SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE VARAŽDIN

Benkus Maja, Črepinko Marko, Ćurić Krešimir, Garić Luka, Košmerl Igor, Krištofić Miro

Primjena suvremenih tehnologija temeljenih na grafovima za reprezentaciju i analizu znanstvenih publikacija istraživača u Republici Hrvatskoj

Zagreb, 2019.

Ovaj rad izrađen je u Laboratoriju za podatkovne tehnologije pod vodstvom prof. dr. sc. Kornelija Rabuzina i Martine Šestak, mag. inf. te je predan na natječaj za dodjelu Rektorove nagrade u akademskoj godini 2018./2019.

Popis i objašnjenje kratica

CC (engl. *Current Contents Connect)* je najpoznatija baza podataka koja je u Hrvatskoj dostupna putem Web of Science sučelja. Bazu karakteriziraju visoki kriteriji odabira za časopise.

CSV (engl. *Comma Separated Values*) je tip datoteke koji koristi određeni znak (najčešće zarez) kao graničnik između pojedinih zapisa.

GDBMS (engl. *Graph Database Management System*) je sustav za upravljanje grafovskim bazama podataka.

JSON (engl. *JavaScript Object Notation*) je format otvorenog standarda koji koristi tekst čitljiv ljudima za prijenos podataka u obliku objekata koji se sastoje od parova atribut-vrijednost te nizova.

NoSQL (engl. *Non relational, Not only SQL*) je pojam koji označava nerelacijske baze podataka, odnosno baze podataka koje se ne temelje na relacijskom modelu podataka i ne koriste SQL kao primarni upitni jezik.

SQL (engl. *Structured Query Language*) je standardizirani jezik za rad s bazama podataka. Omogućuje kreiranje baze podataka, manipuliranje podacima i dohvaćanje podataka iz baze.

ACID (engl. *Atomicity, Consistency, Isolation, Durability*) je skup svojstava (atomnost, konzistentnost, izolacija i trajnost) vezan uz transakcije u bazama podataka koji omogućuje kontinuiranu konzistentnost baza podataka.

HTML (engl. *HyperText Markup Language*) je standardizirani jezik oznaka (engl. *markup language*) za razvoj Web stranica.

CRUD (engl. *Create, Read, Update, Delete*) je skup operacija koji obuhvaća kreiranje, čitanje (dohvaćanje), ažuriranje i brisanje sadržaja.

Sadržaj

Popis slika

1. Uvod

Dužnost svakog pojedinog znanstvenika i istraživača je pridonijeti rastu i razvoju znanstvene zajednice u kojoj djeluje. Rastu i razvoju znanstvene zajednice može se pridonijeti na mnogo različitih načina: prenošenjem svog vlastitog znanja i iskustva na ostale pojedince, znanstvenim projektima te izradom i objavljivanjem znanstvenih radova. Izrada i objavljivanje znanstvenih radova jedna je od najbitnijih stavki napretka znanstvenika, bilo profesionalnog (napredovanje u znanstveno zvanje) ili osobnog. Tematika ovog rada upravo je vezana uz objave znanstvenih radova hrvatskih istraživača te analizu objavljenih znanstvenih publikacija istraživača Republike Hrvatske. U radu proučavamo načine prikaza znanstvenih publikacija i mreža suradnje radi osmišljavanja drugačijeg načina prikaza podataka iz domene. Osim toga, rad se bavi ekstrakcijom podataka za izradu grafovske baze podataka na kojoj se temelji izrada grafovskog skladišta podataka koje će služiti kao izvor podataka za naknadnu analizu objavljenih publikacija istraživača.

Podaci korišteni u ovom radu prikupljeni su s CROSBI-ja, hrvatske znanstvene bibliografije. Nakon prikupljanja podataka, provedene su potrebne transformacije kako bismo prikupljene podatke prikazali u obliku mreže znanstvenika i publikacija. Za prikaz prikupljenih podataka korišten je *Neo4j* – jedna od najkorištenijih tehnologija u svijetu grafovskih baza podataka. Pregledom znanstvenih radova kojima je tematika prikaz kolaboracijskih mreža znanstvenika, ustanovili smo da je „prirodan“ način prikaza mreže pomoću grafa, što je i temelj grafovskih baza podataka.

Između ostalog, krajnji cilj ovog rada je izrada grafovskog skladišta podataka temeljenog na grafovskoj bazi podataka, pa će tako u sklopu rada biti provedeno eksperimentalno istraživanje koje obuhvaća znanstvene publikacije hrvatskih istraživača uz pomoć prethodno navedenih tehnologija za izradu grafovskog skladišta podataka.

Glavna motivacija za ovaj rad proizlazi iz činjenice da je intuitivan prikaz mreže znanstvenika, njihove suradnje i publikacija u obliku grafa, a tijekom posljednjih nekoliko godina sve važniju ulogu dobivaju *NoSQL* grafovske baze podataka koje upravo omogućuju pohranu takvih struktura u obliku grafa. Dodatno, budući da se postojeći podaci o znanstvenoj produktivnosti istraživača dostupni putem servisa rijetko mijenjaju, skladište podataka kao tehnologija pohrane može se iskoristiti kao dobar način pohrane podataka iz te domene, gdje unaprijed agregirani podaci mogu pozitivno utjecati na performanse budućih upita i omogućiti preciznije analize u odnosu na transakcijske baze podataka.

U nastavku uvoda opisuju se opći i specifični ciljevi rada te se postavljaju hipoteze. Drugo se poglavlje bavi opisom domene rada te analizom pristupa za analizu znanstvene produktivnosti spomenutih u radovima različitih autora. Sljedeće, treće poglavlje opisuje grafovske baze podataka i način njihova korištenja. Nakon toga, u četvrtom poglavlju, objašnjava se što su to skladišta podataka, koje su njihove prednosti i za što se koriste. U petom poglavlju detaljno se objašnjava proces izrade grafovskog skladišta podataka. Ovdje se opisuje izrada modela skladišta dimenzijskim modeliranjem, prikupljanje potrebnih podataka, njihova transformacija te u konačnici njihovo spajanje. Osim toga, opisane su sve tehnologije korištene u realizaciji rada te naposljetku rezultati upita u skladištu podataka čiji su rezultati grafički prikazani u obliku vizualizacija.

1.1. Hipoteza / Opći i specifični ciljevi rada

Svaki znanstveni rad i istraživanje sastoji se od nekoliko bitnih koraka. Prvi korak je određivanje problematike rada i ciljeva koji se tim radom žele ostvariti. Nakon identificiranja i definiranja problema potrebno je postaviti hipoteze koje će se kroz izradu rada testirati. Problematika kojom se naš rad bavi je primjena tehnologija temeljenih na grafovima za reprezentaciju i analizu znanstvenih publikacija hrvatskih znanstvenika.

Opći ciljevi koje želimo ostvariti u ovom radu su:

- Primijeniti pristup temeljen na grafovima za reprezentaciju znanstvenih publikacija istraživača u Republici Hrvatskoj.
- Dobiveni model mreže znanstvenika i njihovih publikacija pohraniti u grafovsku bazu podataka u konačnom vremenu.
- Temeljem grafovske baze podataka izraditi grafovsko skladište podataka nad kojim će biti moguće izvršavati upite, odnosno služit će kao dobar izvor podataka za daljnju analizu hrvatske mreže znanstvenika i njihovih publikacija.

Na temelju općih ciljeva moguće je navesti specifične ciljeve: naučiti modelirati grafovsko skladište podataka i usporediti model grafovskog skladišta podataka s relacijskim modelom, upoznati različite načine prikaza kolaboracijskih mreža znanstvenika te istražiti mogućnosti interakcije aplikacija s grafovskom bazom podataka i skladištem podataka.

Nakon postavljanja ciljeva, a prije same izrade rada, postavili smo tri hipoteze koje ćemo dokazati u našem radu. Hipoteze su sljedeće:

- Grafovsko skladište podataka je prikladna tehnologija za prikaz domene znanstvenih publikacija i njihovih autora (H1).
- Postojeće tehnologije za razvoj aplikacija omogućuju razvoj sučelja za interaktivan prikaz podataka pohranjenih u grafovskom skladištu podataka (H2).
- Grafovska baza podataka omogućuje pripremu podataka koji se potom mogu pohraniti u grafovsko skladište podataka (H3).

2. Opis domene i pristupa za reprezentaciju i analizu mreža znanstvenika

Znanstvenici diljem svijeta bave se analizom mreže znanstvenika (engl. *scientific network*), njihovom međusobnom suradnjom te znanstvenom produktivnošću. Podaci se prikupljaju iz različitih baza podataka (IEEE Xplore, ESI, Web of Science) te se nad podacima vrše različite tehnike analize. U ovom poglavlju opisat ćemo mrežu znanstvenika u Republici Hrvatskoj, dostupne servise koji sadrže podatke o znanstvenoj produktivnosti i pokazati različite pristupe za analizu znanstvene produktivnosti.

2.1. Opis domene

Domena našeg rada bit će mreža znanstvenika i njihove međusobne suradnje u Republici Hrvatskoj. Promatrat ćemo podatke sakupljene sa servisa koji sadrže podatke o znanstvenoj produktivnosti unutar Republike Hrvatske. Servis koji smo odabrali je CROSBI (Hrvatska znanstvena bibliografija). Osim CROSBI-ja, postoji još nekoliko servisa koji bilježe znanstvenu produktivnost hrvatskih znanstvenika čiji pregled donosimo u nastavku ovog poglavlja.

CROSBI, Hrvatska znanstvena bibliografija prvi puta je objavljena 1997. godine i razvijena od strane Knjižnice Instituta Ruđer Bošković (sadašnji Centar za znanstvene informacije) [1]. Iako radove hrvatskih znanstvenika možemo pronaći u poznatim bibliografskim bazama podataka (poput Web of Science i Scopus), CROSBI je razvijen s ciljem da se na jednom mjestu okupi cjelovita publicistika hrvatskih znanstvenika. Danas CROSBI sadrži podatke o više od 520 000 radova te je taj broj u konstantnom porastu [2]. 2018. godine

upisano je 24 914 novih zapisa u CROSBI. Najveći broj radova, njih čak 191 905 čine radovi u časopisima čije smo podatke i mi prikupljali u izradi ovog rada.

Digitalni akademski arhivi i repozitoriji, Dabar, jedna je od komponenti e-infrastrukture Republike Hrvatske koja ustanovama iz sustava visokog obrazovanja omogućava kreiranje vlastitog repozitorija kako bi na sustavan način mogli brinuti o svojoj digitalnoj imovini. Kreiranje digitalnih repozitorija i arhiva je besplatno za ustanove iz sustava visokog obrazovanja i znanosti. Dabar je nastao, a i dalje se razvija, međusobnom suradnjom i okupljanjem ustanova i članova hrvatske akademske zajednice [3]. Trenutačno pohranjuje 127 repozitorija te 89 166 objekata. Fakultet organizacije i informatike također posjeduje repozitorij na Dabru te trenutačno broji 3 687 objekata od čega njih 57.7% čine završni radovi [4].

Hrčak je portal hrvatskih znanstvenih i stručnih časopisa koji na jednom mjestu okuplja časopise s otvorenim pristupom svojim radovima. Trenutačno se u Hrčku nalazi 470 časopisa i 206 451 rad. Kao i Dabar, realiziran je u Srcu – Sveučilišnom računskom centru [5].

Google Znalac [6] hrvatska je inačica poznatog svjetskog pretraživača znanstvene i stručne literature Google Scholar. Kao i većina servisa koji sadrže podatke o znanstvenoj produktivnosti omogućava pretragu prema autoru, naslovu rada ili prema godinama. Parametri pretraživanja mogu se međusobno kombinirati. Od 2012. godine moguće je kreirati svoj osobni profil na Google Znalcu te će Google pretražiti Internet te pohraniti sve radove novokreiranog autora u posebnu datoteku. Svojim korisnicima Google Znalac nudi nekoliko zanimljivih funkcionalnosti, od kojih je najzanimljivija „Moja knjižnica“ gdje se pohranjuju članci koje kasnije želimo pročitati.

2.2. Pristupi za analizu znanstvene produktivnosti istraživača

U radu „Analyzing International Scientific Collaboration Pattern for China by Using ESI Database“ [7] autori Xiang i Li promatraju suradnju kineskih znanstvenika na međunarodnoj razini. Naspram ostalih znanstvenih radova koji promatraju broj znanstvenih suradnji (međunarodnih ili unutar zemlje) ovaj rad se koncentrira na kvalitetu znanstvenih suradnji, a ne na njihovu kvantitetu.

Kao skup podataka (engl. *dataset*) korišten je *Essential Science Indicators* (u daljnjem tekstu: ESI) skup podataka gdje se promatraju najviše citirani radovi. Značajke u ESI skupu podataka su sljedeće: broj citiranja, naslov rada, autori rada, izvor rada, adrese autora i

znanstveno područje. Autori su broj značajki smanjili na tri najbitnije: broj citiranja, zemlja autora te znanstveno područje. Za analizu znanstvenih suradnji korišten je programski jezik *Python* dok je za vizualizaciju ove međunarodne znanstvene mreže korišten Netdraw program.

Analiza je vizualizirana pomoću grafa gdje čvorove čine zemlje, dok je brid zapravo reprezentacija znanstvene suradnje između autora iz dvije različite zemlje (iz Kine i neke druge). Zemlje s kojima je Kina najviše surađivala u znanstvenom radu i istraživanju su Sjedinjene Američke Države, Japan, Ujedinjeno Kraljevstvo, Njemačka i Kanada.

Autori Li i ostali u članku „Evolving model of weighted networks inspired by scientific collaboration networks“ [8] opisuju postupak kako doći do modela težinskih mreža koji se temeljni na analizi mreže znanstvenika. Kao što je i poznato, kod težinskih mreža ili grafova, težina brida označava snagu interakcije između dva čvora. Što je težina brida veća, to je suradnja između dva znanstvenika veća.

Kada analiziraju mrežu znanstvenika do težine brida dolaze na način da broje suradnje između dva čvora veze, odnosno, broje broj suradnji između dva autora (broj članaka i sl.). U članku se razmatra na koji način težinska mreža znanstvenika može evoluirati dodavanjem novog čvora (autora).

U radu „Key Community Analysis In Scientific Collaboration Network“ [9] autori konstruiraju mrežu znanstvenika pomoću skupa podataka u CSV formatu prikupljenih iz IEEE Xplore baze s ključnim riječima iz područja računarske znanosti.

Cilj rada je pronaći zajednice znanstvenika pomoću nekoliko parametara. Prvi parametar je sličnost čvorova koja se računa Jaccardovom sličnosti ili Kosinus sličnosti te pokazuje sličnost između čvorova (autora) koji su surađivali zajedno u znanstvenom radu. Sljedeći parametar kojeg autori koriste je stupanj čvorova kako bi pronašli podgrafove odnosno klike. Klika (engl. *clique*) je inducirani podgraf grafa koji je potpuno povezan [10]. Koristili su Bron Kerboch algoritam za traženje najveće klike. Zadnji parametar je doseg čvora, broj radova gdje je autor imao ulogu koautora. Autori su računali centralizaciju mreže, gustoću mreže i prosječni koeficijent klasteriranja. Svi izračuni su provedeni u alatu MATLAB 2015.

U radu „The Evolution of International Scientific Collaboration in Fuel Cells during 1998–2017: A Social Network Perspective“ [11] je opisano istraživanje u kojem su se analizirali uzorci i dinamika internacionalne znanstvene kolaboracijske mreže na području gorivih članaka (engl. *fuel cells*). Temelj istraživanja su bili radovi koji su se dobili iz baze Web of Science pretraživanjem prema ključnim riječima *fuel cell* ili *fuel cells* te filtriranjem tako da su im autori iz različitih država. Za analizu se koristio *Bibexcel* što je alat za analizu bibliografskih

podataka. Analiza je ocjenjivala države prema koautorstvu te ih je prikazala u mrežnoj strukturi. Osim toga, alat je identificirao koautorstvo država u pojedinom radu. Čvor je mreži predstavljao državu, a brid koautorsku vezu.

Istraživanje opisano u radu „Bowling Together: Scientific Collaboration Networks of Demographers at European Population Conferences“ [12] bavilo se analizom kolaboracijskih mreža demografskih znanstvenika čiji su radovi bili predstavljeni na Europskim demografskim konferencijama. Analiza se temeljila na lokaciji institucija, spolu autora i općoj kategoriji rada. Lokacija institucije se najčešće određivala pomoću *Google Maps API*-ja, spol autora se odredio pomoću *genderize.io* baze što je najveća baza podataka poznatih kombinacija spola i imena s društvenih mreža.

Rad „Co-authorship network analysis in health research: method and potential use“ [13] opisuje važnost kolaboracije znanstvenika u području medicine te metodu i korištenje analize mreže koautora. Zatim opisuje korake metode analize koautorstva te daje primjer takve analize. Primjer govori o globalnoj istraživačkoj mreži određenog cjepiva te podacima prikupljenima iz Web of Science baze podataka. Čišćenje i mapiranje podataka odradilo se pomoću softvera *VantagePoint*. Ti su se podaci pretvorili u CSV datoteku koja se uvezla su *Gephi* softver koji je služio za vizualizaciju i analizu. Čvorovi u grafu su bile države ili institucije.

U radu „A fast method for identifying worldwide scientific collaborations using the Scopus database“ [14] se opisuje metoda za analizu znanstvenih kolaboracijskih mreža na sveučilištima i u istraživačkim institucijama koje su čvorovi u mreži. Ta se metoda temelji na dobivanju bibliografskih podataka iz znanstvenih radova kroz *Scorpus Database API* koji su zatim analizirani koristeći softver za vizualizaciju grafova i statističke alate. Opisali su se koraci u dobivanju i čišćenju podataka te se spominje korištenje alata *Gephi*.

U radu „A new network model for the study of scientific collaborations: Romanian computer science and mathematics co-authorship networks“ [15] se govori kako kolaboracijske mreže s autorima kao čvorovima imaju nedostatke i da ne prikazuju autorov učinak. Prema tome, predlaže se drugačiji model mreže u kojoj su čvorovi zapravo objavljeni radovi, a veze među njima su autori. Navedeni model testiran je nad radovima i autorima rumunjskih institucija s područja računalne znanosti i matematike te su primijećena poboljšanja u odnosu na originalni pristup modeliranju mreže. U nastavku teksta su uspoređene mreže radova (gdje su radovi čvorovi) s mrežama autora (gdje su autori čvorovi). Zaključci su kako su mreže radova manje i gušće, ukazuju na uzorke u kolaboraciji i prikazuju specifične aspekte različitih polja istraživanja. Takve mreže su efikasnije i više ciljane na same autore, tj. bolje su

u ocjenjivanju autorovog učinka i uspoređivanju autora u tom smislu, no da je za cjelokupnu sliku potrebno analizirati oba modela kolaboracijskih mreža.

Nadalje, u radu „An analysis of the structure and evolution of the scientific collaboration network of computer intelligence in games“ [16], autori se bave analizom mreže suradnje računalne inteligencije u računalnim igrama. Pritom je naglasak stavljen na konstrukciju kolaboracijske mreže znanstvenika temeljene na bibliografskom poslužitelju *Digital Bibliography and Library Project* (u daljnjem tekstu: DBLP) te razvoj upita i vizualizacija istih korištenjem grafovskih baza podataka. Temeljna svrha ovog istraživanja bila je vizualizirati međusobnu povezanost autora radova s područja računalne inteligencije u igrama (engl. *Computational Intelligence in Games,* u daljnjem tekstu: *CIG*) u razdoblju od 1997. do 2011. godine. Pritom se kao dva najkorištenija alata spominju *Cytoscape* za crtanje i analizu grafova te biblioteka *igraph* korištena za analizu kompleksne mreže. Cjelokupno *CIG* područje analizirano je iz vremenske perspektive na makroskopskoj, mezoskopskoj i mikroskopskoj razini. Takva vrsta analize omogućena je prikazom kolaboracijske mreže u obliku ponderiranog grafa, u kojem svaki čvor simbolizira autora, a svaka veza suradnju povezanih autora. Osim toga, ovakvom pozadinskom strukturom omogućena je konstrukcija upita nad skupom podataka poput:

- pretraživanja svih članaka objavljenih u sklopu određene konferencije ili časopisa s područja *CIG*
- pretraživanja svih članaka s konferencija ili časopisa korištenjem ključnih riječi vezanih uz temu igara

Jedini nedostatak navedenog modela je pojavljivanje duplikata u imenima, prezimenima i inicijalima zbog korištenja uzorka s lako dostupne baze podataka bibliografije, no autori smatraju da je takvo odstupanje prihvatljivo zbog relativno velikog skupa podataka. Iako se naš znanstveni rad strogo ne dotiče s *CIG* tematikom, aspekt kolaboracijske mreže biti će primijenjen na sličan način, ali u domeni znanstvenih publikacija hrvatskih istraživača. Nedostatak pojavljivanja duplikata u ovom radu izbjeći će se korištenjem jedinstvenog identifikatora svakog autora i rada.

Radom „The effect of data pre-processing on understanding the evolution of collaboration networks“ [17] objedinjeno je 125 dnevnika s područja informacijskih sustava (u 30 godina) s ciljem da se prikaže utjecaj odabira metoda preprocesiranja podataka na konačno razumijevanje razvoja koautorskih mreža. Podaci su preuzeti s DBLP-a kao i u prijašnjem radu. Ciljana skupina podataka bila su imena autora, pri čemu je za usporedbu s metodom preprocesiranja korišten algoritamski pristup za analizu podataka. Rezultati su pokazali da

preprocesirani zapisi eksponencijalno povećavaju ukupan broj autora, dok je korištenjem algoritama dobiven precizniji rezultat. Utvrđeno je da takve greške u zapisu rastu s vremenom te kao takve mogu značajno utjecati na interpretaciju podataka, shvaćanje cjelokupne topologije sustava i predviđanje budućih trendova. J. Kim i J. Diesner također naglašavaju da bi upravo zbog toga studenti trebali biti pažljivi pri odabiru metode preprocesiranja podataka.

