Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.1:766
DOI: https://doi.org/10.24867/15BE38Tarjan

SISTEM ZA DISTRIBUIRANU DETEKCIJU PLAGIJARIZMA

DISTRIBUTED PLAGIARISM DETECTION SYSTEM

Tamaš Tarjan, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SOFTVERSKO INŽENJERSTVO I INFORMACIONE TEHNOLOGIJE

Kratak sadržaj – *U radu je predstavljen sistem za distribuiranu detekciju plagijarizma. Sistem nudi centralizovani pristup distribuiranim čvorovima koji su prilagođeni protokolu za distribuiranu proveru plagijarizma. U ovom radu je dat opis navedenog protokola. Distribuirani čvorovi imaju mogućnost da se priključe u sistem pomoću registra servisa. Prilikom registracije čvora u sistem, izvršava se osnovna provera poštovanja protokola simulirajući očekivani tok komunikacije između datog čvora i sistema. Svaki čvor ima svoje skladište za digitalne dokumente. Dokumenti nisu deljeni između članova sistema.*

Ključne reči: *Distribuirani sistem, detekcija plagijarizma, obrada dokumenata*

Abstract – *This paper describes a system for distributed plagiarism detection. The system offers centralized access to distributed nodes that adhere to a protocol for distributed plagiarism detection that is described in this paper. Distributed nodes can join the system through service discovery. Adherence to the mentioned protocol is verified during node registration by simulating the expected request / response communication flow. Each node has its own digital document storage implementation. Documents are not shared between the system nodes.*

Keywords: *Distributed system, plagiarism detection, document processing*

1. UVOD

Plagijarizam predstavlja potpunu ili delimičnu upotrebu tuđih dela i misli, i predstavljanje istih kao sopstvene ideje.

U današnje vreme sve je veći broj domena u kojima postoji potreba za skladištenjem neke vrste digitalne intelektualne svojine. Samim tim broj ustanova koji imaju svoje privatne sisteme za skladištenje navedenog digitalnog sadržaja je velik. Data skladišta digitalnih intelektualnih svojina mogu biti javna ili privatna.

Jedan od glavnih problema prilikom utvrđivanja plagijata je to što je broj izvora iz kojih može da potiče deo teksta koji je plagiran previše velik.

Sistemi za proveru plagijarizma imaju pristup samo relativno malom podskupu relevantnog sadržaja.

Određeni broj ustanova pored skladištenja digitalnih dokumenata takođe imaju mogućnost provere plagijarizma u okviru njihovog skupa dokumenata.

Distribuirani sistem je sistem čije komponente se nalaze na različitim umreženim računarima koji komuniciraju i orkestriraju svoje akcije pomoću slanja poruka jedni drugima [1].

U ovom radu je predstavljen sistem koji pruža mogućnost postojećim skladištima dokumenata da se priključe u distribuirani anonimizovan sistem za detekciju plagijarizma bez deljenja svojih digitalnih dokumenata.

2. TEORIJSKE OSNOVE

Način komunikacije između čvorova u distribuiranom sistemu zavisi od topologije datog sistema. U ovom sistemu postoji jedan centralni čvor koji komunicira sa ostalim učesnicima i omogućuje korisniku da vrši proveru plagijarizma na proizvoljnim čvorovima sistema. U narednom delu su predstavljeni ključni pojmovi za opis ovog sistema.

2.1. Dokument

U ovom radu pojam dokumenta se definiše kao digitalna jedinica obrade koja predstavlja intelektualnu svojinu.

2.2. Metapodaci dokumenta

Metapodaci dokumenta predstavljaju podatke o dokumentu. Primeri metapodataka su naziv dokumenta, autori, tip dokumenta, veličina dokumenta, lokacije različitih reprezentacija dokumenta, itd.

