod rukovodstvom prof. dr Nenada Mitića. Doktorske studije studijskog programa Informatika na Matematičkom fakultetu je upisala 2016. godine.

Milana Grbić je od 2013. godine zaposlena na Prirodno-matematičkom fakultetu Univerziteta u Banjoj Luci, prvo na poziciji asistenta, a od 2017. godine na poziciji višeg asistenta. Do sada je izvodila vježbe iz više informatičkih predmeta: Uvod u programiranje, Osnove računarskih sistema 1, Osnove računarskih sistema 2, Osnove računarskih sistema 3, Osnove računarskih sistema 4, Operativni sistemi, Bioinformatika i dr.

Učesnik je više domaćih istraživačkih projekata i jednog međunarodnog istraživačkog projekata iz oblasti bioinformatike. U toku studiranja na doktorskim studijama boravila je na više ljetnjih i zimskih škola. Učestvovala je na više međunarodnih konferencija iz oblasti bioinformatike.

Prilog 1.

Izjava o autorstvu

Potpisana: Milana Grbić

broj upisa: 2011/2016

Izjavljujem

da je doktorska disertacija pod naslovom

Računarske metode particionisanja i grupisanja u biološkim mrežama

- rezultat sopstvenog istraživačkog rada,
- da predložena disertacija u celini ni u delovima nije bila predložena za dobijanje bilo koje diplome prema studijskim programima drugih visokoškolskih ustanova,
- da su rezultati korektno navedeni i
- da nisam kršio/la autorska prava i koristio intelektualnu svojinu drugih lica.

Potpis doktoranda

U Beogradu, 16.12.2019.

Prilog 2.

Izjava o istovetnosti štampane i elektronske verzije doktorskog rada

Ime i prezime autora: Milana Grbić

Broj upisa: 2011/2016

Studijski program: Informatika

Naslov rada: Računarske metode particionisanja i grupisanja u biološkim mrežama

Mentor: dr Gordana Pavlović-Lažetić, redovni profesor

Potpisana Milana Grbić

izjavljujem da je štampana verzija mog doktorskog rada istovetna elektronskoj verziji koju sam predao/la za objavljivanje na portalu **Digitalnog repozitorijuma Univerziteta u Beogradu.**

Dozvoljavam da se objave moji lični podaci vezani za dobijanje akademskog zvanja doktora nauka, kao što su ime i prezime, godina i mesto rođenja i datum odbrane rada.

Ovi lični podaci mogu se objaviti na mrežnim stranicama digitalne biblioteke, u elektronskom katalogu i u publikacijama Univerziteta u Beogradu.

Potpis doktoranda

U Beogradu, 16.12.2019.

Prilog 3.

Izjava o korišćenju

Ovlašćujem Univerzitetsku biblioteku „Svetozar Marković“ da u Digitalni repozitorijum Univerziteta u Beogradu unese moju doktorsku disertaciju pod naslovom:

Računarske metode particionisanja i grupisanja u biološkim mrežama

koja je moje autorsko delo.

Disertaciju sa svim prilozima predao/la sam u elektronskom formatu pogodnom za trajno arhiviranje.

Moju doktorsku disertaciju pohranjenu u Digitalni repozitorijum Univerziteta u Beogradu mogu da koriste svi koji poštuju odredbe sadržane u odabranom tipu licence Kreativne zajednice (Creative Commons) za koju sam se odlučio/la.

1. Autorstvo

2. Autorstvo - nekomercijalno

3. Autorstvo – nekomercijalno – bez prerade

4. Autorstvo – nekomercijalno – deliti pod istim uslovima

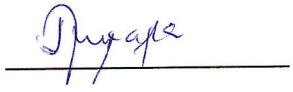
5. Autorstvo – bez prerade

6. Autorstvo – deliti pod istim uslovima

(Molimo da zaokružite samo jednu od šest ponuđenih licenci, kratak opis licenci dat je na poleđini lista).

Potpis doktoranda

U Beogradu, 16.12.2019.



1. Autorstvo - Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora na način određen od strane autora ili davaoca licence, čak i u komercijalne svrhe. Ovo je najslobodnija od svih licenci.

2. Autorstvo – nekomercijalno. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora na način određen od strane autora ili davaoca licence. Ova licenca ne dozvoljava komercijalnu upotrebu dela.

3. Autorstvo - nekomercijalno – bez prerade. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, bez promena, preoblikovanja ili upotrebe dela u svom delu, ako se navede ime autora na način određen od strane autora ili davaoca licence. Ova licenca ne dozvoljava komercijalnu upotrebu dela. U odnosu na sve ostale licence, ovom licencom se ograničava najveći obim prava korišćenja dela.

4. Autorstvo - nekomercijalno – deliti pod istim uslovima. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora na način određen od strane autora ili davaoca licence i ako se prerada distribuira pod istom ili sličnom licencom. Ova licenca ne dozvoljava komercijalnu upotrebu dela i prerada.

5. Autorstvo – bez prerade. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, bez promena, preoblikovanja ili upotrebe dela u svom delu, ako se navede ime autora na način određen od strane autora ili davaoca licence. Ova licenca dozvoljava komercijalnu upotrebu dela.

6. Autorstvo - deliti pod istim uslovima. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora na način određen od strane autora ili davaoca licence i ako se prerada distribuira pod istom ili sličnom licencom. Ova licenca dozvoljava komercijalnu upotrebu dela i prerada. Slična je softverskim licencama, odnosno licencama otvorenog koda.