U istraživanju znanstvenika Sveučilišta u Indiani (SAD) [18] za analizu područja znanstvene suradnje koristi se algoritamski pristup. Ciljano se traži odgovor na dva pitanja: teže li produktivni autori suradnji sa znanstvenicima iz područja istog interesa te teže li često citirani autori međusobnoj suradnji. Vrste algoritma korištene u analizi usko su vezane uz područje strojnog učenja, pa se tako koristi algoritam za modeliranje tema (engl. *Topic modeling algorithm*) i algoritam za pronalaženje najkraćeg puta (engl. *Path-finding algorithm*) kako bi se dubinski analiziralo podatke o istraživačkim temama radova znanstvenika s raznih područja znanosti. Konačan cilj dubinske analize podataka jest njihova analiza i donošenje konačnog odgovora na prethodno spomenuta pitanja. Autori ističu da kombinacija algoritma za modeliranje teme i analize mreže suradnje proširuje raspon dobivenih informacija sa saznanjima o društvenoj interakciji. Konačan rezultat istraživanja, između ostalog, novih saznanja o dotad netaknutom području, pokazuju da autori s istog područja interesa teže međusobnoj suradnji i citiranju, dok često citirani autori međusobno ne surađuju, no često se međusobno citiraju.

Sljedećim radom [19] obuhvaćeni su podaci iz domene biomedicine (*MEDLINE*), fizike (*Los Alamos e-Print Archive*) i računalnih znanosti (engl. *Networked Computer Science Technical Reference Library, NCSTRL*) u razdoblju od pet godina (1995.-1999.). Sami podaci povučeni su iz nekoliko velikih baza podataka na području Sjedinjenih Američkih Država te je za svaku domenu kreirana zasebna mreža suradnje. Među mnogim pronađenim značajkama ističe se nekoliko bitnijih. Prva značajka kaže da je prosječno potrebno napraviti pet do šest koraka među čvorovima (autorima) iz istog područja znanosti kako bi se došlo od jednog nasumično odabranog autora do nekog drugog nasumičnog autora (pri čemu su veze radovi koje je neki autor napravio). Druga, možda i bitnija značajka je klasteriranje koje je u velikoj mjeri prisutno u istraživanim mrežama suradnje. Klasteriranje se očituje u tome da će dva autora koja imaju zajedničkog suradnika na nekom svojem radu vjerojatnije međusobno surađivati, nego neka dva nasumično odabrana autora.

Autori Jakawat, Favre i Loudcher [20] predstavljaju model grafa za reprezentaciju bibliografske mreže. Žele proučavati mrežu u više dimenzija te pomoću *Online Analytical Processing* (u daljnjem tekstu: OLAP) analize pokazati podatke iz različitih točki gledišta. U

radu se spajaju podaci iz tri bibliografske baze podataka. Bibliografska baza podataka je zapravo mreža koja nema samo jednu dimenziju već više njih. Kao različite dimenzije možemo promatrati različita svojstva rada: naziv, autori, godina i ostalo. Ako promatramo naše podatke s njihovog motrišta, možemo vidjeti više dimenzija mreže: ime autora (*author_firstname*), prezime autora (*author_lastname*), naziv časopisa (*publication_title*), tip rada – članak ili rad u zborniku (*publication_type*) i ostalo. Nakon što se spoje podaci iz tri bibliografske baze podataka, kreira se više mreža podataka (za suradnju u znanstvenom radu i sl.) te taj graf obogaćuju kockama kako bi mogli izvoditi OLAP analizu. Kocke zapravo sačinjavaju činjenice o radovima ili autorima. Ovim postupkom su autori od više bibliografskih mreža podataka dobili jednu mrežu koju možemo promatrati iz više različitih gledišta.

U [21] ponovno se spominje klasteriranje kao jedna od značajki mreža suradnje. Istražuje se struktura i evolucija mreže suradnje znanstvenika, pri čemu se koriste podaci iz baze DBLP iz razdoblja od 1936. do 2013. godine. U rezultatima analize potvrđen je Lotkin zakon učestalosti objavljivanja novih radova prema kojem nekolicina studenata objavljuje velike količine radova, dok većina studenata objavljuje nekoliko radova. Uz navedeni rezultat dokazana je činjenica koja se spominje u radu „The structure of scientific collaboration networks" vezana uz klasteriranje, odnosno dokazano je da i ovaj skup podataka rezultira visokim koeficijentom klasteriranja.

2.2.1. Pristupi za analizu znanstvene produktivnosti istraživača u Republici Hrvatskoj

U radu autori Grba i Meštrović predstavljaju model složene mreže temeljene na analizi mreže znanstvenika [22]. Analizom ove mreže moguće je vidjeti kvalitetu suradnji između znanstvenika, ali i predvidjeti buduće suradnje.

Autori pri analizi mreže koriste mnogo parametara kako bi uvidjeli sklonost mreže rastu i razvoju. Prosječni stupanj čvora (prosječan broj bridova povezanih s čvorom i), snaga čvora (zbroj težina svih bridova povezanih s čvorom) i gustoća mreže pokazuju veličinu mreže, dok parametar udaljenosti pokazuje koliko je mreža povezana. Nakon definiranja parametra mjerenja, autori identificiraju zajednice unutar mreže pomoću Louvain algoritma.

Autori analiziraju podatke prikupljene sa *Short-Term Scientific Missions* (u daljnjem tekstu: STSMs) pomoću *SemanticKEYword-based Search on Structured Sata Sources* (u daljnjem tekstu: KEYSTONE. STSMs je program razmjena znanstvenika među članicama COST-a (*European Cooperation in Science and Technology*). KEYSTONE omogućava analizu podataka na način da se promatraju veze između različitih znanstvenih područja interesa. Rast mreže kroz pet vremenskih perioda pokazali su putem pet težinskih usmjerenih mreža. Broj

zajednica unutar mreže analizirali su pomoću Louvain algoritma i programa Gephi. U trećem koraku primijenili su algoritme za predviđanje povezanosti mreže: engl. *Resource Allocation Index* (RAI), *Jaccard Coefficient* (JC), *Adamic Adar Index* (AAI) i *Preferential Attachment* (PA). Louvain algoritam bio je dobar za identificiranje zajednica unutar mreže, ali su algoritmi za predviđanje povezanosti mreže dali loše rezultate.

Autori Aparac i Pehar u radu „Information Sciences in Croatia: A View from the Perspective of Bibliometric Analysis of two Leading Journals“ [23] analiziraju razvoj informacijske znanosti na području Republike Hrvatske. Uz razvoj znanosti promatraju i razvoj znanstveno-stručnih časopisa koji se bave temama unutar područja informacijskih znanosti te iste bibliometrijski analiziraju. Uspoređuju časopise „Vjesnik bibliotekara Hrvatske“ i „Informatologia“ prema broju izdanja, broju radova te broju citata u radovima. U radovima u časopisu „Vjesnik“ prosječno je citirano 12.8 ostalih radova, dok je u časopisu „Informatologia“ prosječno citirano 11.5 drugih radova. Osim navedenih parametara, autori su istraživali suradnju autora između ova dva časopisa. Pokazano je da je 28 od 846 autora objavilo članak u oba časopisa te da je suradnja među autorima koji objavljuju unutar istog časopisa veća u časopisu „Informatologia“.

3. Grafovske baze podataka

Grafovske baze podataka su jedna od kategorija *NoSQL* baza podataka. Pojam *NoSQL* opisuje nerelacijske baze koje ne zahtijevaju striktno definiranu (relacijsku) shemu, ali, ovisno o GDBMS-u, ta opcionalnost može biti podržana. Grafovske baze podataka podatke ne pohranjuju u tablice (relacije), već reprezentaciju i obradu podataka temelje na teoriji grafova. Dakle, podaci su u grafovskim bazama podataka reprezentirani kao čvorovi, tj. vrhovi grafa, a veze između podataka kao bridovi grafa te takva reprezentacija omogućuje modeliranje vrlo složenih veza između podataka. Nadalje, čvorovima i vezama u grafu mogu se dodijeliti pojedine oznake (engl. *label*). Na oznake čvorova možemo gledati kao na svojevrsni tip čvora ili veze, gdje svakom čvoru ili vezi može biti dodijeljen veći broj oznaka. U daljnjem će tekstu umjesto vrha i brida grafa biti korišteni pojmovi čvor (engl. *node*) i veza (engl. *relationship*) respektivno, ali po potrebi će biti napomenuto drugačije radi boljeg razumijevanja konteksta.

Kada bismo prevodili relacijski model podataka u grafovski, onda bi nazivi relacija (tablica) u grafovskom modelu bile oznake, redovi tablice bi bili čvorovi, a veze između primarnih i vanjskih ključeva tablica bi bili bridovi grafa.

Podaci koji pripadaju nekom čvoru nazivaju se njegova svojstva (engl. *property*), a ekvivalentni su atributima u relacijama. Kako relacijske baze podataka imaju striktno definiranu relacijsku shemu, grafovske baze omogućuju čvorovima istih oznaka posjedovanje različitih svojstava, što u relacijskim bazama podataka nije moguće jer, ako se nad nekom relacijom stvori novi atribut (stupac), tada svaki zapis u njoj mora posjedovati vrijednost za taj atribut. No, kao što je ranije rečeno, postoje GDBMS-ovi koji omogućuju postavljanje ograničenja na čvorove koja nalažu da npr. čvorovi iste oznake moraju posjedovati ista svojstva.

Posebnost grafovskih baza podataka koja se osobito ističe nad relacijskim su veze između podataka. Grafovske baze uvelike ubrzavaju izvršavanje upita nad podacima vezanim kompleksnim vezama bez upotrebe po performanse skupih „JOIN“ upita iz SQL-a [24]. Ta prednost proizlazi iz činjenice da su pojedine instance podataka (čvorovi) i veze izravno povezani na fizičkoj razini pa ne zahtijevaju dugotrajna iteriranja kroz tablice. No, takva povezanost podataka omogućuje analizu i obradu cijelih mreža podataka koje se vrlo lako mogu prikazati grafički. Upravo ta grafička reprezentacija grafova omogućuje vrlo brzo i intuitivno oblikovanje modela podataka u grafovskim bazama podataka. Radi se o tome što se model podataka vrlo lako može prevesti s papira u grafovsku bazu podataka, tj. prijenos odnosa (veza) između entiteta u model je znatno lakši nego kod relacijskih baza jer ne

zahtijeva normalizaciju koja narušava razumljivost krajnjeg modela, već on ostaje intuitivno jasan - poput skice na papiru.

Nadalje, iz karakteristike *NoSQL* baza podataka slijedi da nemaju sve tehnologije ni približno toliko strukturiran deklarativni jezik kao što je *SQL*, što povlači i činjenicu da ne postoji jedan univerzalni upitni jezik za *NoSQL* tehnologije. Sam *SQL* dosta je razumljiv i sličan ljudskom jeziku pa tehnologija koja podržava takav jezik, koji je k tome dobro strukturiran, u prednosti je pred ostalima. Jedna takva tehnologija je *Neo4j* grafovska baza podataka u kojoj se koristi deklarativni upitni jezik *Cypher* te je ona korištena u ovom radu.

3.1. *Neo4j*

Kao što stoji u [25], *Neo4j* je predvodnik u području grafovskih baza podataka. To je višeplatformska tehnologija otvorenog koda koju razvija tvrtka *Neo4j* još od ranih godina prošlog desetljeća. Iako je u početku bila zamišljena kao nadgradnja na postojeće relacijske baze podataka u smislu lakšeg povezivanja podataka, s vremenom su autori *Neo4j*-a svoju tehnologiju u potpunosti odvojili od relacijskih te ju odlučili iz temelja izgraditi kao izvornu, *NoSQL* grafovsku bazu podataka [26]. Riječ *izvorna* u ovome kontekstu znači da *Neo4j* u potpunosti pohranjuje podatke i veze između njih na temelju matematičkog modela grafova. Ova karakteristika programske arhitekture na kojoj je izgrađen *Neo4j* omogućuje vrlo brze upite nad podacima i još bitnije – nad vezama između podataka [27].

Nadalje, dodatna vrlina *Neo4j*-a jest fizičko odvajanje pohrane podataka o strukturi grafa (čvorovi i veze) od pohrane korisničkih podataka (vrijednosti svojstava čvorova). *Neo4j* tako podatke o čvorovima i vezama dodatno fizički pohranjuje u različite datoteke fiksnih veličina zapisa. Pažljivi čitatelj zaključit će kako to omogućuje brza pronalaženja, npr. traženih čvorova, u vremenu $O(1)$, tj. u konstantnom vremenu koje je potrebno da se izračuna pozicija nekog zapisa o čvoru u datoteci. Kako je *Neo4j* izrađen u programskim jezicima *Java* i *Scala*, implementirani su koncepti jednostruko i dvostruko vezanih lista za ostvarivanje veza između podataka o strukturi grafa te podataka o svojstvima čvorova. Tako se jedan fizički zapis o čvoru, između ostalog, sastoji od pokazivača prema prvoj vezi (bridu) koja je spojena na taj čvor te pokazivača prema prvom svojstvu čvora, dok se jedan fizički zapis o vezi (bridu), između ostalog, sastoji od pokazivača prema čvorovima koje spaja, pokazivača prema zapisu sljedeće i prethodne veze (brida) koje su „spojene“ na početni i završni čvor te pokazivača na tip veze. Iz navedenog se može zaključiti kako *Neo4j* omogućuje vrlo brze obilaske po grafu.

Za postavljanje upita nad grafovskom bazom, odnosno obilazak po grafu, *Neo4j* nudi tri rješenja – tzv. *Core API*, *Traversal Framework* i deklarativni upitni jezik *Cypher. Core API* i *Traversal Framework* su programska sučelja u programskom jeziku *Java* koja služe za precizan obilazak po grafu te zahtijevaju posjedovanje vještina programiranja u samom programskom jeziku *Java*, što nije najjednostavnija opcija koju korisnik može imati [27]. Ovakav pristup nije korišten u izradi ovog rada, stoga neće biti detaljnije objašnjen, dok je upitni jezik *Cypher* objašnjen u poglavlju 3.2.

Sljedeća bitna vrlina *Neo4j*-a jest što ispunjava zahtjeve *ACID* karakteristika jedne transakcijske baze podataka, što mu daje prednost u odnosu na ostale NoSQL baze podataka koje nisu transakcijske [24]. Od te četiri karakteristike zanimljiva je druga, tj. konzistentnost (engl. *consistency*), budući da je *Neo4j NoSQL* baza podataka pa ne zahtijeva striktno definiranu relacijsku shemu, ili u ovom slučaju samo shemu. Jedini zahtjev koji *Neo4j* nalaže, a koji se tiče karakteristike konzistentnosti, jest to da svaka veza ima početni i završni čvor. Nadalje, *Neo4j* nudi opcionalnu shemu koju realizira u obliku indeksa i ograničenja nad svojstvima čvorova i veza. Dostupna ograničenja u *Neo4j* su ograničenja vezana uz jedinestvene vrijednosti svojstava koja osiguravaju da će svi čvorovi ili veze iste oznake posjedovati jedinstvene vrijednosti za određeno svojstvo. Osim toga, postoji i ograničenje koje provjerava postojanje određenog svojstva za sve čvorove iste oznake (engl. *property existence constraint*) i nalaže da svi čvorovi iste oznake moraju posjedovati određeno svojstvo. Dodatno, postoji i ograničenje koje nalaže da svi čvorovi iste oznake moraju imati određena svojstva s jedinstvenom kombinacijom njihovih vrijednosti (engl. *node keys*). Bitno je za napomenuti kako su posljednja dva svojstva dostupna samo u komercijalnom, *Enterprise* izdanju *Neo4j*-a [28].

3.2. *Cypher*

Cypher je deklarativni upitni jezik razvijen za potrebe *Neo4j* grafovske baze podataka, a autor jezika je Andresa Taylor. Iako je *Cypher* nastao prilikom izgradnje *Neo4j*-a, danas je on jedan od najkorištenijih upitnih jezika grafovskih baza podataka u industriji. Njegov razvoj kao tehnologije otvorenog koda, podržan je kroz *openCypher* projekt [29].

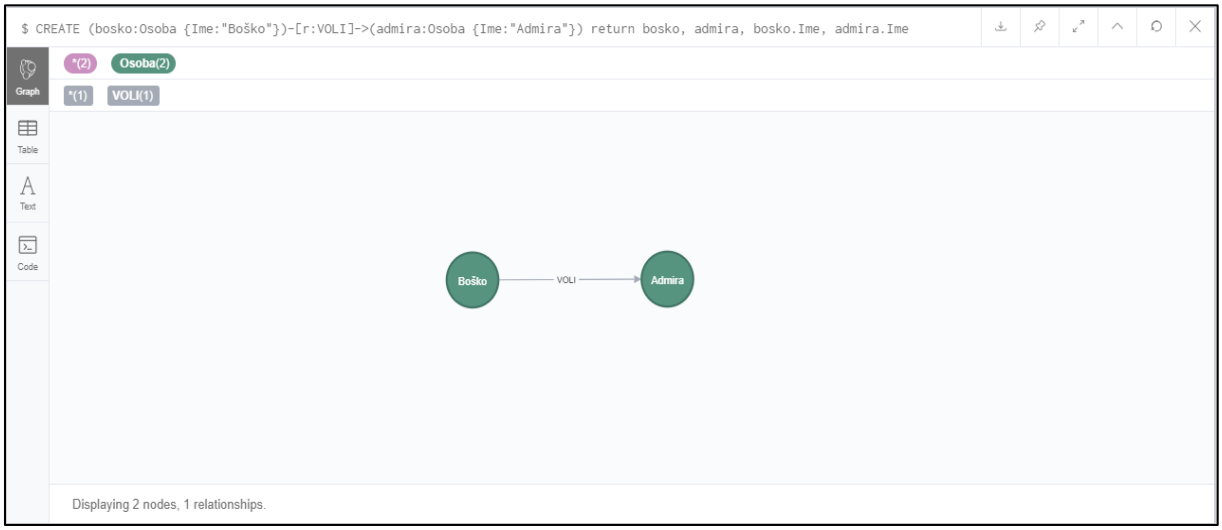
Svojstvo deklarativnosti *Cypher-a* označava da se prilikom postavljanja upita nad grafovskom bazom definira obrazac odnosa između podataka koji se želi pronaći pa je razumljivo da je na razvoj *Cypher-a* uvelike utjecao SQL koji je jednostavan za tumačenje, tj. vrlo je blizak ljudskom jeziku (engl. *human-readable*). Sintaksa *Cypher-a* je također vrlo intuitivna, budući da se veze reprezentiraju pomoću usmjerenih veza i uglatih zagrada, a čvorovi pomoću okruglih zagrada, npr.

```
1. (:čvorA)-[:JE_POVEZAN_S_ČVOROM]->(:čvorB) .
```

Navedena sintaksa razumljiva je bez opširnog objašnjenja, no ona konkretno znači da je neki čvor oznake (tipa) „čvorA“ povezan s čvorom oznake „čvorB“ vezom vrste *JE_POVEZAN_S_ČVOROM* kojoj je ishodišni čvor neki čvor oznake „čvorA“, a odredišni neki čvor oznake „čvorB“. Izraz „**->“** označava usmjerenost veze. Već ovakav jednostavni primjer daje naslutiti koliko se brzo model podataka, na primjer nacrtan na papiru, može prevesti u *Cypher*, tj. koliko se lako grafički predočene veze između podataka mogu „prevesti“ u graf pomoću *Cypher-a*. *Cypher* je također idempotentan, što znači da će višestruko izvršavanje iste naredbe nad nekim skupom podataka promjene izazvati samo prvi put [26].

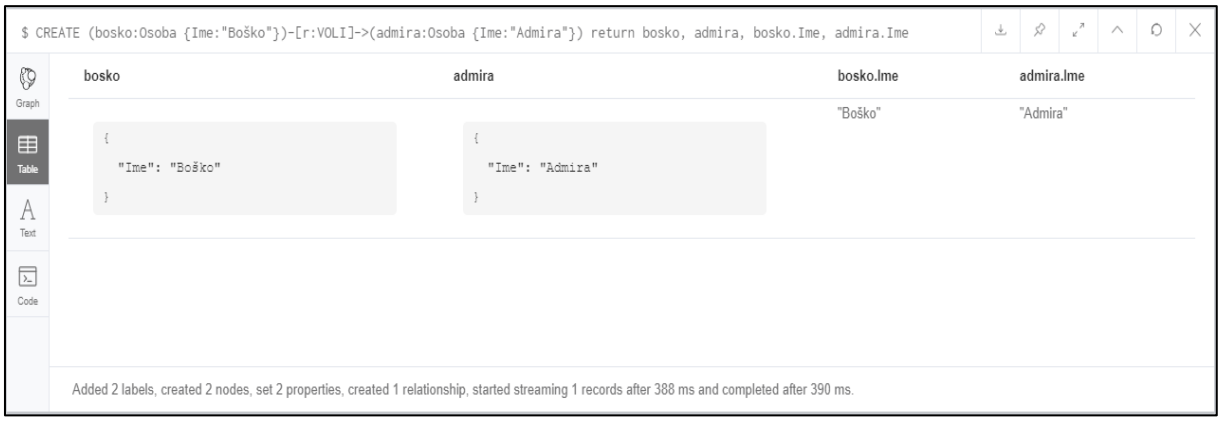
Sljedeći primjer opisuje stvaranje jednostavne veze između dva čvora oznake *Osoba*. Takav će čvor očito predstavljati jednu osobu iz realnog svijeta koja će detaljnije biti opisana svojstvom *Ime*, a dvije će osobe biti povezane vezom tipa *VOLI*. Za stvoriti bilo kakav podatak pomoću *Cypher-a* potrebna nam je ključna riječ „CREATE“**.** Nadalje, *Cypher* omogućuje korištenje pronađenih čvorova u nastavku upita pomoću varijabli. Konkretno, u ovom su primjeru *Osobe* imena Boško i Admira predstavljeni pomoću varijabli *bosko* i *admira*. Te varijable koristimo u „RETURN“ izrazu kako bismo specificirali vrijednosti koje će vratiti upit. U ovom će to slučaju biti ta dva novonastala čvora te vrijednosti njihovih svojstava *Ime*. Slika 1 prikazuje grafički prikaz povezanosti ta dva čvora (postignuto pomoću izraza „RETURN“ *bosko***,** *admira*), dok Slika 2 prikazuje tablični prikaz vrijednosti njihovih svojstava (postignuto pomoću izraza „RETURN bosko.Ime, admira.Ime**“**). Primjer glasi:

```
1. CREATE (bosko:Osoba {Ime:"Boško"})-[r:VOLI]->(admira:Osoba
   {Ime:"Admira"}) RETURN bosko, admira, bosko.ime, admira.ime .
```



Slika 1. Grafički prikaz rezultata upita (Snimka zaslona, 2019)

Pritiskom na opciju *Table* u grafičkom sučelju *Neo4j*-a možemo vidjeti i tablični prikaz podataka.