2.3. Reprezentacija dokumenta

Format originalne reprezentacije dokumenta koja se skladišti nakon uspešnog otpremanja na sistem, ne mora da bude kompatibilna sa određenim algoritmom za detekciju plagijarizma, bez obzira da li konceptualni tip dokumenta odgovora datom algoritmu (na primer tekstualni dokument u PDF formatu).

Da bi detekcija plagijarizma za dati dokument bila moguća, treba kreirati određenu reprezentaciju sa kojom algoritam za detekciju plagijarizma može da rukuje.

2.4. Predprocesiranje dokumenata

Predprocesiranje dokumenta predstavlja proces u kojem se na osnovu ulaznog dokumenta dobija određeni izlaz. Izlaz predprocesiranja može biti određena reprezentacija dokumenta ili metapodatak koji je dobijen na osnovu ulaznog dokumenta.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivanović, red. prof.

2.5. Detekcija plagijarizma

Detekcija plagijarizma predstavlja proces u kojem se kandidat dokument upoređuje sa svakim dokumentom iz referentnog skupa dokumenata. Referentni dokumenti su dokumenti koji pripadaju referentnom skupu. U slučaju da postoji značajno preklapanje između kandidata i referentnog dokumenta, kandidat se markira kao potencijalni plagijat.

2.6. Referentni skup

U idealnom slučaju referentni skup je skup svih relevantnih dokumenata sa kojima treba uporediti kandidat dokument da bi utvrdili da li je dati dokument plagijat. Ovaj skup možemo nazvati apsolutnim referentnim skupom. U praktičnom smislu nemamo mogućnost da pristupimo svim dokumentima iz ovog skupa, to znači da je suštinski nemoguće dokazati da određeni dokument nije plagijat. Iz tog razloga fokus je na dokazivanju da dati dokument jeste plagijat, zato što je to potencijalno izvodljiv zadatak. Skup koji sadrži dokumente sa kojima upoređujemo kandidat dokument možemo nazvati realnim referentnim skupom (ili samo referentnim skupom).

Ako pretpostavimo da imamo algoritam koji sa sigurnošću upoređuje dva data dokumenta i daje procenu da li ja kandidat dokument plagijat, tada verovatnoća da se detektuje plagijarizam kod datog dokumenta zavisi od broja dokumenata u referentnom skupu i od broja dokumenata koji su bili plagirani u datom kandidat dokumentu. Ovu verovatnoću možemo maksimizovati maksimizacijom broj dokumenata u referentnom skupu.

3. SPECIFIKACIJA SISTEMA

U ovom poglavlju je opisana specifikacija centralizovanog sistema koja komunicira sa krajnjim čvorovima koji vrše proveru plagijarizma u okviru svog referentnog skupa dokumenata. Čvorovi za detekciju plagijarizma moraju da se prilagode protokolu koji je opisan u nastavku.

3.1. Protokol za distribuiranu detekciju plagijarizma

Protokol za distribuiranu detekciju plagijarizma posmatra krajnje čvorove kao REST resurse. Komunikacija sa datim čvorovima se vrši isključivo preko HTTP prokola upotrebom JSON poruka. Krajnje tačke koje čvorovi moraju da podrže su navedene na listinzima 3.1.1. - 3.1.3.

```
Request: GET <BASE_URL>/api/v1/algorithms
Response:
[
    {
        "id": "1",
        "name": "<algorithm-name>",
        "description": "<algorithm-description>",
        "supportedMimeTypes": [
            "text/plain"
        ]
    }
]
```

Listing 3.1.1. *Zahtev za dobavljanje podržanih algoritama za detekciju plagijarizma*

```
Request: POST <BASE_URL>/api/v1/tasks HTTP/1.1
Content-Type: multipart/form-data; boundary=----FormBoundary
Content-Length: <content-length>
----FormBoundary
Content-Disposition: form-data; name="file"; filename="file-path"
Content-Type: application/pdf
(data)
----FormBoundary
Content-Disposition: form-data; name="algorithmId"
(algorithm-id)
----FormBoundary
Response:
{
    "id": "<task-id>",
    "status": "<status>",
    "algorithmId": "<algorithm-id>",
    "candidateDocumentId": "<id assigned by the node>",
    "candidateDocumentMd5": "<MD5 string>"
}
```