Slika 2. Tablični prikaz rezultata upita (Snimka zaslona, 2019)

Prva dva stupca na Slici 2., *bosko* i *admira*, predstavljaju tekstualnu reprezentaciju pronađenih čvorova kao popis njihovih svojstava i pripadnih vrijednosti u obliku zapisa *ključ:vrijednost*, koja je vrlo slična JSON formatu, što je karakterističan način zapisivanja podataka u NoSQL bazama podataka. Vidimo da su vraćene vrijednosti "*Boško"* i "*Admira"* znakovni nizovi po tipu podataka, stoga je ovo dobar trenutak za napomenuti koje sve vrste podataka podupire *Cypher* glede vrijednosti svojstava čvorova. To mogu biti:

- numeričke vrijednosti, npr. *Integer* (cjelobrojne) ili *Float* (decimalne),
- znakovni nizovi, tj. *String*,
- logičke, tj. *Boolean*, koje mogu poprimiti vrijednosti *true* ili *false*,
- polja koja se moraju sastojati od niza podataka istog tipa,
- prostorni tip podataka, tj. *Spatial*, koji mogu predstavljati geografske ili Kartezijeve koordinate te mogu prikazati podatke o dvije ili tri dimenzije, i
- vremenski tip podataka, tj. *Temporal*, u koje se pohranjuju podaci o vremenu (*Time, LocalTime*), datumu (*Date)*, kombinaciji vremena i datuma (*DateTime, LocalDateTime*) i duljini trajanja nekog vremenskog intervala (*Duration*).

Sljedeća vrlo bitna ključna riječ u *Cypheru* jest „MATCH" koju koristimo kada u bazi želimo pronaći neki uzorak. Sljedećim ćemo upitom pokušati pronaći postoji li u bazi zapis, tj. čvor oznake *Osoba* koji kao vrijednost svojstva *Ime* ima *Admira*:

```
1. MATCH (o:Osoba) WHERE o.Ime="Admira" RETURN o .
```

Ovdje je predstavljena još jedna ključna riječ *Cypher-a* – „WHERE“. Ona je ekvivalent istoimenoj ključnoj riječi u SQL-u, a u kombinaciji s „MATCH“ predstavlja ograničenje na uzorak koji se pokušava pronaći u bazi. U ovom se slučaju upit odnosi na sve čvorove, ako ih postoji više, koji kao vrijednost svojstva *Ime* imaju *Admira*. Isti bismo rezultat dobili i za upit:

```
1. MATCH (o:Osoba {Ime:"Admira"}) RETURN o .
```

U „WHERE“ klauzuli možemo specificirati i uvjet s kojim želimo ograničiti upit na čvorove kod kojih postoji ili ne postoji neko traženo svojstvo. To možemo učiniti *Cypher* funkcijama *exists()*, odnosno *not(exists()).* Upit kojim bismo željeli vratiti imena svih osoba kod kojih postoji svojstvo *Starost* glasio bi:

```
1. MATCH (o:Osoba) WHERE exists(o.Starost) RETURN o.ime .
```

U našem slučaju, budući da trenutno u bazi ne postoji ni jedna osoba sa svojstvom *Starost*, upit bi vratio nula imena.

Kada bismo željeli našim osobama u bazi pridodati atribut *Starost*, morali bismo ažurirati čvorove. To možemo učini ključnom riječi „SET“. Sljedećim ćemo upitom dohvatiti sve čvorove oznake *Osoba* te im postaviti svojstvo *Starost* na neku vrijednost. Tim se upitom kod čvorova koji već sadrže to svojstvo njegova vrijednost samo postavlja na novu, a kod čvorova koji ga ne sadrže ono se stvara te se postavlja na zadanu vrijednost. Na kraju, vraćamo vrijednosti svojstava *Ime* i *Starost* tih osoba:

```
1. MATCH (o:Osoba) SET o.Starost="25" RETURN o.Ime, o.Starost .
```

Slika 3. Rezultati ažuriranja čvorova u grafu (Snimka zaslona, 2019)

Kada bismo željeli obrisati neki čvor iz baze, koristili bismo ključnu riječ „DELETE“. Recimo da želimo obrisati čvor oznake *Osoba* sa svojstvom *Ime* jednakim *Boško*:

```
1. MATCH (o:Osoba {Ime:"Boško"}) DELETE o.
```

Ovaj upit ne bi uspio radi veze *VOLI* koja traženi čvor povezuje s nekim drugim čvorom. U tom slučaju alternativu predstavlja brisanje veze *Voli* između ta dva čvora:

```
1. MATCH (o:Osoba)-[r:VOLI]->() DELETE r,
```

pa potom brisanje traženog čvora, ili korištenje ključne riječi „DETACH“**:**

```
1. MATCH (o:Osoba {Ime:"Boško"}) DETACH DELETE o,
```

što bi obrisalo i čvor i vezu jer u *Neo4j*-u ne smije postojati veza bez početnog ili završnog čvora.

Ključna riječ koja se također često koristi jest „MERGE“. Ona provjerava postoji li određeni obrazac u grafu te ako on ne postoji – stvara ga. Pri tome omogućuje specificiranje naredbi za oba slučaja klauzulama „ON MATCH“ za slučaj kada obrazac postoji te „ON CREATE**“** za slučaj kada se specificirani obrazac stvara. „MERGE“ se najčešće koristi za stvaranje veza između već postojećih čvorova. Sljedeći primjer demonstrira stvaranje novog čvora oznake *Osoba* te nove vrste veze *JE_PRIJATELJ* kojim će se novonastali čvor povezati s već postojećim:

```
1. MATCH (admira:Osoba) WHERE admira.Ime="Admira" MERGE
2.  (ana:Osoba{Ime:"Ana", Starost:"21"})-[:JE_PRIJATELJ]->(admira) .
```

Slika 4 prikazuje izgled grafa nakon provedenog posljednjeg upita. Već se sada može primijetiti koliko je lako stvoriti i razumijeti veze između podataka u grafovskoj bazi. Podsjećamo, sličan bi upit u relacijskim bazama zahtijevao skupe „JOIN“ operacije i višestruka iteriranja po tablicama. U [24] autori Vukotic i Watt u poglavlju *1.2 Graph data in a relational database* detaljnije opisuju i analiziraju razlike ovakvih vrsta upita u relacijskim bazama podataka naspram onih u grafovskim.

Slika 4. Izgled grafa nakon provedenog „MERGE“ upita (Snimka zaslona, 2019)

Prilikom izrade baze podataka i skladišta, za pripremu ovog rada korišteni su i koncepti indeksa. Kao što je već ranije rečeno, indeksi su podržani u *Neo4j* grafovskoj bazi podataka, a općenita sintaksa *Cypher* naredbe kojom se stvaraju glasi:

```
1. CREATE INDEX ON :oznakaČvora/veza(svojstvo),
```

odnosno,

```
1. REMOVE INDEX ON :oznakaČvora/veza(svojstvo) .
```

Neo4j podržava stvaranje indeksa i na svojstvima čvorova i na svojstvima veza, a oni konkretno služe za brže izvršavanje upita.

Iz navedenih primjera vidimo kako *Cypher* podržava *CRUD* naredbe (engl. *Create, Read, Update, Delete*), koncepte jezika za rad s podacima, ali on također obiluje i ostalim funkcijama, klauzulama i mogućnostima čije bi nabrajanje premašilo opseg ovog rada. Dokumentacija aktualne verzije *Cypher-a* v3.5 dostupna je na službenim stranicama *Neo4j*-a, odnosno na poveznici [30]. Ovime su nabrojene sve osnovne funkcionalnosti *Cypher-a* koje su većinom bile korištene u izradi ovog rada.

4. Skladišta podataka

U bazu podataka smještaju se svi podaci koji se prikupljaju unutar određene organizacije te ih organizacija svakodnevno koristi. Nad istim podacima se povremeno provode upiti s ciljem generiranja određenih izvješća. Izvršavanje takvih upita nad bazom podataka koji mogu imati višemilijunske podatke vremenski je neefikasno. Zbog toga se kreira model skladišta podataka koji će sadržavati samo najosnovnije informacije iz baze podataka za kreiranje potrebnih izvješća. Na taj način se isključuju svi nepotrebni podaci te se vremensko trajanje kreiranja izvješća drastično smanjuje. Dakle, baza podataka služi za svakodnevnu pohranu, izmjenu i pristup svim podacima organizacije, dok skladište podataka služi za povremenu pohranu najvažnijih podataka za izradu određenih izvješća. Podaci u skladištu podataka su namijenjeni isključivo za izradu izvješća te se ne izmjenjuju niti brišu [31].

Najčešći logički model skladišta podataka je shema zvijezde (engl. *star schema*) u kojem postoje dvije vrste tablica [32]:

- Činjenične tablice kojima se definira što mjerimo
- Dimenzijske tablice kojima definiramo prema čemu mjerimo

Činjenične tablice sadrže numeričke podatke koje nazivamo „činjenice“ nad kojima se temelji izrada svih izvješća. Ti podaci daju odgovor na pitanje „što mjerimo?“ te su glavni rezultat izvješća. Međutim, podaci iz činjeničnih tablica nemaju veliko značenje ako nemaju kontekst koji se veže uz samu činjenicu (odgovori na pitanja kada, tko, gdje?). Odgovore na ta pitanja pružaju podaci iz dimenzijskih tablica. Dakle, dimenzijske tablice su većinom tekstualni podaci koji nadopunjuju podatke iz činjeničnih tablica.

Činjenična tablica je povezana vezom N - 1 (više naspram jedan) s više dimenzijskih tablica od kojih svaka opisuje jednu dimenziju činjenice (vremensku ili opisnu). Primarni ključ činjenične tablice je složeni primarni ključ sastavljen od vanjskih ključeva dimenzijskih tablica. Uz složeni primarni ključ, činjenična tablica sadrži i nekoliko numeričkih atributa koji detaljnije opisuju činjenicu. Model skladišta podataka može imati više činjeničnih tablica, no svaka činjenična tablica mora biti povezana vezom N – 1 s dvije ili više dimenzijskih tablica koje joj pridodaju kontekst. Opisani model skladišta podataka je temeljen na modelu zvijezde u kojemu su dimenzijske tablice vrhovi, a činjenična tablica samo tijelo zvijezde.

4.1. Prednosti skladišta podataka

Baze podataka su dizajnirane da svakodnevno prikupljaju informacije koje pojedina organizacija kreira prilikom svog poslovanja. Zbog toga se prilikom izrade modela baze podataka fokusira na smanjenje redundantnosti podataka kako ne bi došlo do nepotrebne pohrane istog podatka nekoliko puta. Takav model smanjuje količinu podataka koji se pohranjuju te pojednostavljuje unos, izmjenu i brisanje zapisa. No ukoliko se nad podacima u bazi podataka izvrši upit za kreiranje izvješća koji uzima u obzir veliki broj zapisa, tada postaju vidljivi nedostaci baze podataka. Budući da su takve baze podataka namijenjene za svakodnevni unos podataka, nije preporučljivo da se nad njima se izvode zahtjevni upiti koji bi na duži vremenski period opteretili bazu podataka te samim time blokirali upis novih zapisa za druge korisnike. Također je nerijetko da pojedina organizacija pohranjuje podatke u nekoliko različitih baza podataka, čime se otežava pristup svim informacijama koje su potrebne za izradu traženih izvješća.

Skladišta podataka se dizajniraju na način da obuhvate sve važne informacije organizacije za kreiranje određenih izvješća. Dakle, skladišta podataka sadrže samo najvažnije podatke pohranjene u obliku pogodnom za izvršavanje složenih upita čime se ubrzava i olakšava generiranje raznih izvješća nad velikim količinama podataka. Kako se u skladište podataka spremaju samo obrađeni podaci od strane stručne osobe ili alata za transformaciju podataka, svi podaci su agregirani i bez anomalija što eliminira potrebu ažuriranja i pregledavanja podataka.

S napretkom organizacije povećava se količina zapisa koja se svakodnevno unosi u baze podataka što rezultira velikom broju informacija rasprostranjenih na nekoliko baza podataka. Takav sustav funkcionira na operativnoj razini no problemi se javljaju na strateškoj razini gdje postaje nemoguće donositi strateške odluke bez važnih izvješća. Zbog toga prednosti koje skladišta podataka ima nad bazama podataka postaju sve važniji kako organizacija napreduje te s vremenom postaje važna i bitna stavka u odlučivanju. Skladišta podataka obuhvaćaju važne agregirane podatke iz svih baza podataka te omogućuju brzo i jednostavno generiranje izvješća na temelju svih dostupnih informacija organizacije. Dakle skladište podataka ujedinjuje sve strateški važne podatke na jednom mjestu te ih koristi za izradu izvješća ključnih u provođenju analiza.

5. Materijali i metode

U sklopu rada provedeno je eksperimentalno istraživanje u kojem su tehnologije grafovskih baze podataka i skladišta podataka primijenjene za reprezentaciju i analizu znanstvenih publikacija hrvatskih istraživača. U radu je mreža publikacija reprezentirana kao graf pohranjen u grafovskom skladištu nad kojim se vršila kvantitativna analiza publikacija. Za izradu modela skladišta podataka korišteno je dimenzijsko modeliranje detaljno opisano u poglavlju 5.2. Obrada i učitavanje podataka je vršena prema engl. *Extract-Transform-Load* (u daljnjem tekstu: ETL) modelu prema kojemu se podaci prvo dohvaćaju iz izvora podataka, nakon čega se vrši transformacija nad dohvaćenim podacima kako bi se uklonile sve anomalije. Zadnji korak ETL procesa je učitavanje obrađenih podataka u skladište podataka. Manipulacija nad podacima je izvršena u sustavu za upravljanje grafovskim bazama podataka *Neo4j*. Za prikaz podataka nakon obrade u *Neo4j* GDMBS-u korišteno je ugrađeno *Neo4j* korisničko sučelje te sučelje izrađeno u *Flask* razvojnom okviru.

5.1. Analiza zahtjeva vezanih uz domenu reprezentacije znanstvenih publikacija istraživača

Jedna od postavljenih hipoteza pri izradi ovog rada bila je prikazati podatke o mreži znanstvenih publikacija hrvatskih istraživača pomoću grafovskog skladišta podataka nad kojim ćemo izvršavati upite. Model grafovskog skladišta podataka detaljno je opisan u poglavlju 5.4. i prikazan na Slici 6. Nad izrađenim modelom u sustavu *Neo4j* izvršavat ćemo sljedeće upite:

- Deset autora koji su izdali najviše znanstvenih radova ovisno o godini i znanstvenom području
- Publikacije (časopisi ili zbornici), ovisno o znanstvenom području, u kojima je objavljeno najviše znanstvenih radova ili najviše CC radova
- Broj znanstvenih radova i broj CC radova prema području

5.2. Model skladišta podataka

Prema R. Kimball-u [33] dimenzijsko modeliranje skladišta podataka je tehnika modeliranja skladišta podataka čiji je cilj dostaviti bitne podatke poslovnim korisnicima u jednostavnom i razumljivom obliku uz što kraći vremenski period. Dimenzijsko modeliranje definira četiri koraka prilikom izrade modela skladišta podataka:

1. Odabrati proces koji se modelira
2. Odrediti značenje podatka u činjeničnoj tablici
3. Odabrati dimenzije koje se odnose na redak u činjeničnoj tablici
4. Identificirati činjenice kojima će se popuniti činjenična tablica

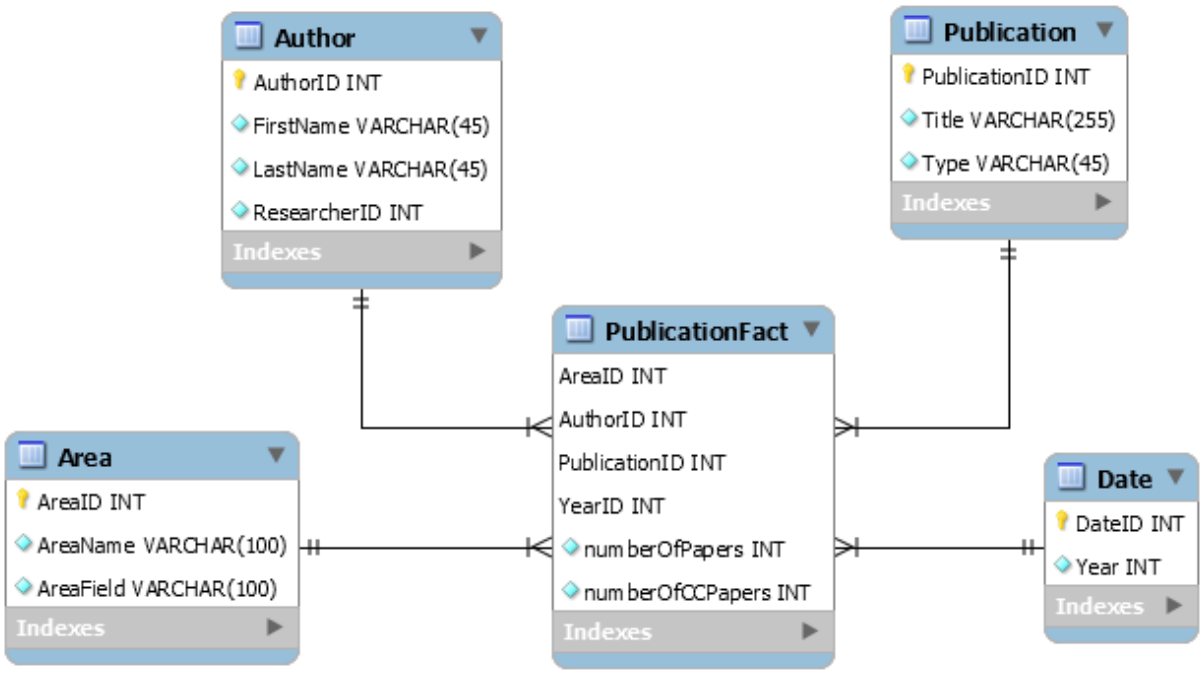
Odabrani proces koji će se modelirati u ovom radu je mreža znanstvenih publikacija hrvatskih istraživača. Kako su dostupni samo podaci o znanstvenim radovima s imenima autora, analiza će se bazirati na broju objavljenih radova od strane pojedinog istraživača, dok se broj referenci određenog rada u drugim radovima neće uzeti u obzir zbog nedostatka podataka. Dakle, jedan podatak u činjeničnoj tablici će reprezentirati jedan rad od strane jednog istraživača.

Odabir dimenzija koje se odnose na pojedini redak u činjeničnoj tablici određen je na temelju dobivenih podataka iz CROSBI baze podataka te su odabrane sljedeće četiri dimenzijske tablice: „Date“ (vremenska dimenzija - godina izdavanja), „Author“ (istraživač), „Publication“ (znanstveni rad - detalji publikacije) i „Area“ (znanstveno područje). Zadnji korak je identifikacija činjenica kojima će se popuniti sama činjenična tablica. Kako se svi opisno važni podaci nalaze u dimenzijskim tablicama, za atribute činjenične tablice odabrani su *numberOfCCPapers* i *numberOfPapers* koji predstavljaju broj znanstvenih radova sa CC kategorizacijom te ukupan broj znanstvenih radova odabranog istraživača respektivno.

5.3. Izrada modela relacijskog skladišta podataka

U ovom radu analizirat će se produktivnost istraživača i znanstvenika prema broju radova koje je pojedini istraživač/znanstvenik objavio. Prikupljeni podaci iz CROSBI baze podataka su grupirani u četiri dimenzijske tablice. Dimenzijska tablica „Author“ opisuje koji autor je napisao odabrani znanstveni rad, „Area“ opisuje u kojem znanstvenom području i polju rad objavljen, „Date“ označava vremensku komponentu godine objave rada te „Publication“ sadrći podatje o nazivu rada i tipu rada (rad u časopisu ili zborniku konferencije).

Nakon definiranih dimenzijskih tablica kreirana je činjenična tablica „PublicationFact“ koja je povezana identificirajućim vezama s dimenzijskim tablicama. Vanjski ključevi u tablici „PublicationFact“ čine složeni primarni ključ. Činjenična tablica sadrži i dodatne atribute *numberOfPapers* i *numberOfCCPapers* koji označavaju broj običnih i CC radova koje je autor objavio respektivno.



Slika 5. Model relacijskog skladišta podataka (Samostalna izrada, 2019)

Na Slika 5 je prikazan model relacijskog skladišta podataka na kojem je vidljiva shema zvijezde [32], gdje je činjenična tablica „PublicationFact“ tijelo zvijezde, dok su četiri dimenzijske tablice vrhovi zvijezde koji podacima iz činjenične tablice pridodaju kontekst.

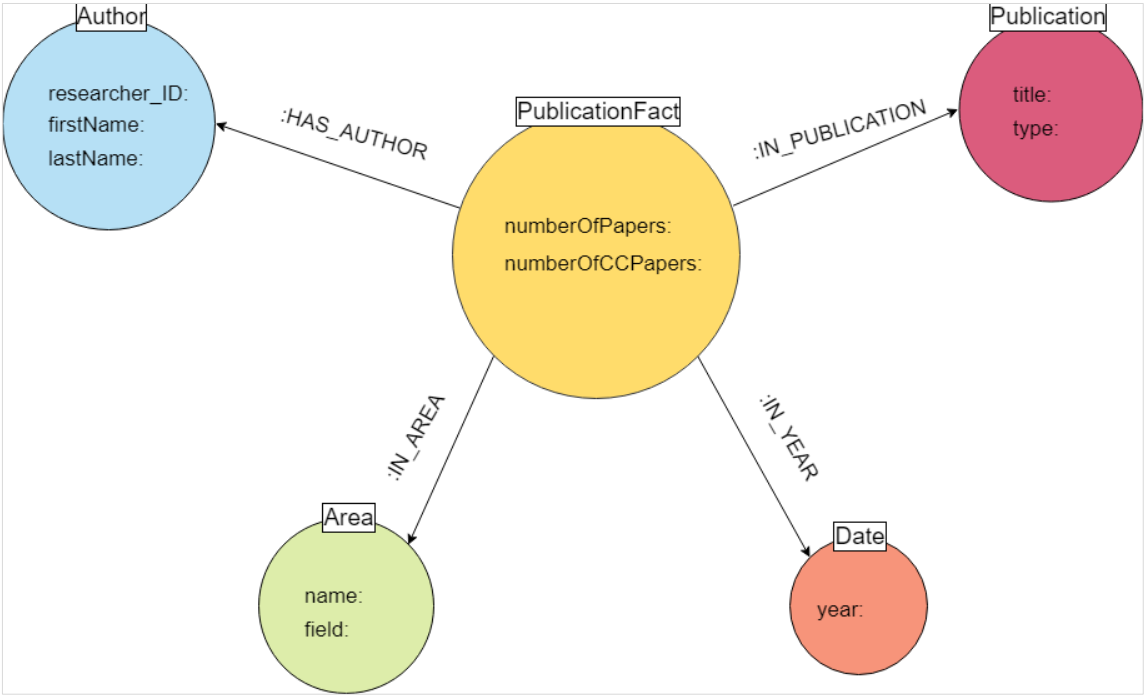
5.4. Izrada modela grafovskog skladišta podataka

Iako je razvoj NoSQL tehnologija uvelike pridonio razvoju GDBMS-ova te popularnosti grafovskih baza podataka, grafovska skladišta podataka su još uvijek, moglo bi se reći, nedovoljno istražena tehnologija. Autori Rahim, Chakraborty, Debnath i Debnath [34] u svom radu analiziraju modeliranje grafovskog skladišta podataka te njegovu konverziju u objektno-orijentirani model. U realnom svijetu su problemi često višedimenzionalne prirode te je potrebno provoditi kompleksne upite nad podacima kako bi se provela kvalitetna analiza istih. Već ranije spomenuta tehnologija, OLAP, korištena je i u ovom radu kao središnja tehnologija pri radu s skladištima podataka. Autori u radu predstavljaju model grafovskog skladišta podataka za *online t*rgovački centar te donose skup pravila kako višedimenzionalni podatkovni model pretvoriti u nekoliko semantičkih grupa i pomoću njih realizirati jednu od najpoznatijih shema skladišta podataka – zvjezdanu shemu. Primijećeno je da je izrađen model grafovskog skladišta podataka za prikaz produktivnosti hrvatskih istraživača izrađen na isti način kao i model u radu spomenutih autora, odnosno, postoji središnji činjenični čvor („*PublicationFact*“) te ostali čvorovi posjeduju veze isključivo s činjeničnim čvorom.

Autori Liu i Vitolo u svom radu „Graph Data Warehouse: Steps to Integrating Graph Databases into the Traditional Conceptual Structure of a Data Warehouse“ [35] problemu su pristupili na način da kreiraju aplikacijsko programsko sučelje (engl. *Application Programming Interface*, API) u Javi koje će pristupati grafovskoj bazi podataka izrađenoj u *Neo4j*-u te provoditi tipične SQL naredbe: „SELECT“, „UPDATE“, „INSERT“ i mnoge druge. Cilj rada je uključiti tehnologiju grafovskih baza podataka u skladište podataka te pokazati prednosti rada s takvim vrstama skladišta podataka.

Model grafovskog skladišta podataka vidljiv je na slici u nastavku. Model se sastoji od sljedećih čvorova: „Author“, „Publication“, „Date“, „Area“ i „PublicationFact“. Čvor „Author“ predstavlja pojedinog hrvatskog znanstvenika ili istraživača te posjeduje sljedeća svojstva: *researcher*_ID (jedinstvenu oznaku autora), *firstName* i *lastName*. Čvor „Publication“ označava časopis ili zbornik te posjeduje svojstvo naslov i tip. Svojstvo *type* omogućuje raspoznavanje publikacije, točnije, u svojstvu *type* je zapisan znak 'J' ako je publikacija časopis, tj. znak 'C' ako je publikacija zbornik s konferencije. Čvor „Date“ predstavlja vremensku komponentu unutar grafovskog skladišta podataka. Čvor „Area*“* sadrži dva svojstva: *name* i *field,* gdje *name* označava znanstveno područje (društvene znanosti, tehničke znanosti i ostalo), a *field* označava polje unutar područja, npr. polje Informacijske i komunikacijske znanosti. Središnji čvor „PublicationFact, je čvor u kojem za određenog autora, publikaciju (časopis ili zbornik),

određenu godinu i određeno znanstveno polje agregiramo broj objavljenih radova i broj objavljenih CC radova.



Slika 6. Model grafovskog skladišta podataka (Samostalna izrada, 2019)

Model grafovskog skladišta podataka gotovo je jednak modelu relacijskog skladišta podataka, odnosno, sve tablice relacijskog modela skladišta podataka (činjenične i dimenzijske) predstavljaju čvorove u graf skladištu. „PublicationFact“, kao što i sam naziv čvora govori je činjenični čvor, dok su ostali čvorovi ekvivalentni dimenzijskim tablicama.

5.5. Prikupljanje (ekstrakcija) podataka

Kao izvor podataka korištena je mrežna bibliografska baza podataka “Hrvatska znanstvena bibliografija” popularnija pod nazivom “CROSBI” [36]. Navedena baza podataka izvrsno pokriva tekuća znanstvena istraživanja i znanstvenu publicistiku u Hrvatskoj. Važno je napomenuti da je posljednji dohvat podataka iz „Hrvatske znanstvene bibliografije” izvršen 6. travnja 2019. godine te će u radu biti obuhvaćeni svi znanstveni radovi na bazi podataka CROSBI do navedenog datuma.

5.5.1. Preuzimanje podataka

Mrežna bibliografska baza podataka „Hrvatska znanstvena bibliografija” nudi pregledavanje zapisa po određenim kategorijama te se na taj način može pronaći pregled po znanstveniku, ustanovi, projektu, znanstvenom području ili, pak, po vrsti rada. Svaki od spomenutih pregleda nudi preuzimanje prikazanih zapisa u CSV datoteci po 10 000 zapisa. No, kako za izradu skladišta podataka treba sadržavati sve podatke, bilo je potrebno filtrirati CROSBI bazu podataka tako da prikaže manje od 10 000 zapisa te na taj način preuzeti sve podatke u datotekama po 10 000 zapisa. Nakon filtriranja i preuzimanja cijele CROSBI baze podataka, bilo je potrebno spojiti 72 preuzete datoteke u jednu koja će se koristiti za transformaciju i učitavanje u skladište podataka. Međutim, u preuzetim podacima je uočen nedostatak identifikacijskog broja svakog autora koji je bio potreban ukoliko želimo jednoznačno identificirati svakog istraživača te eliminirati mogućnost da skladište podataka detektira dva znanstvenika s istim imenom i prezimenom kao jedan zapis. Osim toga, preuzete CSV datoteke nisu sadržavale ni bibliografske jedinice naslova radova čime je jednoznačna identifikacija naslova također bila nemoguća.

Podatke koje nije bilo moguće dohvatiti jednostavnim preuzimanjem CSV datoteka s mrežne bibliografske baze podataka „Hrvatska znanstvena bibliografija„ bilo je potrebno dohvatiti pomoću programskog kôda. Kombinacijom jednostavnog preuzimanja datoteka te dohvaćanja podataka programskim kôdom formiran je željeni skup podataka.

5.5.2. Dohvaćanje podataka

Proučavajući mrežnu bibliografsku bazu podataka „Hrvatska znanstvena bibliografija" uočen je sustavan način organizacije sadržaja odnosno samih radova. Svakom radu posvećena je zasebna mrežna stranica sa svojom jednoznačnom poveznicom, primjerice rad „Diskretna matematika s teorijom grafova" nalazi se na sljedećoj poveznici: https://www.bib.irb.hr/216175. Detaljnijim istraživanjem poveznice uočeno je kako je upravo broj na kraju same poveznice taj koji jednoznačno određuje svaki naslov unutar same mrežne bibliografske baze podataka, te predstavlja jednoznačnu identifikaciju svakog rada. Ova činjenica je uvelike pomogla prilikom dohvata podataka koji su nedostajali. Iteracijom kroz generirane poveznice dohvaćeni su konzistentno strukturirani HTML dokumenti sa svim potrebnim podacima.

Problem je riješen pomoću programskog jezika *Python* koji se zajedno s određenim paketima pokazao kao vrstan odabir. Važno je napomenuti da su sve *Python* skripte pisane unutar *Jupyter Notebook*-a zbog preglednosti prilikom programiranja. Prvo je, dakle, bilo potrebno saznati brojevni raspon u kojem se kreću brojevi bibliografskih jedinica kako bi znali koje stranice odnosno naslovi će se dohvaćati. Koristeći *Python* paket *requests* upućen je GET zahtjev poslužitelju za određenu web stranicu odnosno pojedini naslov. Ovisno o povratnom statusnom kôdu moglo se vidjeti postoji li naslov na toj web stranici, odnosno postoji li naslov čija bibliografska jedinica nosi taj broj. Primjerice, ukoliko je poslužitelj vratio statusni kôd *200 (OK)* zaključeno je da naslov s tim brojem bibliografske jedinice postoji. Ukoliko bi pak poslužitelj vratio statusni kôd *404 (NOT FOUND),* bilo bi suprotno, dakle naslov ne postoji. S obzirom da je samim promatranjem poveznica zaključeno kako je većina (kasnije se pokazalo sve) bibliografskih jedinica u rasponu od 1 do 1 000 000, bilo je potrebno samo provjeriti postoji li radova s bibliografskom jedinicom većom od 1 000 000. Za navedeni postupak dohvaćanja podataka napisana je sljedeća skripta:

```
1. Import pandas as pd
2. Import request
3. %config IPCompleter.greedy=True
4.
5. urlMap = pd.DataFrame(columns=[„URL", „statusCode"])
6. for i in range(1000000, 2000000):
7.          ulr = https://www.bib.irb.hr/{}.format(i)
8. statusCode = request.get(url).status_code
9. urlMap = urlMap.append({„URL": url, „statusCode": statusCode,
   ignore_index = True)
10.  urlMap[urlMap[„statusCode"] == 200]
```

S obzirom da skripta nije proizvela nikakav izlaz, zaključeno je da nema bibliografskih jedinica kojima su pridruženi brojevi iz raspona većeg od 1000000. Napisani kôd zapravo djeluje na sljedeći način. Prvotno se uključuje paket *pandas* te modul *requests*. Modul *requests* omogućuje slanje GET zahtjeva te mnogo više od toga, no za svrhu ovog programskog kôda oni su i više nego dovoljni – dok paket *pandas* suštinski omogućuje rukovanje podacima na znatno efikasniji te jednostavniji način. Naredbom:

```
1. urlMap = pd.DataFrame(columns=["URL", "statusCode"])
```

stvoren je *pandas* podatkovni okvir (engl. *dataframe*) s atributima *URL* te *statusCode*. U spomenuti podatkovni okvir spremljene su poveznice koje su dohvaćane te statusni kôd koji bi poslužitelj vratio u trenu dohvata. Sljedeći dio kôda napisan je kako bi iterativno generirao poveznice, slao GET zahtjeve poslužitelju te bilježio statusne kôdove poslužiteljevih odgovora zajedno s adekvatnim poveznicama unutar prethodno stvorenog podatkovnog okvira..

```
1. for i in range(1000000, 2000000):
2.   url = "https://www.bib.irb.hr/{}".format(i)
3.   statusCode = requests.get(url).status_code
4.   urlMap = urlMap.append({"URL": url, "statusCode": statusCode},
     ignore_index = True)
```

Rezultati zapisani unutar *pandas* podatkovnog okvira konačno su se nastojali ispisati pritom koristeći uvjet da se ispišu samo oni zapisi čiji je pripadajući statusni kôd bio jednak broju 200, odnosno semantički jednak odgovoru *(OK).* Činjenica da programski kôd nije generirao nikakav izlaz dovela je djelomično do zaključka da ne postoje naslovi s brojevima bibliografskih jedinica unutar danog raspona. Korištena naredba za ispis sa spomenutim uvjetom je:

```
1. urlMap[urlMap["statusCode"] == 200]
```

Važno je napomenuti da ova naredba može dati ispis samo zato što je *Python* kôd pisan unutar *Jupyter Notebook*-a. S obzirom da je provjeren samo brojevni raspon od 1 000 000 do 2 000 000, nije bilo zajamčeno da ne postoje naslovi s brojevima bibliografskih jedinica izvan tog raspona, konkretno iznad gornje granice uzetog raspona. No, ovaj problem bio je razriješen ponešto kasnije.

Kada smo saznali okvirni raspon (od 0 do 1 000 000) brojeva bibliografskih jedinica pomoću prethodno pokazane skripte te ručne observacije samog mrežnog mjesta, započeli smo s prikupljanjem podataka koji su nedostajali. Za provedbu prikupljanja odlučili smo koristiti metodu struganja podataka (engl. *web scraping)* čija je srž zapravo dohvaćanje web stranica te ekstrahiranje relevantnih te željenih podataka iz HTML kôda poslužiteljevog odgovora. Za realizaciju spomenute metode korištene su dotaknute *pandas* i *requests* komponente, no i paket *Beautiful Soup* za estrahiranje podataka iz HTML kôda te paket *re* za regularne izraze.

```
1. import pandas as pd
2. import requests
3. from bs4 import BeautifulSoup
4. import re
```

Nakon uključivanja potrebnih biblioteka te komponenti stvoren je *pandas* podatkovni okvir za potrebe spremanja relevantnih podataka, konkretno matičnog broja autora, imena i prezimena istog te broja bibliografske jedinice te konačno i samog naslova.

```
1. tempDataStorage = pd.DataFrame(columns=["authorId", "name",
   "surname", "title"])
```

U dani *pandas* podatkovni okvir su se iterativnim postupkom upisivali podaci.

```
1. for i in range(0, 1000000):
2.  page = requests.get("https://www.bib.irb.hr/{}".format(i))
3.  soup = BeautifulSoup(page.content, 'html.parser')
4.
5.  title = soup.find("h2", class_="title", text=True)
6.  authors = soup.find("div", class_="authors", text=True)
7.  if(title != None):
8.   title = title.text
9.   authors = authors.text.split(";")
10.  authors = dict()
11.  for link in soup.findAll('a', attrs={'href':
   re.compile("znanstvenici")}):
12.   authorId = link.text.replace("(","").replace(")","").split(",")[1]
13.   nameSurname =
   link.text.replace("(","").replace(")","").split(",")[0].split(" ")
14.   authors[authorId] = nameSurname
15.
16.  for authorId, author in authors.items():
17.   tempDataStorage = tempDataStorage.append({"authorId": authorId,
18.   "name": author[0],
19.   "surname": author[1],
20.   "publicationId": i,
21.   "title": title}, ignore_index = True)
```

Svaki korak iteracije otpočeo bi dohvaćanjem sadržaja poveznice. Poveznica je svakom koraku bila naravno drugačija jer se generirala uz pomoć brojača iteracije. Dohvaćeni sadržaj bi se potom parsirao kako bi se jednostavnije manipuliralo istim.

```
1. page = requests.get("https://www.bib.irb.hr/{}".format(i))
2. soup = BeautifulSoup(page.content, 'html.parser')
```

Iz dohvaćenog sadržaja nastojalo se potom dohvatiti naslov. Naslov kao takav je u HTML kôdu bio zapisan kao naslov druge razine te je pripadao klasi „title" - spomenuta činjenica iskorištena je prilikom dohvata.

```
1. title = soup.find("h2", class_="title", text=True)
```

Ukoliko je naslov uspješno dohvaćen, izvršavao se ostatak skripte, dok se u protivnom prešlo na dohvaćanje sljedeće poveznice. Naime, nemogućnost dohvaćanja naslova bi označila da naslov s tim brojem bibliografske jedinice ne postoji. Kada bi se naslov pak uspješno dohvatio, njegov unutarnji HTML (konkretni tekst, engl. *inner HTML*) bi se spremio za daljnje korištenje unutar koraka iteracije. S obzirom da su autori zapisani kao poveznice konzistentno na svim mrežnim stranicama koje su bile od interesa te unutar svojih putanja imaju riječ *znanstvenici* (primjerice: https://www.bib.irb.hr/pregled/znanstvenici/200682), moglo se uz pomoć tih svojstava lakše dohvaćati iste. Pomoću regularnih izraza tražene su one poveznice koje nose prethodno opisana svojstva. Iterativnim postupkom te određenim metodama je unutarnji HTML dohvaćenih poveznica dodatno oblikovan kako bi bio u skladu s potrebama. Primjerice, izbačene su zagrade koje su bile prisutne te se samo puno ime autora dijelilo na ime i prezime. Oblikovani unutarnji HTML, odnosno sam tekst od interesa je potom spremljen u *Python* rječnik (engl. *dictionary*). Iterativnim postupkom kroz spomenuti rječnik spremljeni su relevantni podaci u prethodno stvoreni *pandas* podatkovni okvir, čime bi se okončao korak iteracije.

```
1. if(title != None):
2.  title = title.text
3.  authors = dict()
4.  for link in soup.findAll('a', attrs={'href':
   re.compile("znanstvenici")}):
5.   authorId = link.text.replace("(","").replace(")","").split(",")[1]
6.   nameSurname =
   link.text.replace("(","").replace(")","").split(",")[0].split(" ")
7.   authors[authorId] = nameSurname
8.
9.  for authorId, author in authors.items():
10.  tempDataStorage = tempDataStorage.append({"authorId": authorId,
11.   "name": author[0],
12.   "surname": author[1],
13.   "publicationId": i,
14.   "title": title}, ignore_index = True)
```

Nakon što su bili dohvaćeni svi željeni podaci (kada je glavna iteracija od 0 do 1 000 000 završila svoje izvršavanje), isti su bili zapisani u CSV datoteku. Konkretno, sadržaj *pandas* podatkovnog okvira je zapisan u CSV datoteku. Korišteno kodiranje znakova prilikom zapisa je bilo „UTF-8”, a index argument metode *to_csv* je postavljen na *false* kako bi se izbjeglo zapisivanje *pandas* indeksa koji nisu relevantni.

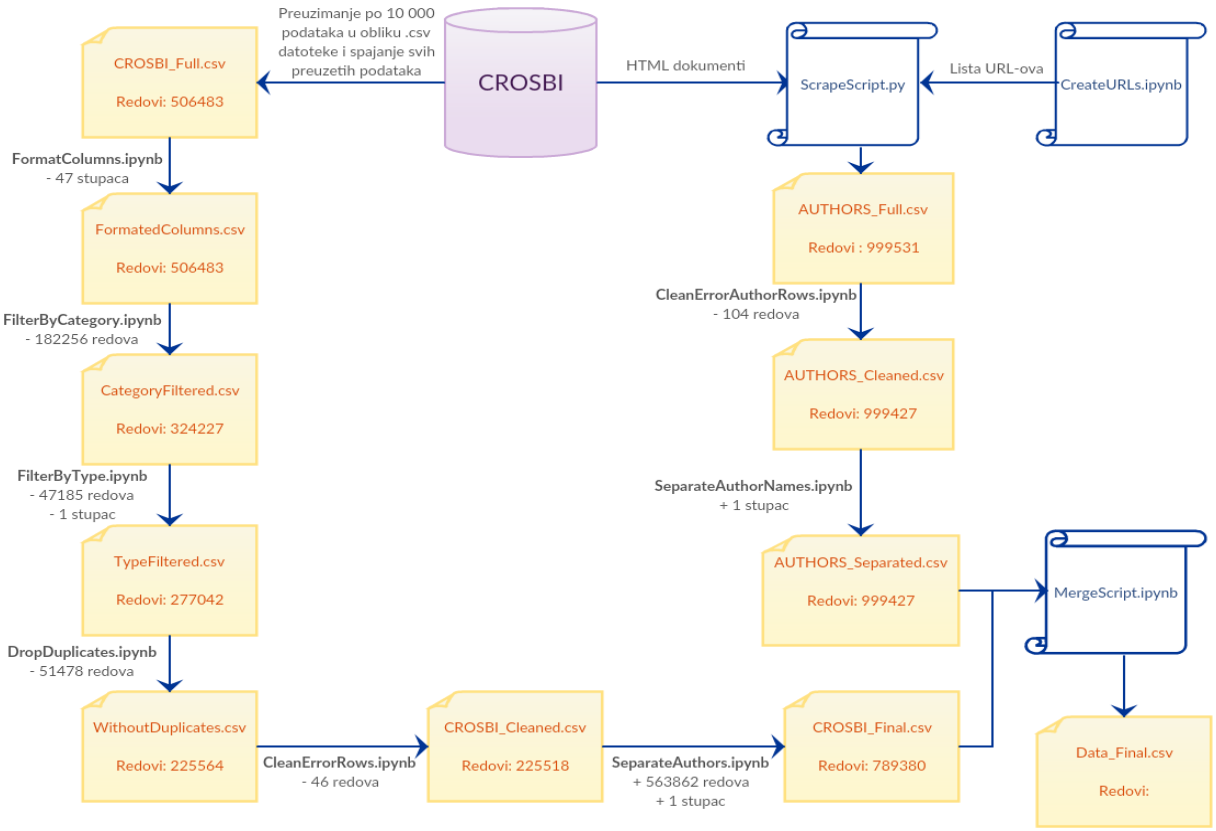
5.6. Transformacija podataka

Prilikom izrade modela skladišta podataka bilo je određeno s kojim podacima se skladište treba popuniti, no kod preuzimanja podataka uočen je problem nedostatka jednoznačne identifikacije svih znanstvenika, zbog čega je bilo potrebno naknadno dohvatiti identifikacijske brojeve znanstvenika. Iz tog razloga je potrebno transformirati dva skupa podataka te iz jednog skupa preuzeti identifikacijske brojeve znanstvenika i prebaciti u drugi skup kako bi se svi podaci nalazili na jednom mjestu.

Preuzimanje podataka je rezultiralo s dvije datoteke:

- CROSBI_Full.csv → (preuzeti podaci)
- AUTHORS_Full.csv → (dohvaćeni podaci)

Datoteka CROSBI_Full.csv sadrži sve zapise CROSBI baze opisane u 57 stupaca, dok datoteka AUTHORS_Full.csv sadrži ime i prezime znanstvenika, naslov rada i identifikacijsku oznaku rada i znanstvenika.



Slika 6. Dijagram transformacije podataka (Samostalna izrada, 2019)

Prilikom transformacije datoteka kreirano je ukupno 11 skripti napisanih u *Pythonu* i *Jupyter Notebook*-u od kojih je svaka zadužena za rješavanje određenog problema s dohvaćenim podacima.