Listing 3.1.2. *Zahtev za kreiranje zadatka za proveru plagijarizma*

```
Request: GET /api/v1/task/<taskId>
Response:
{
    "id": "<task-id>",
    "status": "<status>",
    "algorithmId": "<algorithm-id>",
    "candidateDocumentId": "<id assigned by node>",
    "candidateDocumentMd5": "<MD5 string>",
    "results": [
        {
            "referenceDocumentId": "<id assigned by node>",
            "referenceDocumentMd5": "<MD5 string>",
            "score": <similarity-score>
        }
    ]
}
```

Listing 3.1.3. *Zahtev za proveru statusa zadatka za proveru plagijarizma*

Rezultati koji se mogu dobiti prilikom provere statusa zadatka ne sadrže konkretne informacije o dokumentu osim identifikatora koji je dodeljen dokumentu od strane čvora i MD5 (eng. Message-digest algorithm) checksum vrednosti radi identifikacije dokumenta. Čvorovi mogu opciono da vrate listu sličnih delova dokumenata za svaki rezultat, međutim i u tom slučaju šalju samo MD5 checksum vrednost od datih delova. Ako se utvrdi da postoje slični dokumenti onda su navedene checksum vrednosti dovoljne da se plagirani dokumenti identifikuju.

Čvorovi moraju da podrže *simulate* parametar (eng. query parameter) za svaki zahtev koji se prosleđuje prilikom testiranja čvora. U slučaju da je ovaj parametar naveden u nekom zahtevu koji je specificiran od strane protokola čvor treba da vrati validan tip podataka za svako polje koje može da se pojavi u odgovoru na zahtev.

3.2. Servis za detekciju plagijarizma

Servis za detekciju plagijarizma predstavlja primer implementacije čvora koji je prilagođen protokolu za distribuiranu proveru plagijarizma. Pruža mogućnost za skladištenje tekstualnih dokumenata i njihovih metapodataka kao i osnovno predprocesiranje koje je potrebno za detekciju plagijata tekstualnih dokumenata. Čvor predstavlja serversku stranu klijent-server komunikacije koja opslužuje zahteve specificirane protokolom koji je opisan u poglavlju 3.1.

3.3. Registar servisa za detekciju plagijarizma

Registar servisa je veb aplikacija koja pruža mogućnost čvorovima da se registruju za učestvovanje u procesu distribuirane provere plagijarizma. Kada neki čvor pošalje zahtev za registraciju registar vrši simulirani tok komunikacije sa datim čvorom da bi utvrdio validnost datog čvora. U slučaju da čvor ne poštuje minimalne zahteve prilikom simuliranog toka komunikacije, registracija čvora se otkazuje.

Registar servisa pruža mogućnost centralizovanom servisu za detekciju plagijarizma da dobije lokacije registrovanih čvorova i da uspostavi komunikaciju sa datim čvorovima.

3.4. Servis za upravljanje distribuiranim zadacima provere plagijarizma

Zadatak ovog servisa je da ponudi objedinjeni interfejs odnosno API (eng. Application Programming Interface) za upravljanje distribuiranim zadacima provere plagijarizma. Ovaj servis predstavlja klijenta koji komunicira sa krajnjim čvorovima upotrebom protokola koji je opisan u poglavlju 3.1.

Ovaj servis obezbeđuje korisniku sledeće funkcionalnosti: pregled liste čvorova koji su u određenom trenutku dostupni za izvršavanje distribuirane detekcije plagijarizma, kao i liste algoritama koje navedeni čvorovi nude. Pored toga, servis nudi krajnje tačke za upravljanje distribuiranim zadacima. Ovo podrazumeva kreiranje zadataka koji su deljeni između proizvoljnog broja čvorova kao i proveru statusa i eventualnih rezultata datih zadataka. Zadaci kreirani od strane jednog korisnika u okviru ovog servisa su dostupni samo datom korisniku.