5.6.1. Transformacija preuzetih podataka

Kompletna transformacija preuzetih podataka je odrađena u jednoj skripti u *Jupyter Notebook*-u koja je za lakše razumijevanje podijeljena na 6 zasebnih skripti. Prva skripta „FormatColumns.ipynb“ uzima datoteku CROSBI_Full.csv te iz nje briše nepotrebne stupce i formatira preostale, potrebne stupce. Za svrhu ovog rada odabrani su sljedeći stupci: *CC_INDEKS, SC_INDEKS, GODINA, KATEGORIJA, _NASLOV, _SKUP, _AUTORI, _PODRUCJE_ZNANOSTI, _VRSTA i _CASOPIS*, dok je ostalih 47 stupaca uklonjeno. Tijekom transformacije podataka primijećeno je kako se u naslovu rada na nekim mjestima nalaze posebni znakovi „\r“ (engl. *carriage return*) i „\n“ (engl. *line feed*) te njihove kombinacije, točnije „\r\n“ i „\r\r\n“. Nakon detaljnijeg istraživanja otkriveno je da je kombinacija „\r\r\n“ stavljena na nasumične pozicije unutar samog naslova umjesto razmaka (pretpostavka je da je kombinacija „\r\r\n“ dodana programski kako bi se naslov prelomio unutar HTML stranice). Ostale kombinacije i pojedinačni znakovi su stavljene na kraju naslova. Iz razloga što *Python* funkcija *to_csv* očitava upravo znakove „\r“ i „\n“ kao prelazak u novi red takvi znakovi unutar

naslova rada „lome“ jedan zapis u više redaka čime zapis postaje neupotrebljiv. Zbog toga je bilo potrebno sve „\r\r\n“ pretvoriti u razmak kako bi se sami naslov rada očuvao u izvornom obliku dok se znakovi „\r“, „\n“ i „\r\n“ u potpunosti brišu iz datoteke čime su se uspjeli sačuvati svi radovi. Stupce *CC_INDEKS* i *SC_INDEKS* je također bilo potrebno formatirati. Stupci su bili prazni ako rad nije CC i/ili SC te „1“ ako je rad klasificiran kao CC i/ili SC. Zbog jednostavnosti su svi prazni redovi u navedenim stupcima zamijenjeni s „0“ kako bi podaci bili konzistentni. Skripta nakon toga sprema dobivene rezultate u FormatedColumns.csv.

Nakon formatiranja podataka pokreće se skripta „FilterByCategory.ipynb“ koja filtrira radove prema stupcu *KATEGORIJA*. Svi preuzeti podaci iz CROSBI baze podataka su svrstani u tri kategorije rada: znanstveni, stručni i ostalo. U ovom radu fokus je na znanstvenim radovima, zbog čega se brišu svi radovi koji nisu kategorizirani kao „znanstveni“. Nakon filtriranja radova briše se stupac *KATEGORIJA* jer sadrži samo jednu vrijednost te je samim time nepotreban. Filtriranjem radova po kategoriji na znanstvene radove izbrisano je ukupno 182 256 radova te se dobiveni rezultati spremaju u datoteku CategoryFiltered.csv koja sadrži 324 227 znanstvenih radova.

Kako CROSBI nema unaprijed zadanu kategorizaciju za stupac *_VRSTA* radova kao što ima za stupac *KATEGORIJA* upisivanje vrste rada se odvija ručno te zbog toga navedeni stupac sadrži ukupno 344 različite vrijednosti. No većina vrijednosti je pogreška u pisanju (npr. -, a2, „(znanstveni“ …) ili rijetko korištene vrijednosti (npr. Book review, zivotopis, editorial…) te radovi bez navedene vrste. Zbog konzistentnosti podataka odabrane su najkorištenije i najvažnije vrste radova: članak, sažetak, prošireni, cijeli rad, monografija, izvorni znanstveni članak, znanstveni, pregledni rad i znanstveni rad. Radovi čija vrsta nije navedena su filtrirani te nisu uključeni u ovo istraživanje. Nakon filtriranja znanstvenih radova po vrsti datoteka sadrži 277 042 znanstvenih radova te se sprema u TypeFiltered.csv.

Nadalje, kako jedan rad može biti svrstan u više područja te se može pojaviti na više mjesta, prilikom filtriranja i preuzimanja datoteke iz CROSBI-a postoji velika mogućnost preuzimanja duplikatnih redova zbog čega se moraju izbrisati svi duplikati (jedan od duplikata se ostavlja kao original). Za ovaj posao je zadužena skripta „DropDuplicates.ipynb“ koja briše ukupno 51 478 duplikatnih redova te sprema rezultate u datoteku WithoutDuplicates.csv.

Nakon što su podaci formatirani i bez duplikata potrebno je pregledati anomalije u podacima te ih ukloniti za što je zadužena skripta „CleanErrorRows.ipynb“. Skripta uzima datoteku WithoutDuplicates.csv, prolazi kroz svaki red datoteke te ga briše ako godina nije u rasponu 1900 – 2020, ako CC ili SC indeksi nisu u „0“ ili „1“ te ukoliko stupci *_NASLOV, _PODRUCJE_ZNANOSTI, _AUTORI, _VRSTA* ili *GODINA* ne sadrže nikakve podatke. Prije prolaženja po redovima bilo je potrebno mapirati sve brojčane vrijednosti (stupce *CC_INDEKS,*

SC_INDEKS i GODINA) u format „0:.0f“ kako bi se uklonio decimalni dio .0 koji *Python* funkcija *to_csv* automatski dodaje svakom stupcu s numeričkim vrijednostima. Formatiranje numeričkih stupaca drastično je smanjilo vremensko trajanje prolaženja po redovima datoteke. Skripta „CleanErrorRows.ipynb“ je na opisani način pronašla i eliminirala ukupno 46, od čega u 2 reda godina nije bila u zadanom rasponu, 29 redova koji ne sadrže nikakve vrijednosti u odabranim stupcima te 15 redova koji su sadržavali riječi u stupcu *GODINA*. Nakon uklanjanja anomalija u podacima skripta sprema dobivene rezultate u datoteku CROSBI_Cleaned.csv.

Kako jedan rad može napisati više autora, prilikom preuzimanja radova funkcija koju CROSBI koristi za spremanje podataka u CSV datoteke u stupac *_AUTORI* stavi popis svih autora odvojenih znakom „ ; „ koji su sudjelovali u izradi rada. No kako je svakom autoru potrebno dodijeliti identifikacijsku oznaku, iz druge dohvaćene datoteke potrebno je svaki rad podijeliti po autorima te ih spremiti u posebni red. Kako bi se razriješio uočeni problem, kreirana je *Python* ulančana funkcija (u daljnjem tekstu: *funkcija odvajanja*) koja se sastoji od 8 zasebnih *Python* funkcija:

```
1. Files = files.set_index(files.columns.drop('_AUTORI',1).tolist())
   ._AUTORI.str.split(';',expand=True)
   .stack()
   .reset_index()
   .rename(columns={0:'_AUTORI'})
   .loc[:,files.columns]
```

Prvi korak u funkciji odvajanja je postavljanje vlastitog indeksa koji će obuhvaćati sve podatke u redu osim stupca s popisom autora. Na taj način se prilikom odvajanja autora u zasebne redove indeks tretira kao identifikacijska oznaka reda te ostaje netaknut. Kako bi se kreirao složeni indeks reda, koristi se funkcija *set_index* kojoj se prosljeđuje popis svih zaglavlja koji će predstavljati složeni indeks (u ovom slučaju sva zaglavlja osim _AUTORI, zbog čega se navedeni stupac izbacuje). Nakon što je kreiran složeni indeks, stupac _AUTORI se funkcijom *split* dijeli prema znaku „ ; „ nakon čega zbog dodanog parametra *expand* = *True* svaki podatak postaje zasebni stupac. Tablica tada sadrži složeni indeks te stupce pod nazivima 0, 1, 2, 3… od kojih svaki stupac sadrži ime i prezime jednog autora. Tada se nad dobivenom tablicom poziva funkcija *stack* koja rastavlja sve stupce u retke tablice. Na taj način se prilikom odvajanja zadržava složeni indeks (samim time i svi podaci osim imena) dok se imena autora odvajaju u retke. Funkcija vraća objekt *Series* koji sadrži složeni indeks i ime autora po jednom zapisu. Nakon toga je potrebno resetirati indekse kako bi se *Series* objekt pretvorio u *dataframe* objekt sa svakim podatkom u zasebnom stupcu za što se koristi funkcija *reset_index*. U ovom trenutku podatkovni okvir izgleda kao na slici Slika 7**.**

	CC_INDEKS	SC_INDEKS	GODINA	_NASLOV	_SKUP	_PODRUCJE_ZNANOSTI	_VRSTA	_CASOPIS	level_8	0
0	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Biologija; Kemija	clanak	Marine Pollution Bulletin	0	Bajt, Oliver
1	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Biologija; Kemija	clanak	Marine Pollution Bulletin	1	Ramšak, Andreja
2	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Biologija; Kemija	clanak	Marine Pollution Bulletin	2	Milun, Vesna
3	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Biologija; Kemija	clanak	Marine Pollution Bulletin	3	Andral, Bruno
4	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Biologija; Kemija	clanak	Marine Pollution Bulletin	4	Romanelli, Giulia

Slika 7. Međurezultat funkcije odvajanja (Snimka zaslona, 2019)

Dakle, prvotno se definirao složeni indeks koji je sadržavao sve podatke osim stupca *_AUTORI*, nakon čega se navedeni stupac podijelio zbog čega se stupac *_AUTORI* briše te se za svakog autora stvara novi stupac s generiranim nazivom 0, 1, 2, itd. Nakon razdvajanja autora je pozvana funkcija koja je svakog autora iz generiranih stupaca smjestila u novi red s istim složenim indeksom, što je rezultiralo spremanjem svakog autora u prvi generirani stupac s nazivom „0“. Na kraju je potrebno vratiti dobivene rezultate u prvotni oblik tablice, za što se iskoristila *Python* funkcija *loc* kojoj je kao prvi argument dan znak „ : „ što funkciji označava da obuhvati cijeli stupac, dok je za drugi argument dana lista naziva zaglavlja prvotnog oblika tablice. Dakle, funkcija vraća cijeli stupac ukoliko se nalazi u listi zaglavlja prvotnog oblika tablice istim redoslijedom. No tada nastaje problem s generiranim stupcem „0“, zbog čega je potrebno prije *loc* funkcije preimenovati navedeni stupac u prvotni naziv *_AUTORI* kako bi ga funkcija *loc* uključila u završnu tablicu.

Krajnji rezultat funkcije odvajanja se sprema u varijablu definiranu prije početka funkcije te je prikazan na Slika 8.

	CC_INDEKS	SC_INDEKS	GODINA	_NASLOV	_SKUP	_AUTORI	_PODRUCJE_ZNANOSTI	_VRSTA	_CASOPIS
0	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Bajt, Oliver	Biologija; Kemija	clanak	Marine Pollution Bulletin
1	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Ramšak, Andreja	Biologija; Kemija	clanak	Marine Pollution Bulletin
2	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Milun, Vesna	Biologija; Kemija	clanak	Marine Pollution Bulletin
3	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Andral, Bruno	Biologija; Kemija	clanak	Marine Pollution Bulletin
4	0	0	2019.0	Assessing chemical contamination in the coasta...	NaN	Romanelli, Giulia	Biologija; Kemija	clanak	Marine Pollution Bulletin

Slika 8. Krajnji rezultat funkcije odvajanja (Snimka zaslona, 2019)

Nakon toga je potrebno odvojiti ime i prezime u odvojeni stupac kako bi poštivali pravila 1NF[1]. Kako su ime i prezime zapisani u obliku „prezime, ime“, potrebno je podijeliti stupac po zarezu i spremiti u privremenu varijablu. Varijabla tada sadrži prvi stupac s prezimenom i drugi stupac s imenom, nakon čega se svaki stupac zasebno prepiše u stupce *PREZIME* i *IME*. Na kraju je potrebno izbrisati redundantni stupac *_AUTORI* s punim imenom i prezimenom. Nadalje, CROSBI funkcija za automatsko generiranje liste autora na kraju svake liste ponovno doda znak „ ; “ što prilikom odvajanja svakog autora u zasebni redak kreira novi redak koji ne sadrži ime i prezime (autor obuhvaćen nakon zadnjeg znaka „ ; „ koji ne postoji). Iz tog razloga je potrebno još jednom izbrisati sve redove koji u stupcima *Ime* i *Prezime* ne sadrže nikakve podatke. Kreiranjem novog zapisa za svakog autora veličina datoteke se drastično povećala. Skripta „SeparateAuthors.ipynb“ je dodala ukupno 563 862 redova te završna datoteka sadrži 789 380 redova od čega je 225 518 jedinstvenih znanstvenih radova. Skripta sprema završnu datoteku pod imenom CROSBI_Final.csv.

5.6.2. Transformacija dohvaćenih podataka

Datoteka AUTHORS_Full.csv je dohvaćena pomoću skripte te automatski ne zahtijeva veliku obradu podataka jer se dohvaćaju samo potrebni specifični podaci iz HTML dokumenta. No ipak je uočeno kako postoji anomalija u identifikacijskim brojevima autora gdje postoji mogućnost da je umjesto identifikacijskog broja zapisana riječ ili pak nema ništa zapisano. Iz tog razloga je potrebno proći kroz svaki zapis datoteke te izbrisati zapis ukoliko identifikacijski broj ne postoji ili sadrži slova. Skripta „CleanErrorAuthorRows.ipynb“ je na taj način pronašla te izbrisala ukupno 104 zapisa s opisanim anomalijama te spremila dobivene rezultate u datoteku autoriCleaned.csv.

Kako dohvaćeni podaci ne sadrže listu autora nego samo jednog autora, nema potrebe za odvajanjem autora u zasebne redove no potrebno je odvojiti ime i prezime u zasebne stupce kako bi slijedili pravila 1NF. U ovom slučaju imena su u stupcu *name* zapisana u obliku „ime prezime“ te je potrebno odvojiti stupac po razmaku i spremiti rezultat u privremenu varijablu. Varijabla će tada sadržavati prvi stupac s imenom i drugi stupac s prezimenom koji se prepisuju u stupce *IME* i *PREZIME* respektivno. Nakon toga je potrebno ukloniti redundantni stupac *name* te spremiti dobivene podatke u datoteku autoriSeparated.csv koja sadrži ukupno 999 427 redaka.

[1] Prva normalna forma (1NF) – normalna forma koja osigurava da relacija sadrži samo proste (atomne) vrijednosti koje se ne ponavljaju.

5.6.3. Spajanje preuzetih i dohvaćenih podataka

Nakon što su svi željeni podaci bili preuzeti te dohvaćeni, bilo ih je potrebno spojiti kako bi se stvorila jedna datoteka s kojom će se kasnije raditi. Uostalom, početni cilj u radu je i bio skupiti sve relevantne podatke, no moralo se odraditi dodatno dohvaćanje metodom *struganja podataka* zbog čega su kao posljedica nastale dvije odvojene datoteke – jedna stvorena preuzimanjem, a jedna dohvaćanjem podataka. Spajanje datoteka odrađeno je također programskim jezikom *Python* uz pomoć već nekoliko puta spominjanog paketa *pandas*.

```
1. import pandas as pd
2. autori =
   pd.read_csv('../csv/test/full/AUTHORS_Cleaned_Separated.csv',
   dtype=str)
3. radovi =
   pd.read_csv('../csv/test/full/CROSBI_Cleaned_Separated_Cleaned.csv',
   dtype=str)
```

Skripta otpočinje učitavanjem CSV datoteka nastalih preuzimanjem te dohvaćanjem podataka što je prethodno opisano, također je uključen i paket *pandas* s pseudonimom *pd* jer upravo je taj paket odradio većinski posao prilikom spajanja. Nakon učitavanja potrebitih datoteka, bilo je potrebno još jednom provjeriti te osigurati da datoteke ne sadržavaju duplikate kako bi smanjili vrijeme izvođenja samog programskog kôda te onemogućili nastajanje eventualnih problema u nastavku. Spomenuto je učinjeno pozivanjem metode *drop_duplicates* nad *pandas* podatkovnim okvirima u koje su prethodno učitane CSV datoteke.

```
1. autori = autori.drop_duplicates()
2. radovi = radovi.drop_duplicates()
```

Svim naslovima potom su sva slova promijenjena u mala slova – ovo je učinjeno u obje datoteke. Promjena slova u mala slova učinjeno je jer će se spajanje dvaju datoteka vršiti upravo po atributu naslov (engl. *title),* a sam paket *pandas* razlikuje mala i velika slova što može prouzročiti određene probleme ukoliko je pokoji naslov napisan drugačije unutar datoteka. Primjerice ukoliko je naslov „Diskretna matematika s teorijom grafova” zapisan u jednoj CSV datoteci kao „Diskretna matematika s teorijom grafova” dok je u drugoj zapisan kao „diskretna matematika s teorijom grafova”m ti se zapisi naprosto ne bi spojili, što bi u konačnici rezultiralo gubitkom naslova, konkretno i ostalih podataka vezanih uz te naslova što je naravno nedopustivo. Promjena slova u mala slova za atribut naslov učinjeno je na način da se nad odabranim atributom dohvaća te poziva metoda *lower*.

```
1. autori["title"] = autori["title"].str.lower()
2. radovi["title"] = radovi["title"].str.lower()
```

Nakon što su prethodne pripreme bile izvršene, sve je bilo spremno za izvršavanje spajanja samih datoteka pomoći metode *merge* paketa *pandas.* Kao argumente metode proslijedili smo same podatkovne okvire „autori" i „radovi", listu atributa po kojima želimo izvršiti spajanje (naslov) te način samog spajanja (desno). Odabrana lista atributa predstavlja ključ prema kojem će se spajanje izvršiti dok je odabrani način spajanja – konkretno desno spajanje suštinski identično SQL desnom vanjskom spajanju (engl. *SQL Right Outer Join).*

```
1. merged = pd.merge(autori, radovi, on=['title'], how='right')
```

Odrađeno spajanje rezultiralo je velikim brojem redundantnih redova – nastali su brojni duplikati, *null* vrijednosti te čak i dodatni dupli cirani stupci koji su preuzeti iz lijevog podatkovnog okvira s obzirom da je rađeno desno spajanje, konkretno preuzeti su stupci *name* i *surname* odnosno ime i prezime autora koji su tada zapravo postali duplikati atributa. Razlog tome proizlazi iz činjenice da se ti stupci odnosno atributi nalaze u oba podatkovna okvira, a odabrana metoda spajanja zahtijeva da se u spoj uključuju svi atributi respektivno s odabranim ključem. Kako bi se dobivene anomalije razriješile, bilo je potrebno napraviti samo par poziva *pandas* metoda. Prvotno su izbačeni svi oni redovi koji za vrijednost atributa *authorId* imaju *null* vrijednost jer takvi redovi predstavljaju podatke koji nam nisu od interesa. Potom su izbrisani spomenuti redundantni atributi duplikati te su konačno uklonjeni svi redovi duplikati.

```
1. merged = merged[merged["authorId"].notnull()]
2. merged = merged.drop("name_y", 1)
3. merged = merged.drop("surname_y", 1)
4. merged = merged.drop_duplicates()
```

Naposljetku su preimenovani pojedini atributi čije se ime promijenilo prilikom spajanja. Naime, pri nastanku atributa duplikata podatkovni okvir *pandas* morao je razriješiti imena duplikata atributa te je iste preimenovao. Konkretno, naziv atributa *name* iz podatkovnog okvira autori postao je *name_x,* dok je naziv atributa *name* iz podatkovnog okvira radovi postao *name_y*, analogno i atribut *surname*. Kako bi se to ispravilo, izvršeno je preimenovanje.

```
1. merged = merged.rename(index=str, columns={"name_x": "name",
   "surname_x": "surname"})
2. merged = merged.drop(columns="title")
```

U konačnici izbačen je atribut *title* odnosno naslov jer on nije relevantan za samo grafovsko skladište podataka. Taj atribut bio je potreban samo prilikom spajanja datoteka.

Dobiveni podatkovni okvir nakon svih njegovih preinaka zapisan je u CSV datoteku naziva mergedDataNoTitles.csv. Ovime je spajanje bilo okončano.

```
1. merged.to_csv("../csv/main/exports/mergedDataNoTitles.csv",
   encoding="utf-8", index=False)
```

Prije samog spajanja CSV datoteka AUTHORS_Cleaned.csv sadržavala je 999427 redova, dok je CSV datoteka CROSBI_Final.csv sadržavala 789 380 redova. Nakon spajanja tih dvaju datoteka nastala je kako je spomenuto CSV datoteka mergedDataNoTitles.csv koja je sadržavala 563 922 reda. Na temelju ovih podataka može se primijetiti smanjenje broja redova u odnosu na obje prvotne datoteke. Razlog tomu leži u činjenici da je CSV datoteka AUTHORS_Cleaned.csv sadržavala apsolutno sve naslove koji su dostupni na mrežnoj bibliografskoj bazi podataka "Hrvatska znanstvena bibliografija", a čiji autori imaju matični broj znanstvenika - većina tih naslova, gotovo pola, nije uopće spadala u željeni skup podataka, dok je CSV datoteka CROSBI_Final.csv pak sadržavala popriličan broj naslova čiji autori uopće nemaju matični broj znanstvenika – takvi naslovi također ne spadaju u željeni skup podataka.

S obzirom na opisane činjenice jasno je da su prilikom spajanja brojni redovi odnosno zapisi bili u potpunosti redundantni te su se time onda i izgubili – od tuda proizlazi smanjenje broja redova u spojenoj CSV datoteci. Prilikom opisivanja dohvaćanja podataka provjeravana je domena iz koje proizlaze brojevi bibliografskih jedinica na mrežnoj bibliografskoj bazi podataka "Hrvatska znanstvena bibliografija", no nisu provjerene vrijednosti iznad 2 000 000 – odnosno nije provjereno postoje li naslovi s brojem bibliografske jedinice iznad 2 000 000. S obzirom da je spajanje prošlo kako je predviđano, odnosno izgubljeno je onoliko redova koliko je relativno i predviđano, zaključili bismo da je dohvaćanje podataka prošlo adekvatno, te zaista ne postoje naslovi s brojem bibliografske jedinice iznad 2 000 000. Kako bi taj zaključak bio poduprt još pokojom činjenicom, odlučili smo napisati *Python* skriptu koja će dati broj naslova na mrežnoj bibliografskoj bazi podataka "Hrvatska znanstvena bibliografija" čiji autori imaju matični broj znanstvenika te bi kao takvi spadali u željeni skup podataka, pritom imajući na umu da u željeni skup nisu ušli baš svi naslovi čiji autori imaju matični broj znanstvenika, već su i ti naslovi filtrirani kako je opisano.

Kako bi se došlo do broja naslova čiji autori imaju matični broj znanstvenika, korišteni su isti moduli te paketi koji su korišteni i prilikom dohvaćanja podataka – konkretno: *pandas, requests* i *BeautifulSoup.* S obzirom da je pregled prema znanstveniku na mrežnoj bibliografskoj bazi podataka "Hrvatska znanstvena bibliografija", strukturiran abecedno te same poveznice nose sljedeću strukturu:

https://www.bib.irb.hr/pregled/znanstvenici/slovo/{SLOVO}, iterativnim postupkom su se generirale poveznice za svako pojedino slovo hrvatske abecede. Svaki pregled prema pojedinom slovu prikazao bi sve autore s matičnim brojem te bi pokraj njihovog imena u zagradama stajao broj naslova odnosno radova čiji su oni autori ili koautori. Zbrajanje tih brojeva rezultirati će ukupnim brojem naslova čiji autori, kako je napomenuto, imaju matični broj znanstvenika. S obzirom da je dobar dio autora radio na istim radovima, konačni zbroj neće biti u potpunosti ispravan jer će biti pribrojeni naslovi duplikati – no svejedno će rezultat dati vrijednu informaciju. Prvotno je definirana lista slova hrvatske abecede.

```
1. letters = ['A', 'B', 'C', 'Č', 'Ć', 'D', 'Đ', 'E', 'F'
2. 'G', 'H', 'I', 'J', 'K', 'L', 'LJ', 'M', 'N',
3. 'NJ', 'O', 'P', 'Q', 'R', 'S', 'Š', 'T', 'U',
4. 'V', 'W', 'X', 'Y', 'Z', 'Ž' ]
```

Potom su iterativno generirane poveznice s kojih su se dohvaćali brojevi naslova pojedinih prikazanih autore te su se isti zbrajali.

```
1. totalTitles = 0
2. for letter in letters:
3. page =
   requests.get("https://www.bib.irb.hr/pregled/znanstvenici/slovo/{}".f
   ormat(letter))
4. soup = BeautifulSoup(page.content, 'html.parser')
5. titles = soup.findAll("small", title="Broj pridruženih radova",
   text=True)
6. for author in titles:
7.          totalTitles +=
   int(author.text.replace('(',"").replace(")",""))
8. print(totalTitles)
```

Skripta je rezultirala brojem 660 416, što bi značilo da toliko ukupno ima naslova čiji autori imaju matični broj, respektivno uzimajući u obzir duplikate – dakle naslova je još i manje. S obzirom da naša spojena CSV datoteka sadrži 563 922 redova, možemo zaključiti da je željeni skup podataka validan. Naravno, svjesni smo činjenice da postoji mogućnost da smo izgubili pokoji naslov ili autora, no razloga za to može biti mnogo te se gotovo i ne mogu identificirati.

Nakon spajanja datoteka bilo je još samo potrebno izvršiti nekoliko promjena nad samim skupom podataka. Konkretno atributi *_SKUP, _CASOPIS* te *_PODRUCJE_ZNANOSTI* imali su nekoliko vrijednosti koje su zapravo sadržavale niz vrijednosti jer su naprosto tako zapisane u inicijalnim CSV datotekama koje su preuzete s mrežne bibliografske baze podataka „Hrvatska znanstvena bibliografija”. Primjerice, atribut *_PODRUCJE_ZNANOSTI* imao je vrijednosti „Temeljne medicinske znanosti; Kliničke medicinske znanosti”, što su dvije odvojene vrijednosti. Takve slučajeve valjalo je razdvojiti imajući na umu znak „;” koji je bio

konzistentan u svim takvim vrijednostima atributa. Samo razdvajanje izvršeno je pomoću već opisane funkcije razdvajanja. Tako su vrijednosti atributa *_SKUP* razdvojene na sljedeći način:

```
splitData = (data.set_index(data.columns.drop(['_SKUP'],1).tolist())
._SKUP.str.split(';', expand=True)
.stack()
.reset_index()
.rename(columns={0:'_SKUP'})
.loc[:, data.columns])
```

Vrijednosti atributa *_CASOPIS* te *_PODRUCJE_ZNANOSTI* razdvojeni su analogno prethodnom primjeru. Nakon razdvajanja zamijećeno je da neke vrijednosti atributa sadrže nepotrebne znakove poput razmaka i sl.. Kako bi se takve anomalije razriješile iterativno smo nad vrijednostima podatkovnog okvira *splitData* pozivali metodu *.strip()* iz paketa *pandas.*

```
for column in splitData:
 splitData[column] = splitData[column].str.strip()
```

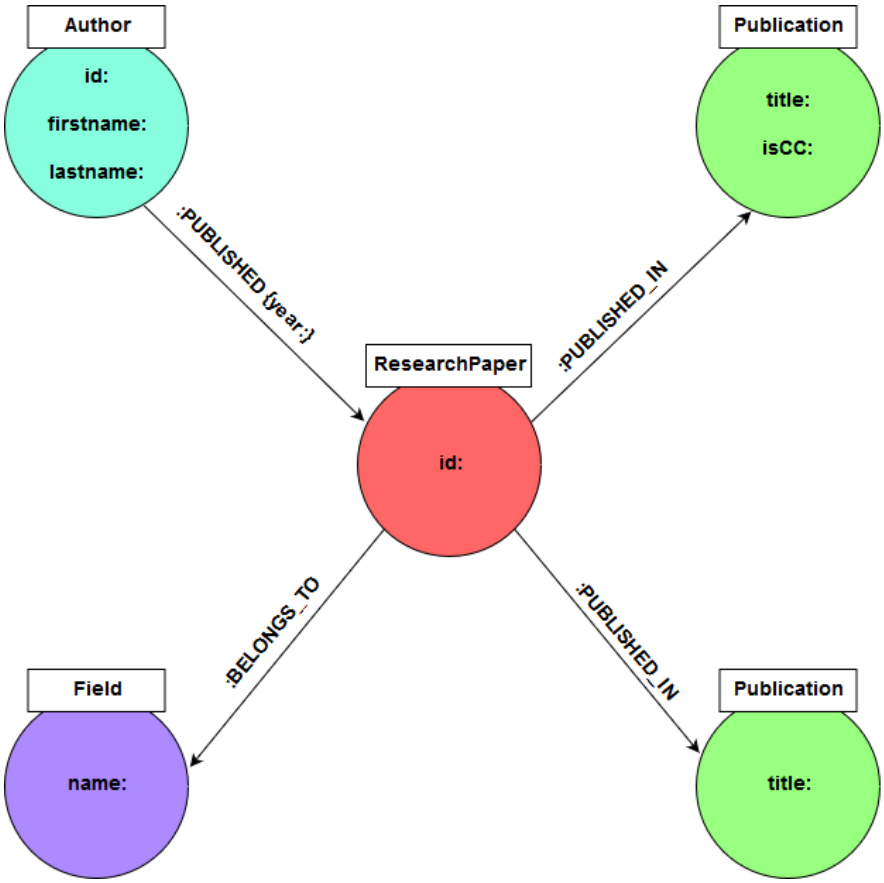
Time smo eventualno zapisano ime „ Luka” promijenili u „Luka” izbacivši nepotreban razmak. Također vrijedi i za vrijednosti koje su imale takve znakove s desne strane – primjerice „Denis „ u „Denis”. Konačno su promijenjena imena atributa kako bi se u nastavku rada imala određena konzistencija nad skupom podataka. Nakon promjene imena atributa sadržaj podatkovnog okvira je zapisan u CSV datoteku naziva dataFinal.csv.

```
splitData = splitData.rename(index=str, columns={"publicationId":
"bibliografskaJedinica",
"authorId": "maticniBrojAutora",
"name": "ime",
"surname": "prezime",
"CC_INDEKS": "ccIndeks",
"SC_INDEKS": "scIndeks",
"GODINA": "godina",
"_SKUP": "skup",
"_PODRUCJE_ZNANOSTI": "podrucjeZnanosti",
 "_VRSTA": "vrsta",
 "_CASOPIS": "casopis"})

 splitData.to_csv("../csv/main/exports/dataFinal.csv",
encoding="utf-8", index=False)
```

5.6.4. Agregiranje podataka u grafovskoj bazi podataka

U ovom će poglavlju biti predstavljene *Cypher* naredbe koje su korištene za uvoz podataka u grafovsku bazu podataka. Bazu smo koristili za agregaciju podataka radi lakšeg izvršavanja izračuna nad njima, čiji su se rezultati kasnije koristili za izradu grafovskog skladišta podataka, no prije toga potrebno je objasniti model podataka koji se koristio u bazi podataka.



Slika 9. Model podataka grafovske baze podataka (izrada autora)

Na Slika 9 možemo zamijetiti 4 čvora različitih oznaka i 3 različite veze. Čvor oznake „ResearchPaper“ u bazi podataka predstavlja bibliografsku jedinicu, tj. istraživački rad te se bilježi podatak o njegovom jedinstvenom identifikacijskom broju u *id* svojstvu čvora. Čvor oznake „Author“ predstavlja autora bibliografske jedinice te se bilježe podaci o njegovom matičnom broju, imenu i prezimenu u svojstvima *id, firstname, lastname* respektivno. Usmjerenost veze *PUBLISHED* predstavlja da je autor objavio neki istraživački rad. Na toj se vezi također bilježi i podatak o godini u kojoj je autor objavio određeni rad u svojstvu *year*. Čvor oznake „Publication“ predstavlja publikaciju.

Publikacije tretiramo drugačije ovisno o tome je li u pitanju znanstveni časopis ili pak neki konferencijski rad. Kod časopisa se bilježi podatak o njegovu nazivu i CC indeksu za godinu u kojoj je neki rad objavljen u tom časopisu u svojstvima *title* te *isCC*, dok se za konferencijske radove bilježi samo podatak o njihovu nazivu u svojstvu *title*. Ovo je klasičan primjer fleksibilne sheme NoSQL baza podataka, pa tako i *Neo4j*-a, u kojoj nije nužno da svaki zapis o nekom entitetu postoji u bazi. U ovom je slučaju to svojstvo vidljivo kod publikacija koje ne moraju sadržavati ista svojstva kao i preostali zapisi. Usmjerenost veze *PUBLISHED_IN* označava u kojoj je publikaciji neki istraživački rad objavljen. Čvor oznake „Field*“* predstavlja znanstveno polje, a orijentacija veze *BELONGS_TO* predstavlja kojem polju pripada neki istraživački rad. U nastavku se nalaze *Cypher* naredbe korištene za uvoz podataka u bazu.

```
1. USING PERIODIC COMMIT 5000
2. LOAD CSV WITH HEADERS FROM 'file:///dataFinal.csv' AS line
3. CREATE
4. (a:Author {id: line.maticniBrojAutora, firstname: line.ime, lastname:
   line.prezime}),
5. (rp:ResearchPaper {id: line.bibliografskaJedinica}),
6. (f:Field {name: line.poljeZnanosti})
```

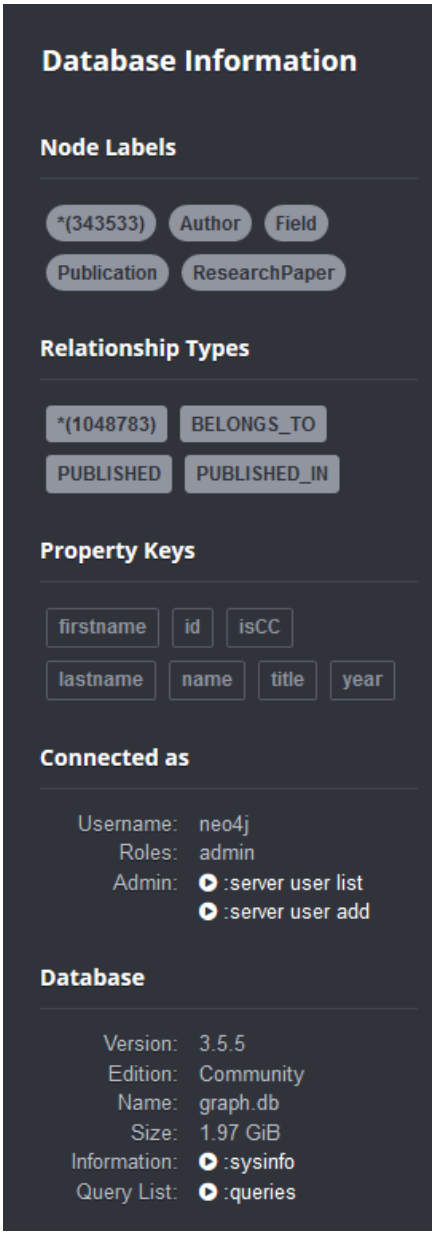
```
1. USING PERIODIC COMMIT 5000
2. LOAD CSV WITH HEADERS FROM 'file:///dataFinal.csv' AS line
3. CREATE (p:Publication {title: line.skup})
```

```
1. USING PERIODIC COMMIT 5000
2. LOAD CSV WITH HEADERS FROM 'file:///dataFinal.csv' AS line
3. CREATE  (p:Publication {title: line.casopis, isCC: line.ccIndeks})
```

Zajednički dio svih triju naredbi odnosi se na učitavanje zapisa iz datoteke *dataFinal.csv*. Dio naredbe „USING PERIODIC COMMIT 5000“ obično se koristi kod uvoza velikih količina podataka, a nalaže *Neo4j*-u da svakih, u ovom slučaju, 5 000 uspješno obrađenih zapisa iz datoteke spremi promjene u bazu. Također sprečava preopterećenje radne memorije podacima. Naglasak je u ovom slučaju na uspješnosti obrađenih zapisa jer, ako se dogodi greška prilikom učitavanja podataka iz CSV datoteke, a pritom je broj uspješno obrađenih zapisa manji od 5 000 – učitani podaci neće se spremiti u bazu, tj. transakcija neće biti zabilježena. To je primjer karakteristike atomnosti *Neo4j*-a kao *ACID* baze podataka. Sljedeći zajednički dio svih naredbi je „LOAD CSV WITH HEADERS FROM 'file:///dataFinal.csv' AS line“. „LAOD CSV WITH HEADERS“ govori *Neo4j*-u da će se učitati CSV datoteka čiji je prvi redak popis stupaca. To nam kasnije u naredbi omogućava pristupanje određenom atributu učitanog zapisa iz datoteke prema njegovu imenu, tj. prema imenu stupca. „ *'file:///dataFinal.csv'* “dio specificira ime i putanju do CSV datoteke koja će se učitati. „*AS line“* označava alias za učitani zapis iz datoteke što nam omogućava lakše pristupanje atributima

zapisa. Nakon klauzule „CREATE“ slijede definicije oznaka čvorova koji će se stvoriti i njihovih svojstava. Svojstvima čvorova pridružene su vrijednosti atributa zapisa iz CSV datoteke. Tako na primjer „f:Field {name: line.poljeZnanosti}“ označava da će se stvoriti novi čvor oznake „Field“ sa svojstvom *name* kojem će biti pridružena vrijednost atributa *poljeZnanosti* učitanog zapisa.

Slika 10 prikazuje stanje u bazi podataka nakon uvoza podataka i veza. Možemo zamijetiti da se u bazi trenutno nalazi 345 533 čvora i 1 048 783 veza. Sama baza zauzima 1.97 GB podataka. Uvoz podataka iz *dataFinal.csv* datoteke trajao je 72.8 sekundi.



Slika 10. Stanje baze podataka nakon uvoza veza i podataka (Snimka zaslona, 2019)

Nakon učitavanja sirovih pročišćenih podataka uslijedila je njihova agregacija unutar baze podataka kako bi se s obzirom na veliku količinu podataka taj proces ubrzao.

Konkrektno, za izračunavanje ukupnog broja radova i broja CC radova u časopisima korišten je sljedeći *Cypher* upit:

```
1. MATCH (a:Author)-[r:PUBLISHED]->(rp:ResearchPaper)-[r2:PUBLISHED_IN]-
   >(p:Publication), (rp)-[r3:BELONGS_TO]->(f:Field)
2. WHERE EXISTS(p.isCC)
3. WITH a.id AS researcher_id, a.firstname AS author_firstname,
4. a.lastname AS author_lastname, f.name AS area_field, r.year AS year,
   p.title AS publication_title, COLLECT(p) AS publications
5. RETURN researcher_id, author_firstname, author_lastname, year,
   area_field, publication_title, "J" AS research_type,
   SIZE(publications) AS numberOfPapers, SIZE(FILTER(x IN publications
   WHERE x.isCC  ="1")) AS numberOfCCPapers .
```

Ovim se upitom dohvaćao broj radova (*numberOfPapers*) te broj *CC* radova (*numberOfCCPapers*) koje je neki autor (*researcher_id, author_firstname, author_lastname*) objavio u nekoj godini (*year*), u nekom časopisu (*publication_title*). Rezultat ovog upita spremio se u datoteku exportCasopisi.csv.

Drugi upit kojim se izračunava broj radova objavljenih u zbornicima konferencija ima približno jednaku sintaksu, uz napomenu da ovdje nismo trebali izračunavati broj CC radova budući da se radi o konferencijskim radovima:

```
1. MATCH (a:Author)-[r:PUBLISHED]->(rp:ResearchPaper)-[r2:PUBLISHED_IN]-
   >(p:Publication), (rp)-[r3:BELONGS_TO]->(f:Field) WHERE
   NOT(EXISTS(p.isCC)) return a.id AS researcher_id, a.firstname AS
   author_firstname, a.lastname AS author_lastname, f.name AS
   area_field, r.year AS year, p.title AS publication_title, "J",
   count(r) AS numberOfPapers .
```

Njime se dohvaćao broj konferencijskih radova koje je neki autor objavio u nekoj godini. U objašnjenju modela podataka grafovske baze razjašnjena je razlika u svojstvima čvorova oznake „Publication“ koja omogućuje diskriminaciju čvorova koji predstavljaju časopise od čvorova koji predstavljaju konferencijske radove. U ovim se upitima ta razlika provjerava *Cypher* funkcijom *EXISTS(),* koja ispituje sadrži li čvor koji je predmet obrade neko svojstvo koje se navodi kao argument funkciji – u ovom slučaju je to svojstvo *isCC*. Rezultati ovog upita izvezeni su u datoteku *exportKonferencije.csv*. Za izvoz rezultata upita u CSV format korištena je opcija „Export to CSV“, dostupna u grafičkom sučelju aplikacije *Neo4j Desktop*.

5.7. Učitavanje podataka

U ovom će poglavlju biti opisani upiti korišteni za učitavanje podataka pripremljenih u prethodnim koracima u grafovsko skladište podataka te konačno stanje u skladištu podataka nakon uvoza podataka. U nastavku slijede naredbe kojima su se podaci izvezeni iz baze podataka u prethodnom koraku učitavali u skladište podataka. One se ne razlikuju osobito od onih naredbi kojima su se podaci uvozili u bazu podataka, stoga će u nastavku biti objašnjene samo bitne razlike. Također, model podataka grafovskog skladišta podataka koji se koristio u pripremi opisan je ranije u poglavlju 5.4. Sintaksa prvog upita je sljedeća:

```
1. USING PERIODIC COMMIT 5000
2. LOAD CSV WITH HEADERS FROM 'file:///exportCasopisi.csv' AS line
3. CREATE
4. (a:Author {researcher_ID: line.researcher_id, firstname:
   line.author_firstname, lastname: line.author_lastname}),
5. (p:Publication {title: line.publication_title, type:"J"}),
6. (d:Date {year: line.year}),
7. (f:Area {name: "", field: line.area_field}) .
```

Ovom su se naredbom učitali podaci iz datoteke exportCasopisi.csv te su stvoreni čvorovi oznaka „Author“, „Publication“, „Date“ i „Area“. Zanimljivo je svojstvo *name* čvorova oznake „Area“. Prilikom uvoza podataka iz datoteka, svakom je čvoru kao vrijednost tog svojstva inicijalno pridružen prazan niz znakova. Vrijednosti ovih svojstava popunjena su kasnije nakon što smo eliminirali duplikatne čvorove. Sljedećom su se naredbom učitali podaci iz datoteke exportKonferencije.csv:

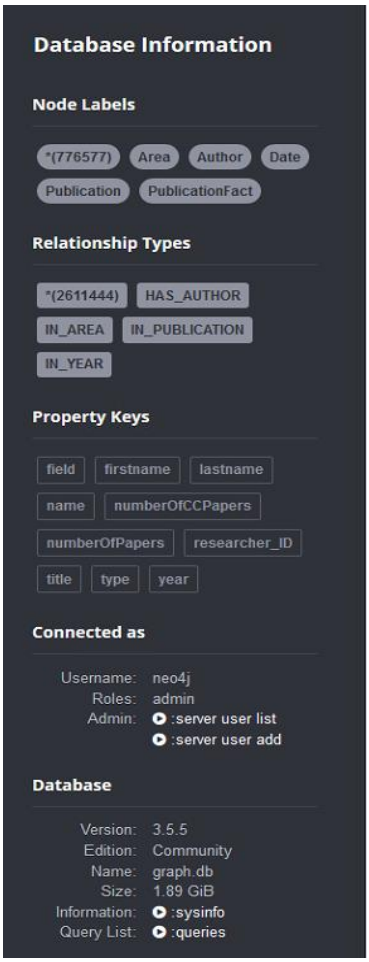
```
1. USING PERIODIC COMMIT 5000
2. LOAD CSV WITH HEADERS FROM 'file:///exportKonferencije.csv' AS line
3. CREATE
4. (a:Author {firstname: line.author_firstname, lastname:
   line.author_lastname, researcher_ID: line.researcher_id}),
5. (p:Publication {title: line.publication_title, type: "C"}),
6. (d:Date {year: line.year}),
7. (f:Area {name: "", field: line.area_field}) .
```

Bitno je napomenuti kako se čvorovi koji predstavljaju časopise razlikuju od onih koji predstavljaju konferencijske radove u skladištu podataka temeljem vrijednosti svojstva *type*, kojem je pridružena vrijednost „J“ ako je u pitanju časopis (engl. *Journal*), odnosno „C“ ako je u pitanju konferencijski rad (engl. *Conference*).