3.5. Korisnički interfejs

Korisnički interfejs nudi funkcionalnosti za registraciju i prijavu korisnika, pregled dostupnih čvorova za detekciju plagijarizma kao i kontrole za upravljanje zadacima detekcije plagijarizma. Korisnički interfejs zajedno sa servisom za upravljanje distribuiranim zadacima provere plagijarizma pruža mogućnost da se na jednostavan način izvrši otpremanje kandidat dokumenta na odabrane krajnje čvorove koji će vršiti proveru plagijarizma na svojim referentnim skupovima. Korisnik ima mogućnost pregleda rezultata u okviru korisničkog interfejsa.

4. IMPLEMENTACIJA SISTEMA

U ovom poglavlju je dat opis implementacije sistema za upravljanje distribuiranim zadacima detekcije plagijarizma kao i primer sistema koji predstavlja jednu implementaciju protokola koji je opisan u trećem poglavlju ovog rada.

Servis za upravljanje distribuiranim zadacima provere plagijarizma, kao i primer krajnjeg čvora koji vrši proveru plagijarizma su implementirani u obliku veb aplikacija pomoću Java programskog jezika upotrebom Spring Boot radnog okvira.

4.1. Servis za upravljanje distribuiranim zadacima provere plagijarizma

Da bi izbegli potrebu da *frontend* aplikacija odnosno korisnički interfejs komunicira direktno sa pojedinačnim čvorovima koji učestvuju u distribuiranom sistemu, implementiran je servis koji nudi jednostavan API za lako upravljanje distribuiranim procesom za detekciju plagijarizma uz oslonac na *Backend for frontends* šablon. *Backend for frontends* šablon se koristi u kontekstu modernih mikroservisa za objedinjavanje različitih API-a koje odvojeni mikroservisi nude [9]. Sličan pristup je upotrebljen prilikom implementacije ovog servisa.

Entiteti u okviru ovog servisa su skladišteni u MongoDB bazi podataka. Entitet koji predstavlja jedan distribuirani zadatak detekcije plagijarizma u okviru ovog servisa se sastoji od niza podzadataka koji su asocirani sa pojedinačnim čvorovima koji vrše detekciju plagijarizma. Podzadaci su asocirani sa kandidat dokumentom preko njegove MD5 checksum vrednosti.

Kada korisnik kreira distribuirani zadatak, ovaj servis šalje po jedan zahtev čvorovima koji su specificirani u korisnikovom zahtevu. Ovim zahtevom krajnji čvorovi dobijaju sve potrebne informacije da bi izvršili traženi zadatak detekcije plagijarizma.

Kada korisnik pošalje zahtev za proveru statusa i rezultata, ovaj servis po potrebi šalje zahtev krajnjim čvorovima i prikuplja status i rezultate datih podzadataka nakon čega vraća objedinjeni odgovor korisniku.

4.1.1. Registar servisa za detekciju plagijarizma

Registar servisa je veb aplikacija implementirana uz oslonac na S*erver-side Service Discovery* šablon. Naziv, opis i lokacija čvorova za detekciju plagijarizma koji se registruju u registar servisa se skladišti u MongoDB bazi podataka. Servis za upravljanje distribuiranim zadacima za detekciju plagijarizma ima mogućnost da dobije navedene podatke o čvorovima koji su dostupni u mreži.

4.1.2. Korisnički interfejs

Korisnički interfejs je implementiran u Java 1.8 programskom jeziku upotrebom Vaadin 21.0.0.beta2 radnog okvira za izradu *frontend* aplikacija. Rezultati sadrže eksterne identifikatore referentnih dokumenata i kandidat dokumenta, kao i njihovu MD5 checksum vrednost. Navedeni identifikatori su dodeljeni od strane čvorova koji su vršili proveru.