Posljednjom su se naredbom korištenom za učitavanje podataka zapravo dodijelile vrijednosti svojstvima *name* čvorova oznake „Area“, a sama naredba glasi:

```
1. LOAD CSV WITH HEADERS FROM 'file:///exportPodrucjaPolja.csv' AS line
2. FIELDTERMINATOR ';' MATCH(ar:Area)
3. WHERE ar.field = line.area_field SET ar.name = line.area_name .
```

Budući da se datoteka exportPodrucjaPolja.csv sastoji tek od nekoliko desetaka zapisa, pri uvozu podataka ovom naredbom nije bilo potrebno koristiti klauzulu „USING PERIODIC COMMIT“. Ovdje je važno istaknuti kako datoteka exportPodrucjaPolja.csv sadrži podatke o znanstvenim poljima i područjima kojima oni pripadaju te je dobivena na temelju ručno prikupljenih podataka s CROBI-ja. Novina koja se javlja u ovoj naredbi je klauzula „FIELDTERMINATOR“, kojom se *Neo4j*-u daje do znanja koji se znak koristi kao graničnik umjesto zadanog znaka „ , “, tj. zareza u CSV datoteci iz koje se uvoze podaci.



Slika 11. Stanje skladišta podataka nakon uvoza veza i podataka (Snimka zaslona, 2019)

Nakon svih uvezenih podataka, otklonjenih duplikata te uvezenih veza između čvorova, skladište podataka sadrži 776 577 čvorova; od toga su 22 213 čvorova autora, 78 znanstvenih područja, 101 357 publikacija (časopisi i zbornici konferencija) i 652 861 činjeničnih čvorova koji povezuju autore, područja, godinu i publikaciju. Skladište podataka sadrži ukupno 2 611 444 veza, ukupno zauzima 1.89 GB prostora na fizičkom mediju, a uvoz podataka u skladište trajao je 3,37 minute. Konačno stanje skladišta može se iščitati sa Slika 11.

5.8. Opis korištenih tehnologija za vizualizaciju rezultata

Python [41] je jednostavan i intuitivan programski jezik otvorenog koda (engl. *open-source*) napisan u programskom jeziku C. Omogućuje vrlo jednostavno i brzo učitavanje, obradu i spremanje tekstualnih datoteka zbog čega je često korišten za transformaciju podataka. Brzinu i jednostavnost pokazuje i prilikom obrade matematički zahtjevnih zadataka zbog čega je prepoznat i korišten od strane matematičara.

Anaconda [37] je platforma za podatkovne znanosti (engl. *data science platform*) koja olakšava rad s manipulacijom podataka. Dolazi zajedno s korisničkim sučeljem *Anaconda Navigator* putem kojeg je moguće jednostavno koristiti funkcionalnosti koje platforma pruža poput instaliranja *Python* biblioteka, pristupa zajednici korisnika i dokumentaciji pojedinih aplikacija te pokretanje istih aplikacija koji dolaze uz platformu.

Jupyter Notebook [38] je aplikacija bazirana na webu koja dolazi uz *Anaconda* platformu. Pruža interaktivno sučelje za kreiranje i izmjenu dokumenata baziranih na *Python* programskom jeziku unutar kojih je moguće pokretanje *Python* skripti te pisanje i uređivanje teksta.

Flask [39] je razvojni mikro okvir za web razvoj koji ne zahtjeva zasebne alate niti biblioteke kako bi ga se mogao koristiti. Temeljen je na biblioteci *Werkzeug* [40], koja je ključna za komunikaciju između *Python* koda i poslužitelja te na web stroju za predloške *Jinja2* [41] koji stvara komunikaciju između *Python* koda i HTML-a. *Flask* je u ovom projektu korišten za izradu klijentskog i poslužiteljskog dijela aplikacije.

Py2neo [42] je *Python* biblioteka za rad sa *Neo4j* bazom podataka unutar *Python* aplikacije. Osim navedene biblioteke, *Neo4j* nudi još nekoliko biblioteka za upravljanje svojim DBMS-om poput *Neo4j* i *Neomodel* biblioteke [43], no za svrhu ovog rada py2neo biblioteka pokazala se najboljom opcijom radi kvalitetne dokumentiranosti i jednostavne sinkronizacije sa *Flask*-om kao temeljnim razvojnim okvirom.

Plotly [44] je univerzalna biblioteka za prikazivanje grafova (moguće korištenje u programskom jeziku R, *JavaScriptu* te kombinaciju sa *Python* bibliotekama *pandas* i *matplotlib*). Osim što sadrži veliku količinu različitih grafičkih prikaza, *Plotly* je idealna biblioteka za prikazivanje rezultata upita na web aplikacijama radi svoje interaktivnosti. Biblioteka je jednostavna za korištenje i potkrijepljena iscrpnom dokumentacijom sa mnogo primjera korištenja raznih vrsta grafova. U ovom radu *Plotly* je korišten za prikaz dva stupčasta i jednog mjehuričastog grafičkog prikaza detaljnije opisanih u poglavlju 7.3.

Flask-bootstrap [45] je *Python* biblioteka korištena za uključivanje Bootstrap predloška. Korišten je Bootstrap predložak otvorenog koda pod nazivom „*SB Admin 2*“ [46] iz kojeg su uklonjeni suvišni dijelovi HTML, CSS i *JavaScript* koda. Prikaz sučelja realiziranog ovim predloškom nalazi se u poglavlju 7.3.

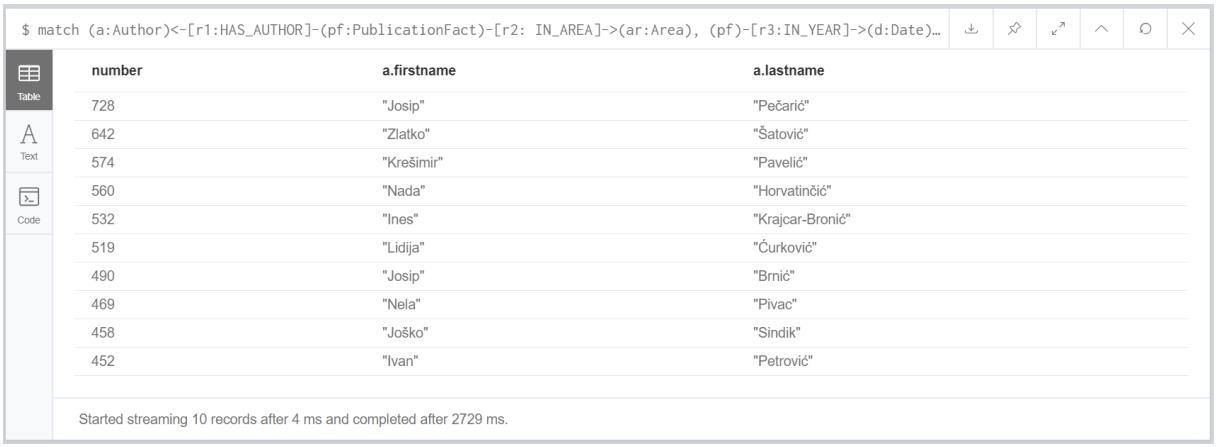
6. Rezultati

Grafovsko skladište podataka izgrađeno u prethodno opisanim koracima sadrži podatke koji se mogu koristiti za provođenje raznih analiza. U nastavku će biti objašnjeni upiti izvršeni na grafovskom skladištu podataka te kasnije prikazani na grafičkom sučelju kako bismo to i dokazali. Odabrana su tri upita koja daju uvid u neke trendove u znanstvenom publiciranju hrvatskih istraživača.

Prvi upit koji je prikazan na grafičkom sučelju je upit koji pokazuje top deset autora s najviše publikacija u određenom području i godini prema broju napisanih radova. Sintaksa upita je sljedeća:

```
1. MATCH (a:Author)<-[r1:HAS_AUTHOR]-(pf:PublicationFact)-[r2: IN_AREA]-
   >(ar:Area), (pf)-[r3:IN_YEAR]->(d:Date) return COUNT(r1) as number,
   a.firstname, a.lastname  ORDER BY number desc limit 10
```

Cilj upita je pronaći uzorak u skladištu koji pokazuje istovremenu međusobnu povezanost sljedećih tipova čvorova: „Author“, „PublicationFact“, „Area“ i „Date“. Povratna vrijednost ovog upita je broj veza između čvora „Author“ i „PublicationFact“ koja nam zapravo daje broj jedinstvenih objava nekog istraživača. Jednostavnim proširenjima može se doći do najboljih autora unutar određenog znanstvenog područja u određenoj godini. Potrebno je nakon ključne riječi „WHERE“ specificirati ime znanstvenog područja i godinu.



Slika 10. Top deset autora s najviše publikacija (Snimka zaslona, 2019)

Međutim, ovaj upit ne vraća deset autora s najviše objavljenih publikacija u određenom području već općenito. Za određeno znanstveno područje i godinu, primjerice za „Društvene znanosti“ i 2019. godinu i broj od 20 autora upit je sljedeći:

```
1. MATCH (a:Author)<-[r1:HAS_AUTHOR]-(pf:PublicationFact)-[r2: IN_AREA]->(ar:Area), (pf)-[r3:IN_YEAR]->(d:Date) WHERE ar.name='Društvene znanosti' AND d.year='2019' RETURN COUNT(r1) AS number, a.firstname, a.lastname ORDER BY number desc LIMIT 20
```

Rezultat upita prikazan je na Slika 12.

number	a.firstname	a.lastname
10	"Aleksandar"	"Štulhofer"
9	"Petar"	"Kurečić"
5	"Igor"	"Klopotan"
4	"Ivo"	"Grgić"
4	"Magdalena"	"Zrakić"
4	"Petar"	"Jandrić"
3	"Kristina"	"Petljak"
3	"Marin"	"Milković"
3	"Tihana"	"Škrinjarić"
3	"Josip"	"Burušić"
3	"Marin"	"Čagalj"
3	"Renata"	"Mekovec"
3	"Maja"	"Munivrana Vajda"
3	"Lucija"	"Ivančić"
3	"Vesna"	"Bosilj Vukšić"
3	"Mario"	"Spremić"
3	"Maja"	"Klindžić"
3	"Dora"	"Naletina"
3	"Anela"	"Nikčević-Milković"
2	"Petar"	"Bačić"

Slika 12. Top 20 autora s najviše publikacija u području "Društvene znanosti" (Snimka zaslona, 2019)

Drugi upit prikazan na grafičkom sučelju je jednostavan upit koji prikazuje publikacije u kojima je objavljeno najviše radova općenito ili najviše CC radova. U upitu se pronalazi uzorak koji povezuje čvorove „PublicationFact“, „Publication“ i „Area“ te kao rezultat vraća naziv publikacije i zbroj radova ili CC radova pomoću agregirajuće funkcije *sum*.

Sintaksa upita koji dohvaća nazive top 10 publikacija s najviše objavljenih CC radova u području „Tehničkih znanosti“ je sljedeća:

```
1. MATCH (pf:PublicationFact)-[r1:IN_PUBLICATION]->(p:Publication), (pf)-[r2:IN_AREA]->(ar:Area) where ar.name="Tehničke znanosti" RETURN p.title AS publicationTitle, sum(toInteger(pf.numberOfCCPapers)) AS numberOfCCPapers ORDER BY numberOfCCPapers  DESC LIMIT 10
```

Naravno, s obzirom na model skladišta podataka, može se zaključiti da će se u rezultatu upita dobiti časopisi budući da zbornici konferencija kao publikacije nemaju atribut koji označava broj objavljenih CC radova.

Rezultat upita prikazan je na Slika 13.

publicationTitle	numberOfCCPapers
"Strojarstvo"	629
"High temperature materials and processes"	94
"Journal of applied polymer science"	81
"Journal of thermal analysis and calorimetry"	74
"Journal of sound and vibration"	65
"Renewable Energy"	54
"International journal of simulation modelling"	49
"Engineering computations"	47
"Journal of elastomers and plastics"	44
"Desalination"	41

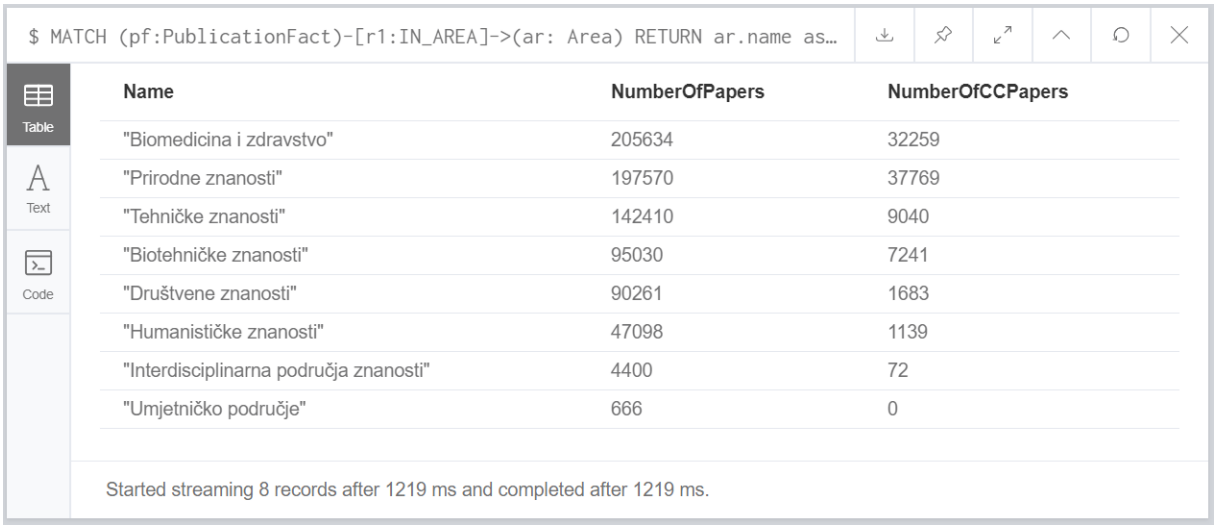
Slika 13. Top 10 publikacija s najviše objavljenih CC radova u području "Tehničke znanosti" (Snimka zaslona, 2019)

Zadnji upit koji je prikazan na grafičkom sučelju je upit koji prikazuje broj radova i broj CC radova prema znanstvenom području. Na Slika 14 je prikazan rezultat upita u *Neo4j*-u. Prvi stupac označava naziv znanstvenog područja, dok druga dva označavaju broj radova i broj CC radova objavljenih u tom području.

Sintaksa samog upita je sljedeća:

```
1. MATCH (pf:PublicationFact)-[r1:IN_AREA]->(ar: Area) RETURN ar.name AS
   Name, sum(toInteger(pf.numberOfPapers)) AS NumberOfPapers,
   SUM(toInteger(pf.numberOfCCPapers)) AS NumberOfCCPapers ORDER BY
   NumberOfPapers DESC
```

Ključna riječ „MATCH“, kao što je već ranije navedeno, pronalazi određen uzorak u grafovskom skladištu podataka. U upitu su promatrane veze *IN_AREA* između čvorova „PublicationFact“ i „Area“. Broj radova i broj CC radova je zapisan u čvoru „PublicationFact“ te se taj broj zbraja (pomoću agregirajuće funkcije *SUM*) i konačno zapisuje u varijablu *NumberOfPapers* i *NumberOfCCPapers,* što je kasnije prikazano i na grafičkom sučelju.



Slika 14. Broj radova i broj CC radova prema području (Snimka zaslona, 2019)

Vidljivo je da područje „Biomedicina i zdravstvo“ te „Prirodne znanosti“ imaju najveći broj CC radova. Razlog tome može se objasniti Pravilnikom o uvjetima za izbor u znanstvena zvanja iz 2013. godine [47] (dostupna i nova verzija iz 2017. godine). Za područje biomedicina i zdravstvo navodi se sljedeće:

„Za izbor u znanstvena zvanja pristupnici moraju imati znanstvene radove objavljene u časopisima koji su zastupljeni u Current Contentsu ili Science Citation Index – Expandedu, od čega dvije trećine radova trebaju biti zastupljene u Current Contentsu prema sljedećem…“

Kod područja „Prirodnih znanosti“, polja „Biologija“, „Fizika“ i „Kemija“ također je definirano da je za izbor u znanstveno zvanje znanstvenog suradnika potrebno imati najmanje dvije trećine od navedenog broja znanstvenih radova objavljeno u časopisima koji su zastupljeni u „Current Contents“ bazi podataka. Kod tehničkih i biotehničkih znanosti broj objavljenih radova se dijeli u skupine te je najčešće prva skupina ona u koju se ubrajaju radovi zastupljeni u bazama podataka „Science Citation Index – Expanded“ i „Current Contents“. Kod društvenih, humanističkih, interdisciplinarnih područja znanosti i umjetničkog područja broj CC radova za izbor u znanstveno zvanje nije uvjetovan pa se time može objasniti manji broj CC radova unutar tih područja.

6.1. Upiti i vizualizacija u obliku programskog sučelja

Konačno rješenje prikazano je u obliku *Flask* web aplikacije koja uključuje primjere nekoliko upita nad *Neo4j* skladištem podataka. Sučelje se sastoji od pomoćnog izbornika u kojem se nalaze primjeri upita. Odabirom pojedinog upita dolazi se do zasebnih sučelja na kojima su upiti prikazani grafičkim prikazima pomoću biblioteke *Plotly*.

Podatke za vizualizaciju dohvatili smo izvršavanjem upita u *Pythonu*, no prije same vizualizacije, trebali smo ih pretvoriti u oblik koji biblioteka *Plotly* razumije. Izvršavanjem upita se od *py2neo* biblioteke dobio kursor na niz zapisa, a pomoću *json* biblioteke su se ti zapisi pretvorili u JSON oblik:


```
[{
    "number": 763,
    "a.firstname": "Josip",
    "a.lastname": "Pečarić"
  },
  {
    "number": 661,
    "a.firstname": "Zlatko",
    "a.lastname": "Šatović"
  },
  {
    "number": 598,
    "a.firstname": "Nada",
    "a.lastname": "Horvatinčić"
  },
  {
    "number": 590,
    "a.firstname": "Ines",
    "a.lastname": "Krajcar-Bronić"
  }
]
```

U nastavku slijedi kratki primjer rezultata dobivenih izvršavanjem upita. Važno je napomenuti da će se pojam grafa u kontekstu ovog poglavlja odnositi na grafički prikaz/dijagram a ne u strukturu spremljenu u skladištu podataka.Budući da smo htjeli da se na grafu, temeljem ovih podataka, prikaže broj radova prema imenu i prezimenu autora, iz ovakvog se oblika podataka uzimao pojedini JSON objekt te se u jednu listu dodavao pojedini broj, a u drugu listu pojedino ime i prezime. Između ostalih informacija o grafu kao što su tip, boja, pozicija teksta i slično, listu brojeva smo postavili kao izvor podataka za y os, a listu imena i prezimena kao izvor podataka za x os. Sve ove informacije su se učahurile u JSON oblik koji je na kraju *JavaScript* zajedno s bibliotekom *Plotly* koristio za prikaz samog grafa.

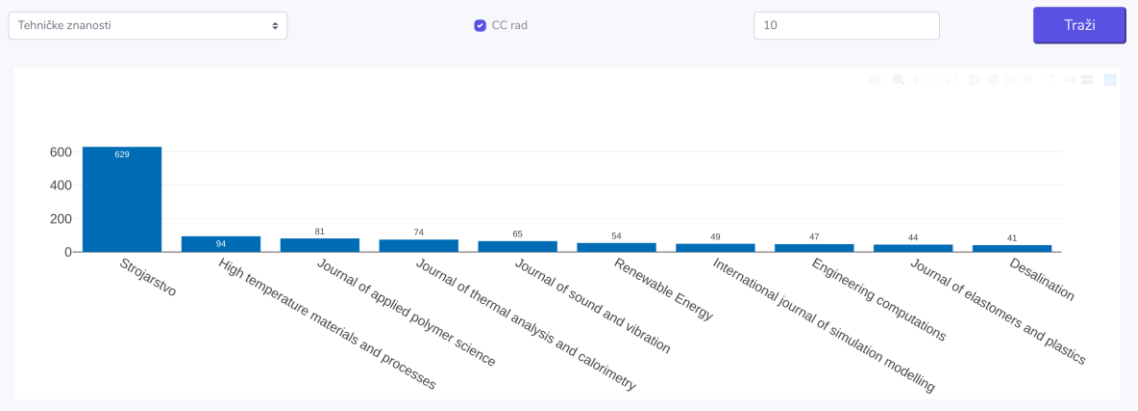
U nastavku na Slika 15 odabran je prvi upit u izborniku koji prikazuje najproduktivnije autore prema području i godini. Na vrhu glavnog dijela sučelja za ovaj upit moguće je odabrati željeno područje i godinu iz padajućih izbornika te unijeti željeni broj autora za prikaz. Pritiskom na gumb „Traži“ generira se prikaz sukladno unesenim podacima, kao što je to prikazano na primjeru Slika 16.

Slika 15. Inicijalni prikaz upita za pretragu autora s najviše objavljenih radova općenito (Snimka zaslona, 2019)

Slika 16. Mjehuričasti grafički prikaz 10 autora s najviše objavljenih radova u 2019. godini u području društvenih znanosti (Snimka zaslona, 2019)

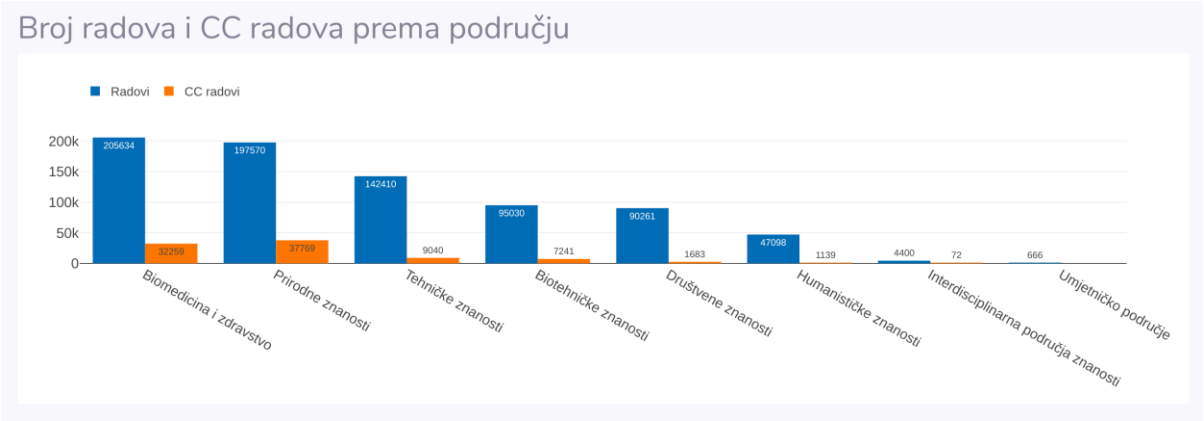
Drugi primjer upita ponovno koristi filtere, no ovaj puta za prikaz publikacija u kojima je objavljeno najviše radova. Za korištenje ovog prikaza potrebno je odabrati željeno područje iz padajućeg izbornika, unijeti željeni broj radova za prikaz te označiti okvir za izbor ukoliko se žele prikazati i CC radovi. U nastavku Slika 17 i Slika 18 prikazuju dva primjera stupčastih grafova koji sukladno unesenim podacima vizualiziraju rezultate pretrage.