4.2. Primer čvora za detekciju plagijarizma

U ovom delu je opisana implementacija primera krajnjeg čvora koji ima mogućnost da se priključi distribuiranom sistemu za proveru plagijarizma.

Prilikom pokretanja servisa automatski se izvršava pokušaj prijave na registar servisa za detekciju plagijarizma. Lokacija registra se definiše pomoću konfiguracionog parametra. Primer zahteva za prijavu na registar servisa je prikazan na listingu 4.2.1.

```
Request: POST <BASE_URL>/api/v1/services
Content-Type: application/json
{
    "port": 7771,
    "description": "Primer cvora za detekciju plagijarizma v1.0.0"
}
```

Listing 4.2.1. *Primer zahteva za prijavu na registar servisa*

4.2.1. Skladištenje dokumenata i metapodataka

Ovaj servis podržava obradu i skladištenje tekstualnih dokumenata predstavljenih PDF datotekom ili datotekom koja sadrži niz kodiranih karaktera pomoću UTF-8 (eng. Unicode Transformation Format) [2] standarda za kodiranje karaktera.

Prilikom otpremanja, servis skladišti originalnu reprezentaciju dokumenta kojoj se dodeljuje novi naziv u obliku UUID (eng. Universally Unique Identifier) vrednosti, nakon čega započinje proces obrade.

4.2.2. Obrada dokumenata

Obrada jednog dokumenta u okviru ovog servisa podrazumeva utvrđivanje formata datoteke. Format datoteke se predstavlja *MIME type* identifikatorom formata datoteki. Validne vrednosti *MIME type* identifikatora su definisane od strane *IANA* (eng. Internet Assigned Numbers Authority) organizacije [3]. Detektovanje *MIME type* identfikatora se vrši pomoću *Apache Tika* Java [4] biblioteke. U

slučaju da je originalna reprezentacija dokumenta predstavljena PDF datotekom, servis vrši ekstrakciju tekstualnog sadržaja pomoću *Apache PDFBox* [5] Java biblioteke. Ekstraktovani tekst se skladišti u zasebnoj datoteci čiji sadržaj je kodiran pomoću *UTF-8* kodiranja. Lokacija ekstraktovane reprezentacije se čuva u skupu metapodataka otpremljenog dokumenta. Metapodaci o dokumentu se čuvaju u MongoDB bazi podataka.

4.2.3. Implementacija protokola za distribuiranu detekciju plagijarizma

U skladu sa protokolom opisanim u trećem poglavlju ovog rada, implementirane su sve krajnje tačke koje su potrebne ovom servisu kako bi mogao da se priključi u distribuiranu mrežu za detekciju plagijarizma. Priključenje u distribuiranu mrežu uključuje i podršku da se simulira tok komunikacije sa ciljem da se utvrdi kompatibilnost prema datom protokolu. Simulirani odgovori na zahteve koji su poslati sa *simulate* parametrom su generisani uz pomoć *PODAM (eng. Pojo Data Mocker)* Java biblioteke [6] koja pruža mogućnost nasumičnog generisanja vrednosti za polja koja se mogu naći u odgovoru na određene zahteve koji su specificirani od strane protokola. Vrednosti u poljima odgovora će uvek biti ispravnog tipa. Za polja koja su tipa *JSON* niza se generiše po pet slučajnih elemenata.

4.2.4. Detekcija plagijarizma

Algoritam za proveru plagijarizma između tekstualnih datoteka je implementiran u Python 3.9.5 programskom jeziku pomoću *Scikit-learn* [7] biblioteke uz oslonac na *TfIdf* (eng. Term frequency-inverse document frequency) vektorizaciju i kosinusnu sličnost (eng. Cosine Similarity) između dobijenih vektora koji predstavljaju parove upoređenih dokumenata.

U ovom radu je fokus na distribuiranoj arhitekturi za detekciju plagijarizma i shodno tome se koristi pojednostavljena verzija algoritma koji je opisan u radu [8] gde su autori predstavili pristup za detekciju plagijarizma koji je baziran na upotrebi *TfIdf* vektorizacije i kosinusne sličnosti. Čvorovi koji implementiraju algoritme sa boljim performansama za veći broj tipova dokumenata se mogu na lak način priključiti u sistem.

5. ZAKLJUČAK

Broj digitalnih dokumenata kao i broj različitih skladišta za date dokumente svake godine raste zbog uvođenja digitalizacije u veliki broj domena u kojima se rukuje dokumentima. U ovom radu je predstavljen sistem koji za cilj ima objedinjavanje referentnih skupova dokumenata koji učestvuju u detekciji plagijarizma radi povećanja pokrivenosti apsolutnog referentnog skupa prilikom provere plagijarizma.

5.1. Dalji razvoj sistema

Trenutna implementacija sistema pruža mogućnost automatizovane distribuirane provere plagijarizma, međutim u slučaju da se pronađu potencijalni plagirani dokumenti, ne postoji proces pomoću kojeg bi moglo da se manuelno proveri da li su dati dokumenti zaista bili plagirani. Razlog ovome je zahtev da referentni dokumenti koji učestvuju u distribuiranoj detekciji plagijarizma ostanu privatni. Jedan način da se ovaj problem delimično reši je da protokol za distribuiranu proveru plagijarizma podrži deljenje podskupa dokumenata iz referentnog skupa. Vlasnici pojedinačnih čvorova bi trebali da odluče koji dokumenti su javni a koji privatni. Ovaj pristup i daje ima nedostatak kada neki čvor detektuje sličnost u privatnom referentnom dokumentu. Jedan od načina da se ovaj problem reši je uvođenje poslovnog procesa u kojem učesnici na određeni način ručno verifikuju da li je dati dokument plagiran.

6. LITERATURA

[1] Van Steen, M. and Tanenbaum, A., 2002. Distributed systems principles and paradigms. Network, 2, p.28.

[2] Allen, J.D., Anderson, D., Becker, J., Cook, R., Davis, M., Edberg, P., Everson, M., Freytag, A., Iancu, L., Ishida, R. and Jenkins, J.H., 2012. The unicode standard. Mountain view, CA.

[3] Iana.org. (2019). Media Types. [online] Dostupno na: https://www.iana.org/assignments/media-types/media-types.xhtml. [Pristupljeno 11. 8. 2021]

[4] tika.apache.org. (n.d.). Apache Tika – Apache Tika. [online] Dostupno na: https://tika.apache.org/. [Pristupljeno 11. 8. 2021]

[5] pdfbox.apache.org. (n.d.). Apache PDFBox | A Java PDF Library. [online] Dostupno na: https://pdfbox.apache.org/. [Pristupljeno 11. 8. 2021]

[6] mtedone.github.io. (n.d.). Podam - Welcome to Jemos PODAM (POjo DAta Mocker). [online] Dostupno na: https://mtedone.github.io/podam/. [Pristupljeno 11. 8. 2021]

[7] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, pp.2825-2830.

[8] Chavan, H., Taufik, M., Kadave, R. and Chandra, N., 2021. Plagiarism Detector Using Machine Learning. International Journal of Research in Engineering, Science and Management, 4(4), pp.152-154.

[9] Brown, K. and Woolf, B., 2016, October. Implementation patterns for microservices architectures. In Proceedings of the 23rd Conference on Pattern Languages of Programs (pp. 1-35).

Kratka biografija:

Tamaš Tarjan rođen je 6. decembra 1996. godine u Novom Sadu. Godine 2015. upisao je Fakultet tehničkih nauka u Novom Sadu smer Softversko inženjerstvo i informacione tehnologije. 2019. godine diplomirao je na osnovnim studijama na Fakultetu tehničkih nauka, iste godine upisuje master studije na smeru Softversko inženjerstvo i informacione tehnologije.

Kontakt: tarjan.nt@gmail.com