Slika 17. Stupčasti grafički prikaz top deset publikacija s najviše objavljenih radova za područje biotehničkih znanosti (Snimka zaslona, 2019)



Slika 18. Stupčasti grafički prikaz top deset publikacija s najviše objavljenih CC radova u području tehničkih znanosti (Snimka zaslona, 2019)

Posljednjim upitom prikazuju se podaci iz skladišta neovisni o parametrima (nema potrebe za korištenjem filtera za pretragu). Upit prikazuje broj radova i CC radova prema određenim područjima u obliku grupiranog stupčastog grafa (Slika 19) u kojem jedan stupac označava obične radove, a drugi CC radove (sukladno legendi grafičkog prikaza).



Slika 19. Grupirani stupčasti grafički prikaz broja radova i CC radova prema područjima (Snimka zaslona, 2019)

7. Rasprava

Objedinjujući pretpostavke hipoteze jasno se mogu izlučiti temeljne zadaće čijim smo rješavanjem nastojali pokazati vrijednosti rezultata istih. Naime, koristeći pristup temeljen na grafovima uspjeli smo zaista izraditi grafovsko skladište čijim smo korištenjem omogućili vividnu reprezentaciju znanstvene produktivnosti u Republici Hrvatskoj i potvrdili hipotezu H1. Spomenuto se zorno očituje u poglavlju 6. u kojima su priloženi upiti čije izvršavanje unutar grafovskog skladišta vraća rezultate glede same znanstvene produktivnosti (mreža istraživača, publikacija i područja) u Republici Hrvatskoj pri iznimnim brzinama zahvaljujući upravo odabranom pristupu.

Slika 12 primjer je upravo takvog rezultata - rezultata koji jasno predočuje autore čija se produktivnost očituje u brojnosti njihovih radova. Dakako, brojnost radova u ni kojem pogledu nije jedini pokazatelj produktivnosti, no držimo da je jedan od najbitnijih te najočitijih. Broj radova pojedinog znanstvenika tijekom dosadašnjeg trajanja njegove znanstvene karijere može svakako dati uvid u produktivnost, no ukoliko se uzmu još pojedini atributi poput područja ili pak vremenskog intervala djelovanja onda se ova produktivnost može smjestiti u određeni kontekst, što pak može pružiti zaista pozamašan uvid u tendencije i interese samog autora. Dodajući na to još i koautore tih radova, ova produktivnost može se posmatrati i unutar pojedinih grupacija znanstvenika u Republici Hrvatskoj - no zadržat ćemo se unutar okvira hipoteze te samih dobivenih rezultata jer oni su ipak empirijski potvrđeni unutar ovog rada. Rezultat upita na koji smo se upravo osvrnuli, htjeli bismo nadodati, dohvaćen je u svega 2 729 ms što se također vidi i na spomenutoj slici. Tomu je tako primarno zbog odabranog pristupa, pristupa temeljenog na grafovima. Implementacija te samo korištenje ovog pristupa ne zahtijeva nizanje velikog broja JOIN klauzula (još poznato pod engleskim nazivom *Join Bombs*), što kao posljedicu daje superiornije izvedbe. Naravno, ukoliko još tome dodamo činjenicu da se radi o skladištu koje se gotovo prema definiciji izrađuje imajući na umu moguće upite od interesa, onda performanse ovih upita još više postaju jasne.

Sljedeći upit koji bismo istaknuli, a čiji smo rezultat priložili u poglavlju 6 jest upit prikazan na Slika 13. Taj upit je prvi upit koji smo naveli, a da ne vraća rezultat vezan uz same autore odnosno znanstvenike kao takve. Smatramo da je ovo bitna činjenica jer to reprezentira mogućnost analize i drugih elemenata znanstvene produktivnosti u Republici Hrvatskoj. Tako je, primjerice, u rezultatu na toj slici vidljivo da on varira ovisno o publikaciji te, globalnije gledano, zasigurno i samom području znanosti. Spomenute varijacije u brojnosti CC radova također su pokazatelj produktivnosti te prepoznatljivosti kvalitete radova unutar pojedinih znanstvenih polja odabranog područja znanosti. Pogledamo li na primjeru publikacija, mogli

bismo odmah reći da polje "Strojarstvo" dominira područjem "Tehničke znanosti" sa čak 629 CC radova, dok ostale publikacije ponešto kaskaju s brojnošću CC radova. Upravo to može biti jedna od polaznih točaka za neko daljnje istraživanje - ovo napominjemo jer naprosto želimo istaknuti spektar mogućnosti daljnjih istraživanja koji proizlazi upravo iz izrađenog grafovskog skladišta podataka.

Govoreći o CC radovima, osvrnimo se na priloženi primjer prikazan na Slika 14 koji raspon interesa podiže na nešto višu razinu. Naime, u tom primjeru prikazana su područja znanosti, broj radova te broj CC radova unutar tih područja, što daje ponešto općenitiji dojam o produktivnosti negoli to daju precizniji upiti poput spomenutog upita vezanog uz publikacije. Iz rezultata ovog upita štošta se dade zaključiti, primjerice vidljivo je da područje "Biomedicina i zdravstvo" kao takvo ima najveći broj radova, što proizlazi iz činjenice da je interes za to područje u Republici Hrvatskoj zaista velik i tome svakodnevno svjedočimo čitajući o brojnim uspjesima upravo na tom području – dakako, ti uspjesi su posljedica rada odnosno produktivnosti znanstvenika u tom području, što je u trenutku pisanja ovog rada nama od interesa. Nadalje, kako je već napomenuto u poglavlju 6.1., može se zamijetiti korelacija između područja "Biomedicina i zdravstvo" te područja "Prirodne znanosti" s brojem CC radova, što je djelomično, kako je spomenuto, vezano uz stari "Pravilnik o uvjetima za izbor u znanstvena zvanja" iz 2013. godine. Ova korelacija te uključenost spomenutog pravilnika također mogu biti polazište za daljnje istraživanje, istraživanje o utjecaju regulativa na znanstvenu produktivnost u Republici Hrvatskoj ili nešto tomu slično.

Samo nekoliko upita, čije rezultate izvršavanja smo priložili unutar ovog rada, u nama su izrodili idejama ili barem inspiracijom za daljnji rad - upravo tu riznicu ideja smatramo jednom od najvažnijih, ako ne i najvažnijim uspijećem ovog grafovskog skladišta podataka. No, to grafovsko skladište podataka ne bi postojalo da prethodno nije izrađena grafovska baza podataka na temelju dobivenog modela mreže znanstvenika. Hipotezu H2 potvrdili smo razvojem interaktivnog grafičkog sučelja pomoću nekoliko tehnologija (Flask, Plotly, itd.), dok smo hipotezu H3 potvrdili u koraku transformacije podataka, gdje smo koristili grafovsku bazu podataka za agregaciju podataka o ukupnom broju objavljenih radova i broju CC radova autora u pojedinom području.

8. Zaključak

Tematika ovog rada vezana je uz domenu hrvatskih znanstvenih publikacija i njihovih autora. U narednim stranicama upoznali smo se s domenom kolaboracijskih mreža znanstvenika te prikazali pregled postojeće literature stranih i hrvatskih autora. Temeljem istraživanja postojeće literature, utvrdili smo da je prirodan prikaz mreže hrvatskih istraživača upravo prikaz pomoću grafa, stoga smo u tu svrhu odlučili koristiti prikladne tehnologije temeljene na grafovima poput grafovskih baza podataka i grafovskih skladišta podataka te istražiti mogućnosti njihove primjene nad spomenutom domenom.

Proveli smo eksperimentalno istraživanje u kojem smo dizajnirali model grafovske baze podataka i grafovskog skladišta podataka te implementirali iste u sustavu Neo4j. Kao izvor podataka koristili smo CROSBI, hrvatsku znanstvenu bibliografiju te smo kroz niz složenih postupaka prikupili te pripremili podatke za realizaciju grafovskog skladišta podataka. Skladište podataka te upiti nad njime su nam koristili kao temelj za analizu produktivnosti hrvatskih istraživača koju smo kasnije prikazali i na izrađenom grafičkom sučelju.

Vjerujemo da je odabrani prikaz mreže hrvatskih istraživača, prikaz pomoću grafovske baze i skladišta podataka vrlo intuitivan te je pogodan za izvršavanje mnogih upita. Rezultate tih upita možemo interpretirati na različite načine te promatrati korelaciju hrvatskih zakona i regulativa te broja objavljenih radova i CC radova. Ovakav prikaz mreže znanstvenika pogodan je za daljnju analizu te vizualizaciju putem grafičkog sučelja.

Vizualizacija podataka nije primarni rezultat ovog rada, već pokazatelj da se znanstvena produktivnost hrvatskih znanstvenika i istraživača može prikazati podacima iz grafovskog skladišta podataka. Tijekom izrade sučelja, zaključili smo da je podatke dobivene upitima jednostavno pretvoriti u oblik razumljiv biblioteci za prikaz grafova.

Za potrebe ovog istraživanja definirali smo tri hipoteze i nekoliko općih i specifičnih ciljeva. Hipoteze smo uspjeli potvrditi izradom i implementacijom modela grafovske baze podataka i grafovskog skladišta podataka te razvojem sučelja za interakciju sa grafovskim skladištem podataka, dok smo ciljeve ostvarili tijekom provedbe ovog istraživanja.

Kako je ovo grafovsko skladište podataka u nama rodilo brojnim idejama, vjerujemo da će svoj inspirativni učinak imati i na druge korisnike - ovakav komad tehnologije - s ovakvim performansama, vjerujemo može uvelike unaprijediti upravo onu produktivnost za čiju je analizu zapravo prvenstveno i izrađen. Dodatno, kako bi se olakšalo korištenje istog izradili smo i aplikativno sučelje čije je korištenje prikazano u poglavlju 6. Daljnja implementacija ovog sučelja te primjena raznih algoritama te analiza na ovom grafovskom skladištu podataka svakako bi bili naši nadolazeći koraci.

U radu smo pokazali kako se izgradnjom grafovskog skladišta podataka mogu na intuitivan način reprezentirati znanstvene publikacije istraživača u Republici Hrvatskoj. Izradom korisničkog sučelja omogućili smo kontrolu nad upitima te vizualni prikaz rezultata odabranog upita u realnom vremenu čime se pojednostavljuje izvršavanje upita. U daljnjem istraživanju je potrebno ubrzati i pojednostaviti te ako je moguće automatizirati ETL proces kako bi podaci u skladištu podataka bili ažurni i bez potrebne za intervencijom. Drugi cilj daljnjeg istraživanja je povećanje broja upita koji se izvršavaju nad podacima u skladištu kako bi se mogao dobiti širi uvid te omogućila detaljnija analiza znanstvenih publikacija.

Popis literature

[1] “CROSBI - O CROSBI-ju.” .

[2] “CROSBI - Statistika.” .

[3] “Što je Dabar? | Digitalni akademski arhivi i repozitoriji.” .

[4] “Repozitorij Fakulteta organizacije i informatike.” .

[5] “Hrčak Portal znanstvenih časopisa Republike Hrvatske.” .

[6] N. Frančula, *Geodetski list glasilo Hrvatskoga Geodetskog Društva.*, vol. 71, no. 3. 2005.

[7] D. Xiang and H. Li, “Analyzing international scientific collaboration pattern for China by using ESI database,” in *IEEE International Conference on Industrial Engineering and Engineering Management*, 2012, pp. 1386–1390.

[8] M. Li, J. Wu, D. Wang, T. Zhou, Z. Di, and Y. Fan, “Evolving model of weighted networks inspired by scientific collaboration networks,” *Phys. A Stat. Mech. its Appl.*, vol. 375, no. 1, pp. 355–364, Feb. 2007.

[9] A. Kumari, S. N. Singh, A. Bihari, and S. Tripathi, “Key community analysis in scientific collaboration network,” in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, 2017, pp. 675–680.

[10] L. Alcón, L. Faria, C. M. H. de Figueiredo, and M. Gutierrez, “Split Clique Graph Complexity,” Springer, Berlin, Heidelberg, 2011, pp. 11–22.

[11] X. Shi, L. Cai, and J. Jia, “The evolution of international scientific collaboration in fuel cells during 1998-2017: A social network perspective,” *Sustain.*, vol. 10, no. 12, 2018.

[12] G. J. Abel, R. Muttarak, V. Bordone, and E. Zagheni, “Bowling Together: Scientific Collaboration Networks of Demographers at European Population Conferences,” *Eur. J. Popul.*, pp. 1–20, 2018.

[13] B. P. F. E Fonseca, R. B. Sampaio, M. V. A. Fonseca, and F. Zicker, “Co-authorship network analysis in health research: Method and potential use,” *Heal. Res. Policy Syst.*, vol. 14, no. 1, 2016.

[14] F. G. Montoya, A. Alcayde, R. Baños, and F. Manzano-Agugliaro, “A fast method for identifying worldwide scientific collaborations using the Scopus database,” *Telemat. Informatics*, vol. 35, no. 1, pp. 168–185, Apr. 2018.

[15] N. Gaskó, R. I. Lung, and M. A. Suciu, “A new network model for the study of scientific collaborations: Romanian computer science and mathematics co-authorship networks,” *Scientometrics*, vol. 108, no. 2, pp. 613–632, 2016.

[16] R. Lara-Cabrera, C. Cotta, and A. J. Fernández-Leiva, “An analysis of the structure and evolution of the scientific collaboration network of computer intelligence in games,” *Phys. A Stat. Mech. its Appl.*, vol. 395, pp. 523–536, 2014.

[17] J. Kim and J. Diesner, “The effect of data pre-processing on understanding the evolution of collaboration networks,” *J. Informetr.*, vol. 9, no. 1, pp. 226–236, 2015.

[18] Y. Ding, “Scientific collaboration and endorsement: Network analysis of coauthorship and citation networks,” *J. Informetr.*, vol. 5, no. 1, pp. 187–203, 2011.

[19] M. E. J. Newman, “The structure of scientific collaboration networks.” 2000.

[20] W. Jakawat, C. Favre, and S. Loudcher, “OLAP cube-based graph approach for bibliographic data,” *CEUR Workshop Proc.*, vol. 1548, pp. 87–99, 2016.

[21] G. Madaan and S. Jolad, “Evolution of scientific collaboration networks,” in *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014*, 2015, pp. 7–13.

[22] B. Grba and A. Mestrovic, “Tracking the evolution of scientific collaboration networks,” in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018 - Proceedings*, 2018, pp. 503–508.

[23] T. Aparac and F. Pehar, “Information sciences in Croatia: A view from the perspective of bibliometric analysis of two leading journals,” in *The Janus Faced Scholar: A Festschrift in Honour of Peter Ingwersen*, Royal School of Library and Information Science, 2010.

[24] A. Vukotic and N. Watt, Neo4j i*n Action*. Shelter Island: Manning Publications, 2014.

[25] Neo4j, “Neo4j,” 2019. [Online]. Available: https://Neo4j.com/. [Accessed: 21-Apr-2019].

[26] J. Baton and R. Van Bruggen, *Learning* Neo4j *3.x - Second Edition*. Packt, 2017.

[27] E. Emil, Robinson Ian, Webber Jim, *Graph Databases, 2nd Edition: New Opportunities for Connected Data*. O’Reilly Media, 2015.

[28] “Neo4j Cypher Manual - Chapter 5. Schema.”

[29] N. Inc., “openCypher.” .

[30] “Neo4j Cypher Manual.” .

[31] P. Ponniah, *Data Warehousing Fundamentals*. Wiley India Pvt. Limited, 2006.

[32] R. Kimball and M. Ross, *The data warehouse toolkit: the complete guide to dimensional modelling*. New York, USA: Wiley, 2002.

[33] R. Kimball and M. Ross, *The Data Warehouse Toolkit Third Edition*. 2013.

[34] S. A. Rahim, B. Chakraborty, J. Debnath, and N. Debnath, "Design Graph Multi-Dimensional Data Model of a Data Warehouse and conversion of its equivalent Object - Oriented Schema," in *2013 IEEE Symposium on Computers and Communications (ISCC)*, 2013, pp. 000079–000084.

[35] Y. Liu and T. M. Vitolo, "Graph Data Warehouse: Steps to Integrating Graph Databases Into the Traditional Conceptual Structure of a Data Warehouse," in *2013 IEEE International Congress on Big Data*, 2013, pp. 433–434.

[36] CROSBI, "CROSBI - O CROSBI-ju," 2019.

[37] Anaconda Inc., "Anaconda," 2019. [Online]. Available: https://www.*Anaconda*.com/. [Accessed: 25-Mar-2019].

[38] Project Jupyter, "Jupyter," 2019. [Online]. Available: https://jupyter.org/. [Accessed: 25-Feb-2019].

[39] A. Ronacher, "Flask," 2018. [Online]. Available: https://www.palletsprojects.com/p/Flask/. [Accessed: 17-Mar-2019].

[40] A. Ronacher, "Werkzeug biblioteka," 2019. [Online]. Available: https://palletsprojects.com/p/werkzeug/. [Accessed: 02-Apr-2019].

[41] A. Ronacher, "Jinja2 stroj za predloške," 2008. [Online]. Available: http://jinja.pocoo.org/. [Accessed: 02-Apr-2019].

[42] N. Small, "Py2neo biblioteka," 2019. [Online]. Available: https://py2neo.org/v4/index.html. [Accessed: 10-Apr-2019].

[43] Neo4j, "Python biblioteke za upravljanje Neo4j bazom podataka," 2019. [Online]. Available: https://Neo4j.com/developer/Python/#Neo4j-Python-driver. [Accessed: 10-Apr-2019].

[44] *Plotly*, "*Plotly* biblioteka," 2019. [Online]. Available: https://plot.ly/Python/. [Accessed: 27-Apr-2019].

[45] M. Brinkmann, "Flask-bootstrap biblioteka," 2013.

[46] "Bootstrap predložak otvorenog koda SB Admin 2."

[47] M. Ivezić, “Zastarjeli pravilnik o uvjetima za izbor u znanstvena zvanja,” 2013. [Online]. Available: http://www.pbf.unizg.hr/stari_pravilnik_o_uvjetima_za_izbor_u_znanstvena_zvanja_narodne_novine_broj_84_05_100_06_136_06_42_07_odluka_usrh_120_07_71_10_116_10_i_38_11_nesluzbeni_procisceni_tekst2. [Accessed: 01-May-2019].

Sažetak

Znanstveni radovi, njihova izrada i objava čine ključni dio napretka svakog znanstvenika te doprinosi napretku cijele znanstvene zajednice. Analiza mreže znanstvenika i kolaborativnih mreža postala je tematika brojnih znanstvenih i stručnih radova. Cilj ovog rada je prikazati trenutno dostupne tehnologije temeljene na grafovima koje se mogu upotrijebiti u domeni mreža znanstvenika zbog njihove reprezentacije podataka u obliku grafova. U radu su definirane tri hipoteze vezane uz mogućnost primjene pristupa temeljenog na grafovima za reprezentaciju domene te izgradnje grafovskog skladišta podataka prikladnog za daljnju analizu i uz zadovoljavajuće performanse. U radu su najprije analizirani različiti pristupi reprezentacije i analize mreža znanstvenika u Republici Hrvatskoj i svijetu. U daljnjem sadržaju rada opisana je teorija vezana uz grafovske baze podataka, jedan od najraširenijih sustava za upravljanje grafovskim bazama podataka *Neo4j* te deklarativni upitni jezik *Cypher*, kao i skladišta podataka. Najveći doprinos ovog rada predstavlja model grafovskog skladišta podataka koji omogućuje reprezentaciju mreže znanstvenih publikacija i istraživača, a koji je implementiran u sustavu *Neo4j* i služi kao izvor podataka za daljnje analize. Korištenjem nekoliko tehnologija (*Flask*, *Plotly* itd.) izrađeno je grafičko sučelje koje omogućuje analizu znanstvenih publikacija hrvatskih istraživača u obliku upita nad grafovskim skladištem podataka.

Ključne riječi: mreža znanstvenika, kolaboracijska mreža znanstvenika, grafovska baza podataka, sustav za upravljanje grafovskom bazom podataka, grafovsko skladište podataka, ETL

Summary

Writing scientific papers is a key part of every scientist's progress and their publication contributes to the advancement of the entire scientific community. Scientific network analysis and scientific collaboration network analysis have become the subject of numerous scientific papers. The goal of this paper is to present currently available graph-based technologies, which can be used in the scientific networks domain due to their data representation in the form of graphs. Three hypotheses have been specified related to the possibility of applying graph-based approach to domain representation and building a graph data warehouse suitable for further analysis and with stable performance. In the paper, different collaborative scientific network analysis approaches are presented as an overview of published papers authored by foreign and Croatian authors. In the following pages, we describe graph database theory, one of the most widely used graph database management systems *Neo4j* and *Cypher* declarative graph query language, as well as data warehouses. The most important contribution of this paper is the graph data warehouse model, which enables the representation of scientific publications and researchers network, it is implemented in *Neo4j* and can be used as a data source for further analysis. By using various technologies (e.g. *Flask*, *Plotly*, etc.) a graphical interface has been developed, which enables the analysis of scientific publications of Croatian researchers in the form of graph data warehouse queries.

Keywords: scientific network, collaboration scientific network, graph database, graph database management system, graph data warehouse, ETL

Zahvale

Zahvaljujemo našem mentoru prof. dr. sc. Korneliju Rabuzinu koji je svojim savjetima i preporukama neizmjerno pomogao u povećanju kvalitete izrade rada te što je uvijek imao vremena, želje i strpljenja odgovarati na naše brojne upite.

Veliku zahvalnost izražavamo Martini Šestak, mag. inf. koja je svojim znanjem i iskustvom pomogla pri izradi znanstvenog i stručnog dijela ovog rada te na nesebičnoj pomoći, ustupljenim materijalima, susretljivosti i oblikovanju konačnog sadržaja rada.